Machine Learning in High Energy Physics II

Benjamin Nachman

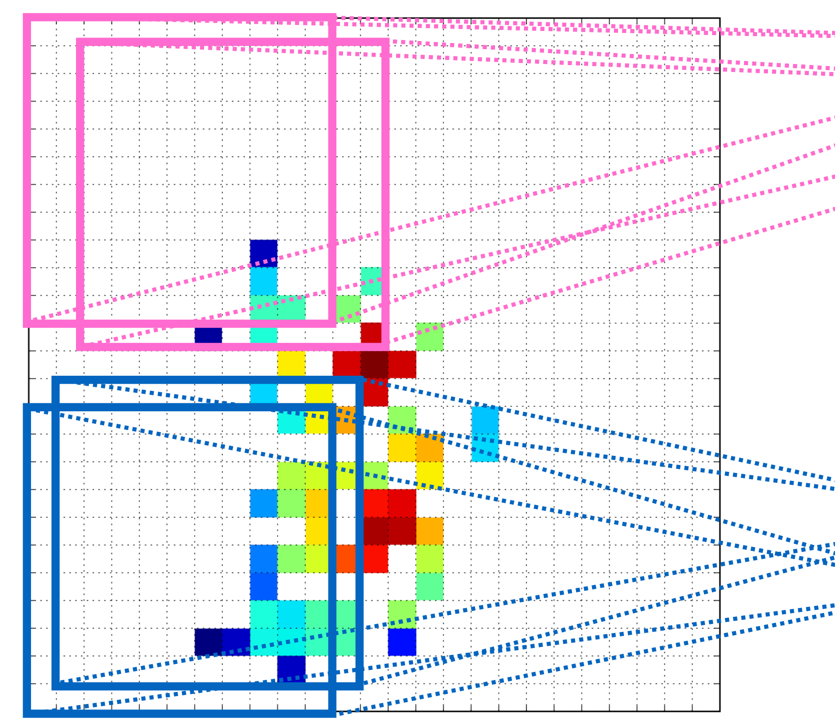
Lawrence Berkeley National Laboratory

cern.ch/bnachman

bpnachman@lbl.gov

@bpnachman

bnachman



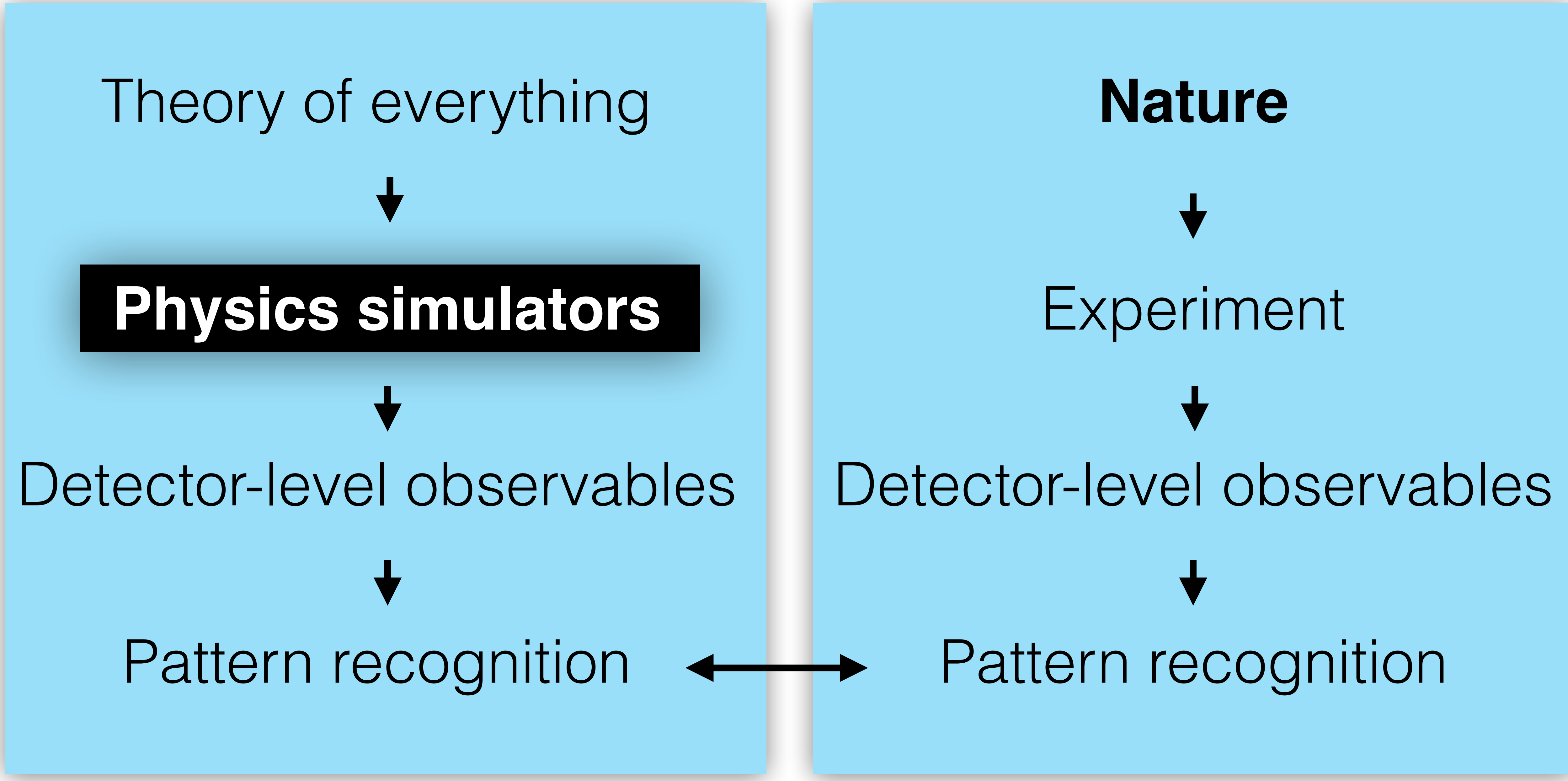
BERKELEY LAB

BERKELEY EXPERIMENTAL PARTICLE PHYSICS

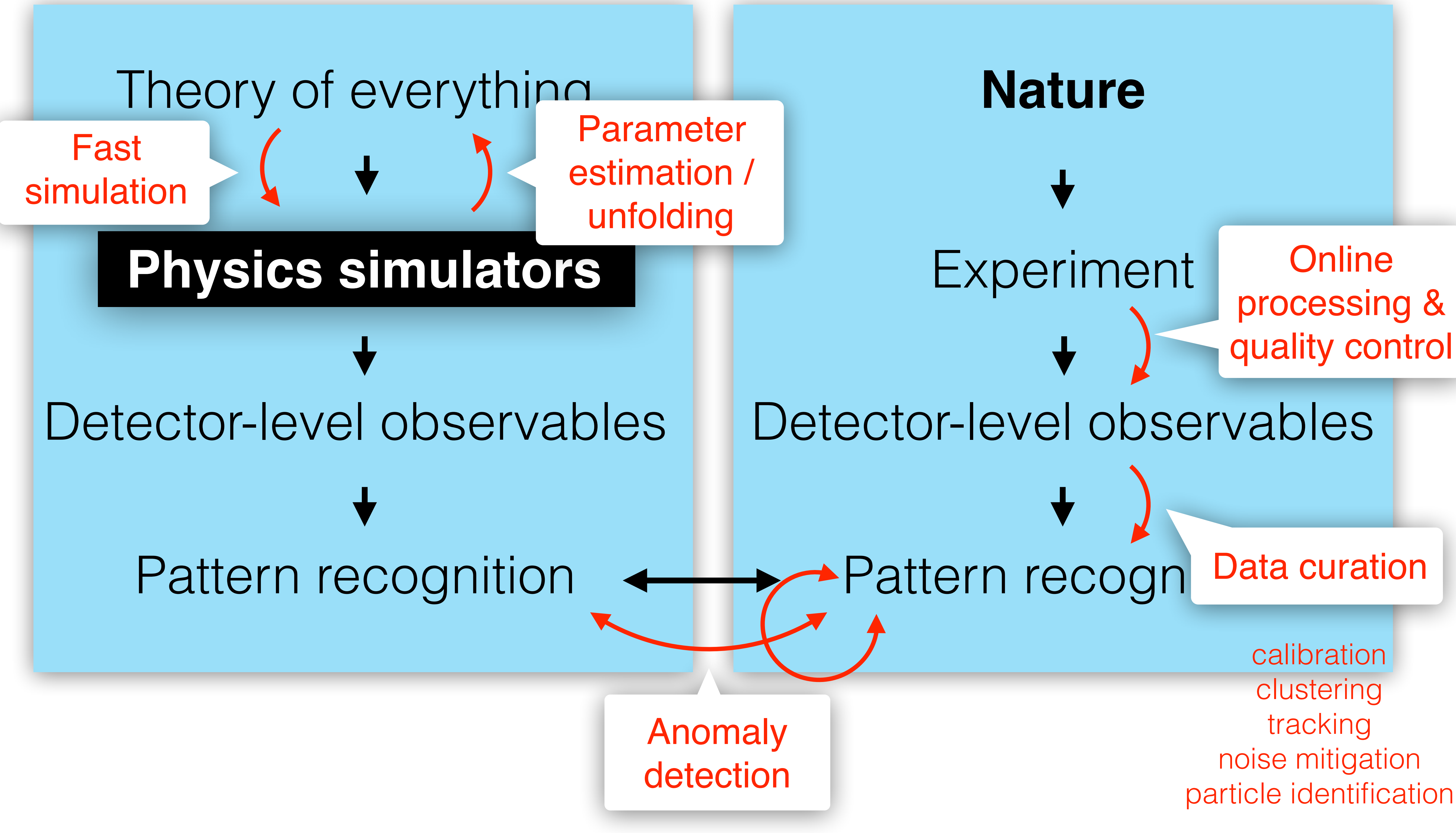
Lund MCNet ML School

June 24, 2020

Data analysis in HEP

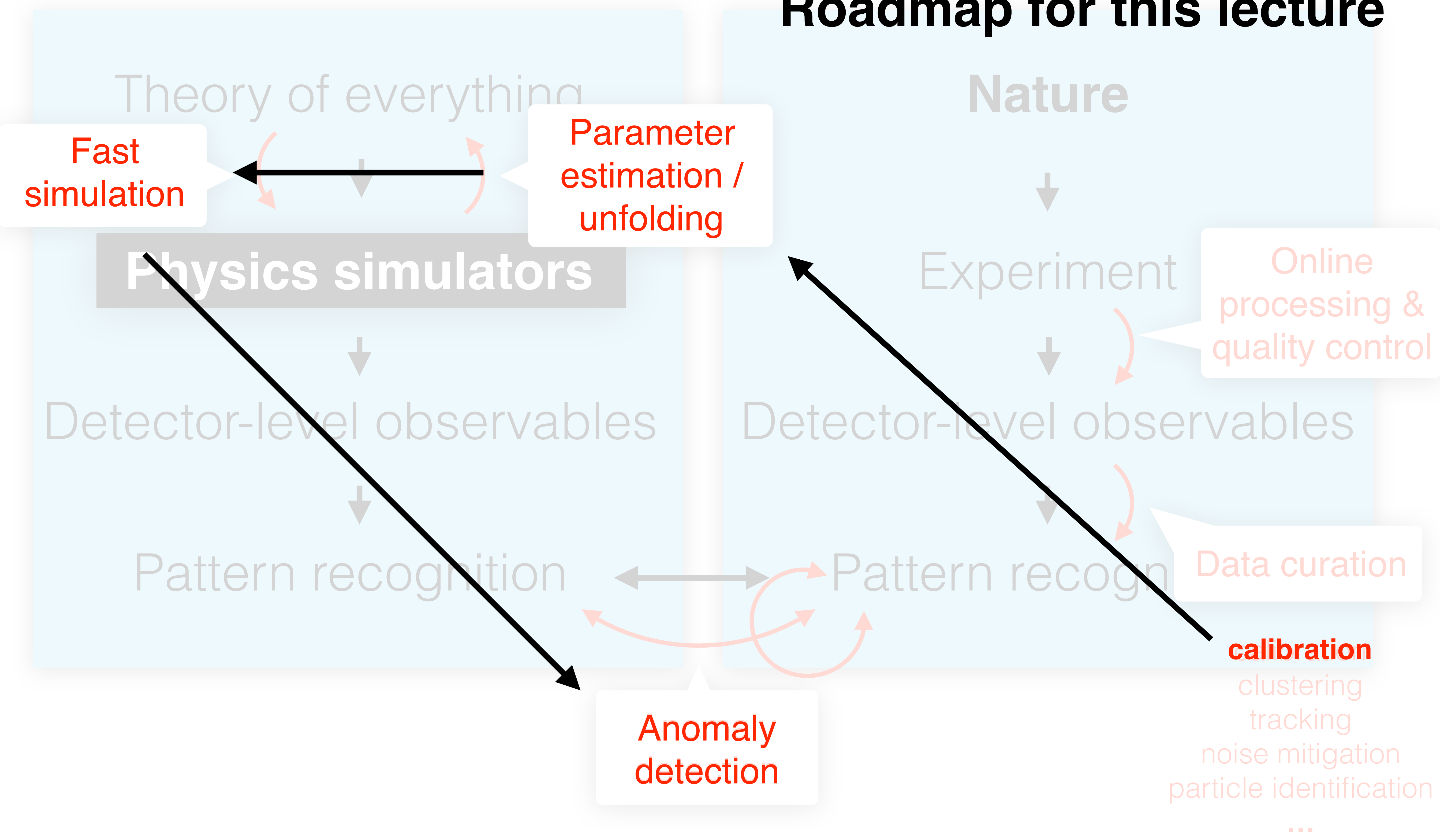


Data analysis in HEP + Machine Learning



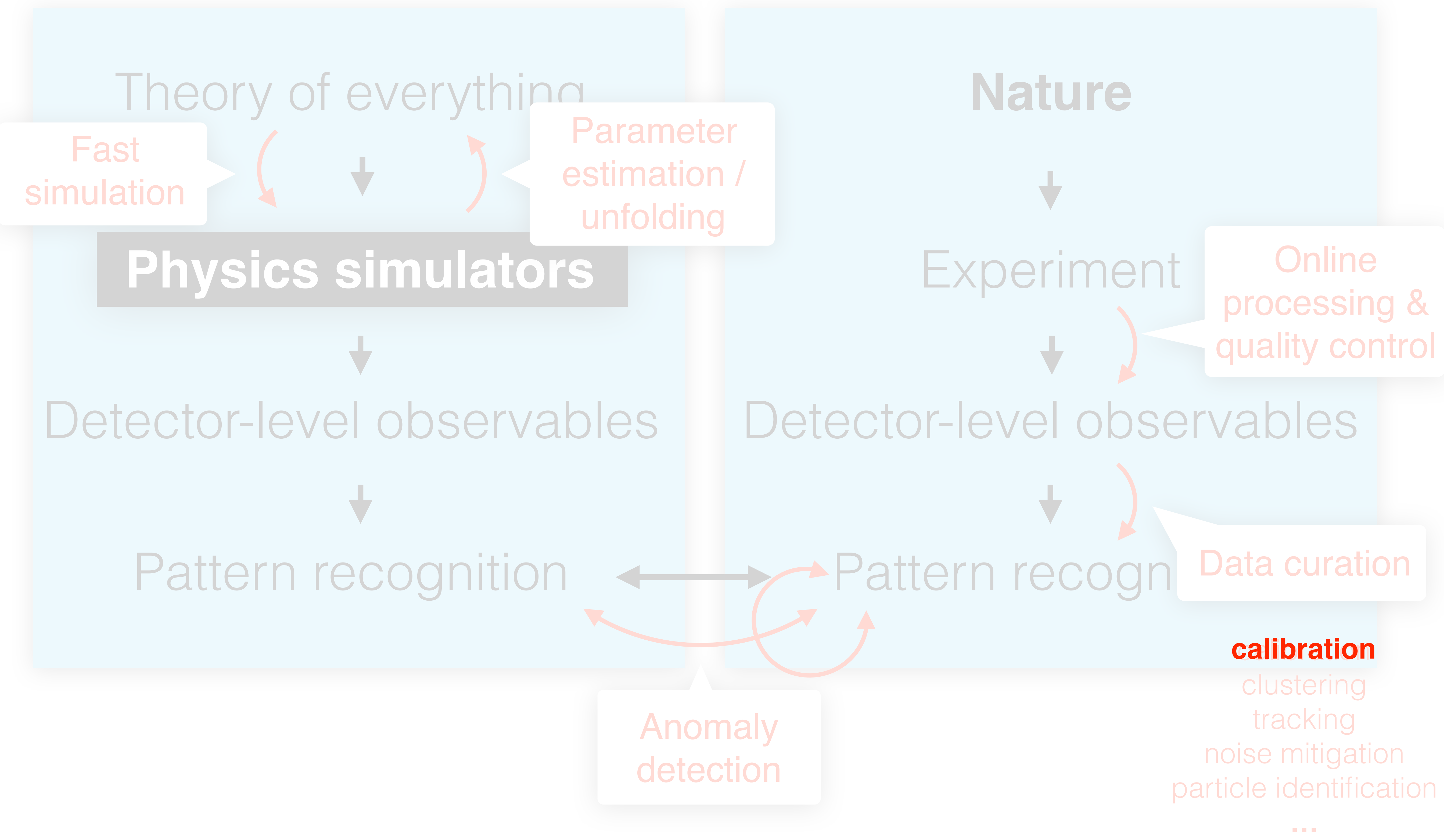
Data analysis in HEP + Machine Learning

Roadmap for this lecture



Data analysis in HEP + Machine Learning

5



Calibration - a regression task

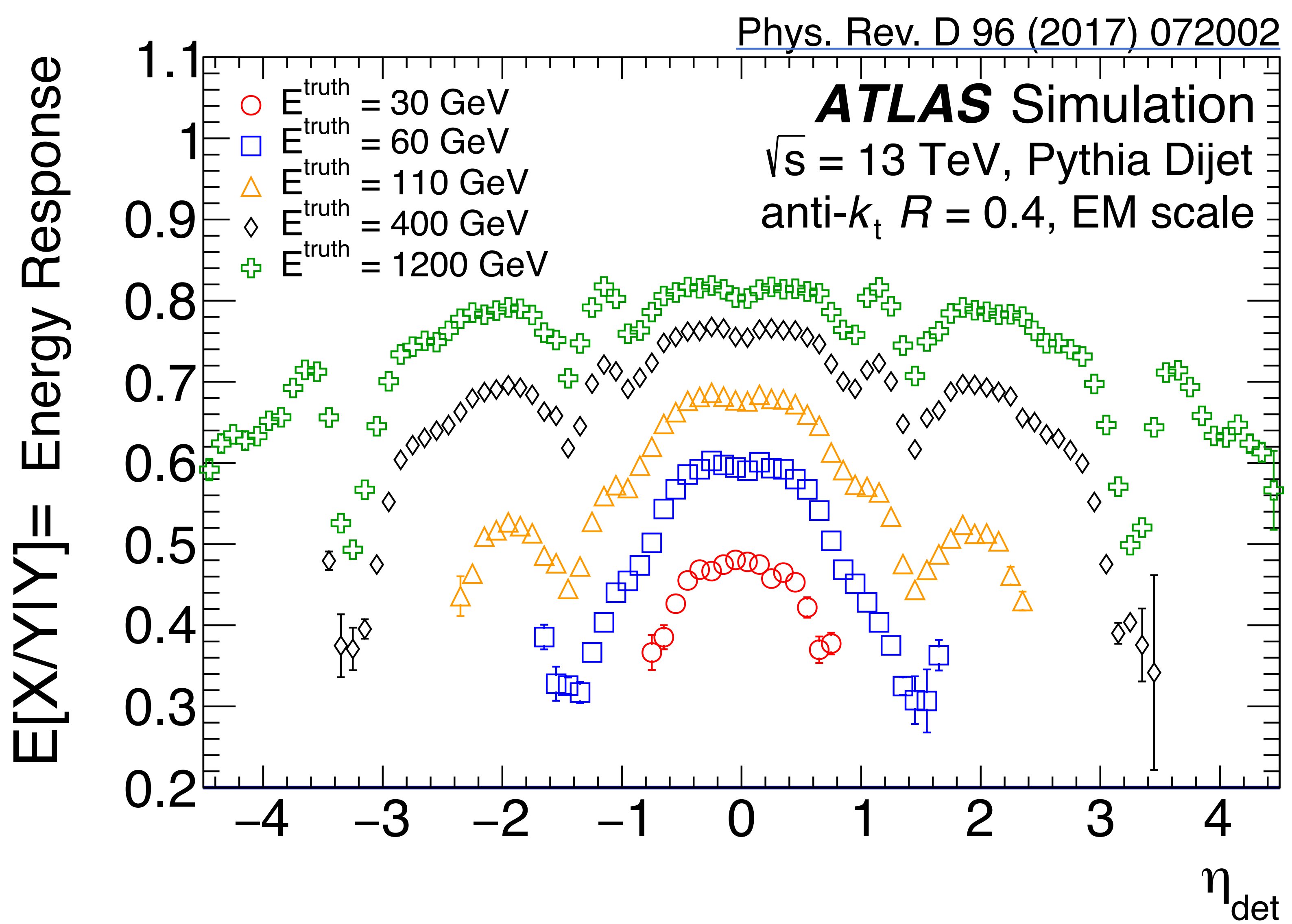
You measure X and want E[X|Y] = Y.

We'll discuss how to perform this calibration using machine learning.

One of the themes throughout this lecture will be mitigating simulation (prior) dependence.

Calibration - a regression task

An example that you can have in mind is jet energy calibration.



We want to predict the true energy given the measured energy

(and possibly other features - more on that soon)

…however what I'm about to say applied more generally (though the impact is biggest when the resolution is poorest)

What can go wrong?

Suppose you have some features x and you want to predict y.

detector energy *true energy*

One way to do this is to find an *f* that minimizes the mean squared error (MSE):

$$f = \text{argmin}_g \sum_i (g(x_i) - y_i)^2$$

Then, f(x) = E[y|x].

Check this using the method we discussed yesterday!

Could this be a problem?

What can go wrong?

$$f(x) = E[y|x] = \int \mathrm{d}y \, y \, p(y|x)$$

$$E[f(x)|y] = \int \mathrm{d}x \, \mathrm{d}y' \, y' \, p_{\mathrm{train}}(y'|x) \, p_{\mathrm{test}}(x|y)$$

this need not be y even if $p_{train} = p_{test}$ (!)

One solution: Numerical inversion

ATLAS and CMS use a trick to be prior-independent:

Numerical inversion *instead of predicting y from x, predict x from y and then invert the function*

… put another way:

learn $f: y \rightarrow x$ and then for a given x, predict $f^{-1}(x)$

by construction, f is independent of p(y) and thus f^{-1} also does not depend on p(y), as desired.

For math details, see A. Cukierman and B. Nachman, NIMA 858 (2017) 1

Caveats about numerical inversion

This procedure is independent of the prior p(y) but may not close exactly, i.e. $E[f^{-1}(x)|y]$ may not be y.

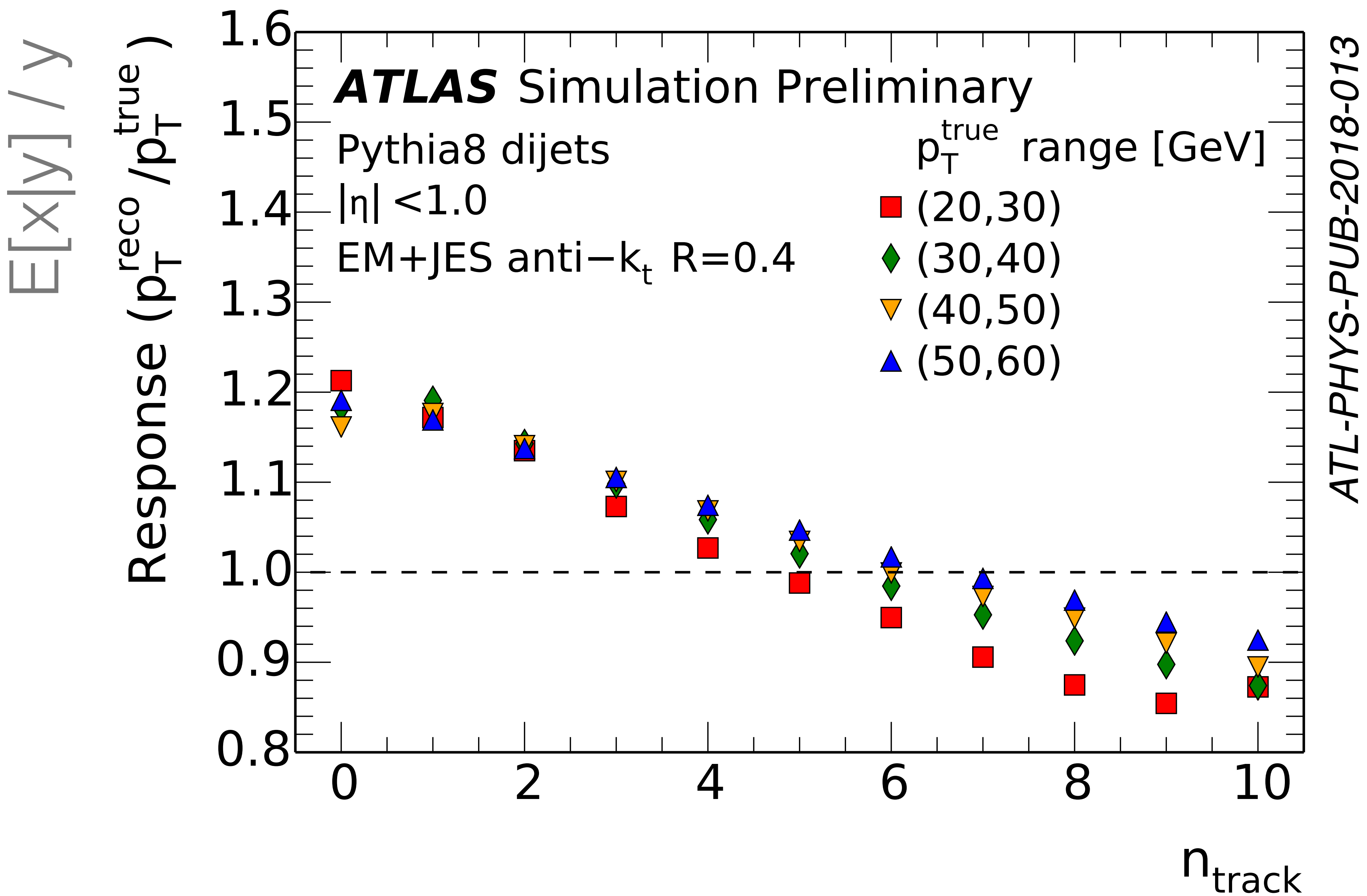
…under mild assumptions, it does close for the mean absolute error, but usually has some non-closure for the MSE.

Also, the calibration procedure can distort the underlying distribution, i.e. if you start with a Gaussian, you almost never end up with exactly a Gaussian.

For math details, see A. Cukierman and B. Nachman, NIMA 858 (2017) 1

+ more features

The detector response of jets depends on many properties of the jet. Ideally, the calibration can include this!

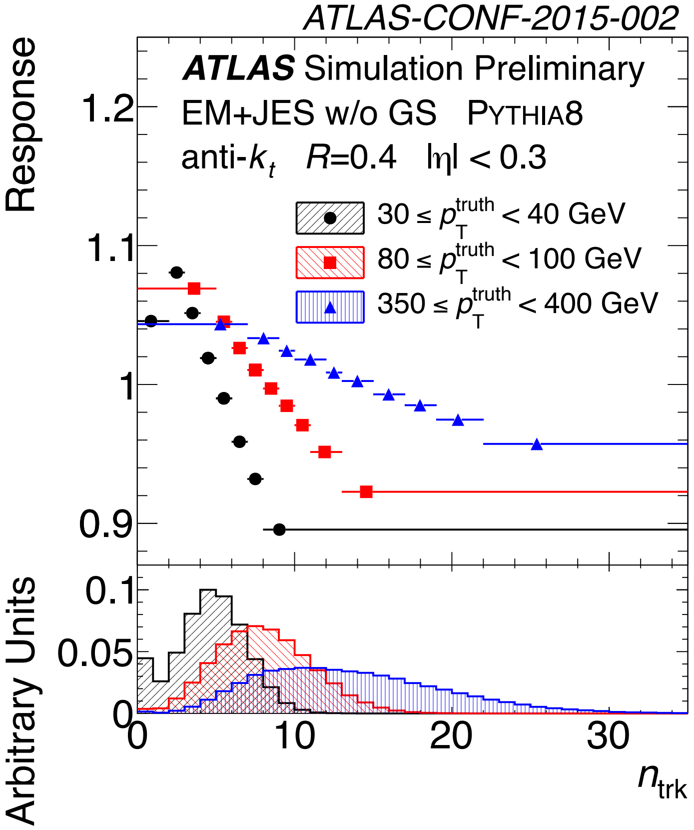


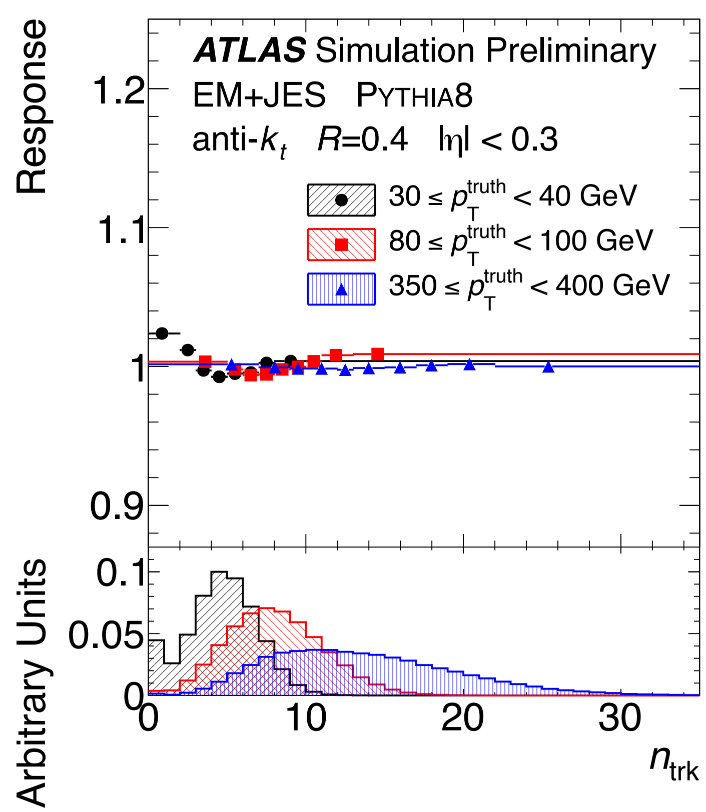
Global sequential calibration

The current ATLAS approach to including more features is to repeat NI sequentially:

$$p_{\mathrm{T}}^{\mathrm{reco}} \mapsto \hat{p}_{\mathrm{T}}^{\mathrm{reco}} = f_{\theta_n}^{-1}\left(\cdots f_{\theta_2}^{-1}\left(f_{\theta_1}^{-1}\left(p_{\mathrm{T}}^{\mathrm{reco}}\right)\right)\cdots\right)$$

ATLAS-CONF-2015-002





This works well when the jet response is independent of θ_i given θ_j.

Machine learning calibration

For reasons discussed earlier, we can't include correlations by learning y given x and all the θ's.

However, it would still be great to use machine learning to automatically and efficiently make use of correlated information.

We cannot use numerical inversion out-of-the-box because we now have a many-to-one function.

Generalized numerical inversion

Since we are not (necessarily) interested in calibrating the θ's, we can generalize NI as follows:

(1) Learn a function f to predict x given y and all the θ's.

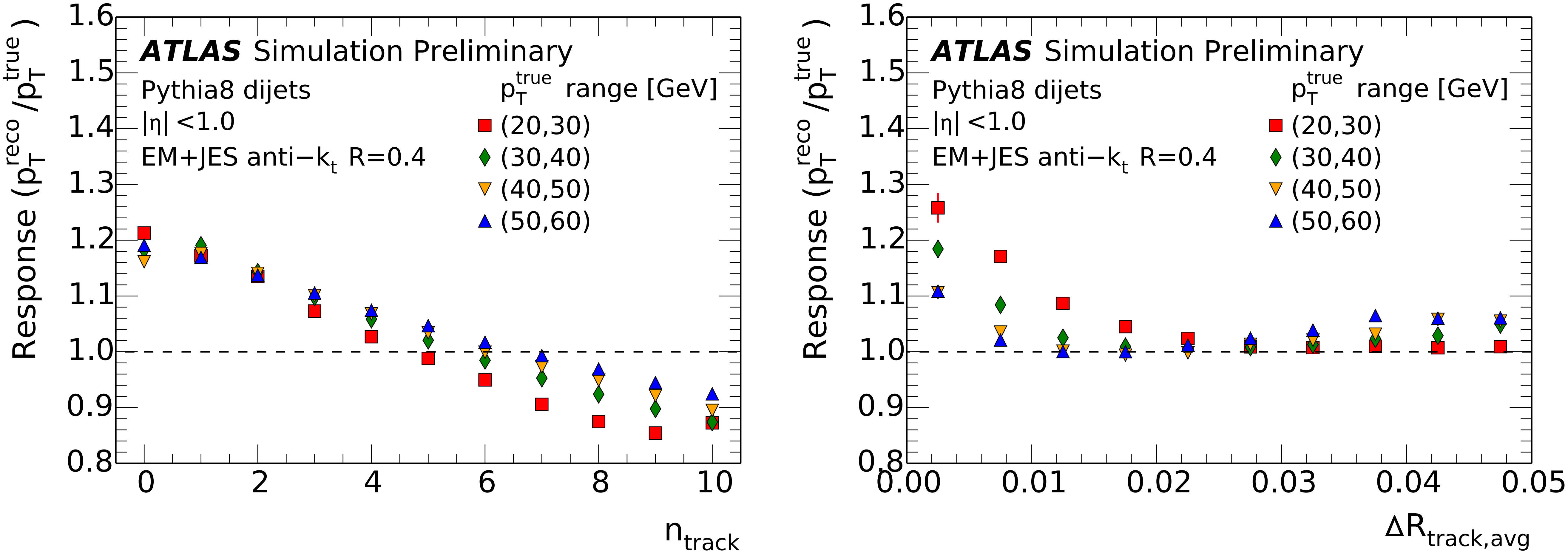
(2) For every combination of θ, invert f.

(3) Calibrate via $x \rightarrow f_\theta^{-1}(x)$

Step (2) is intractable, so replace it with another learning step: predict y given f(y,θ) and θ.

GNI in action

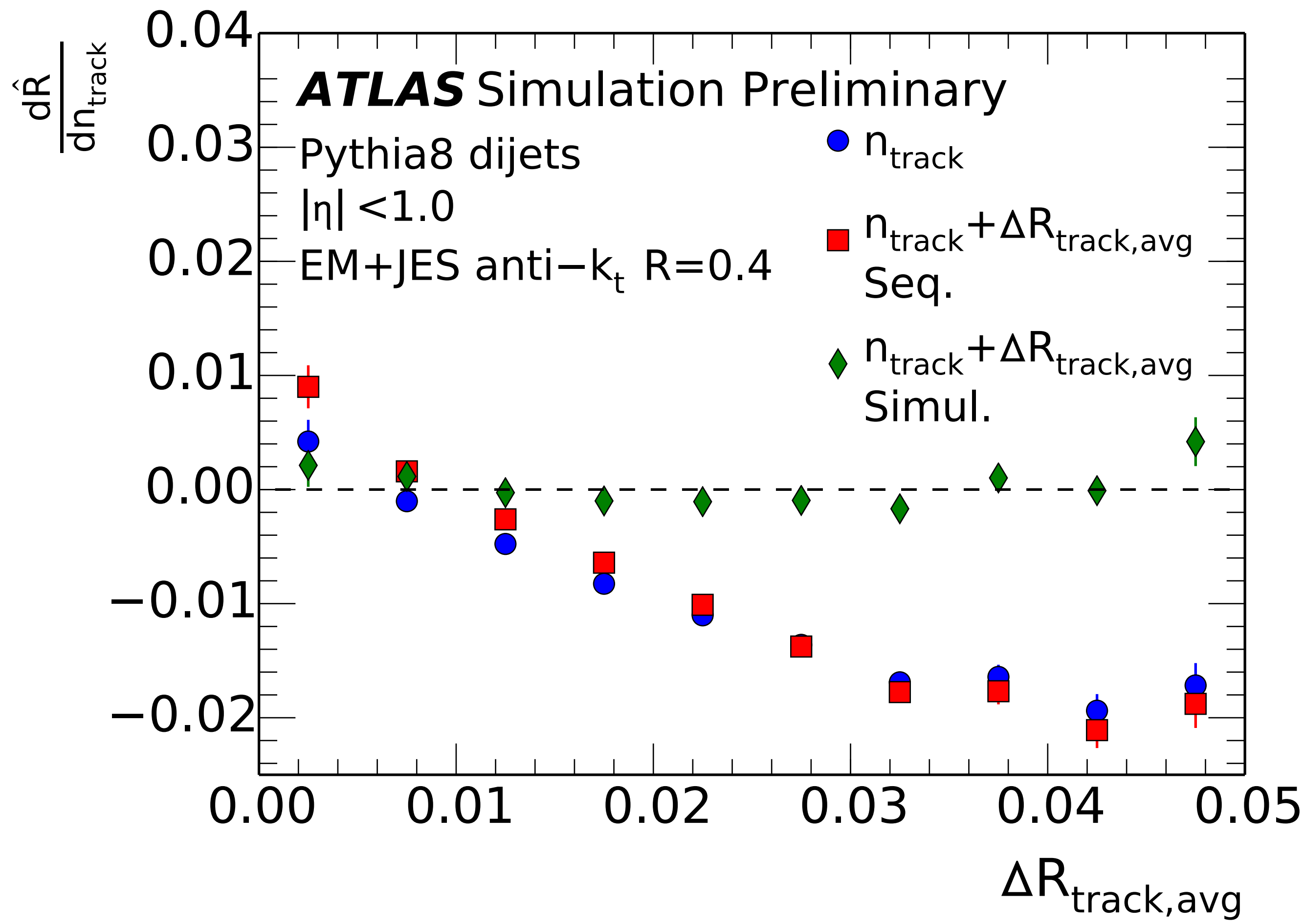
Consider two features:

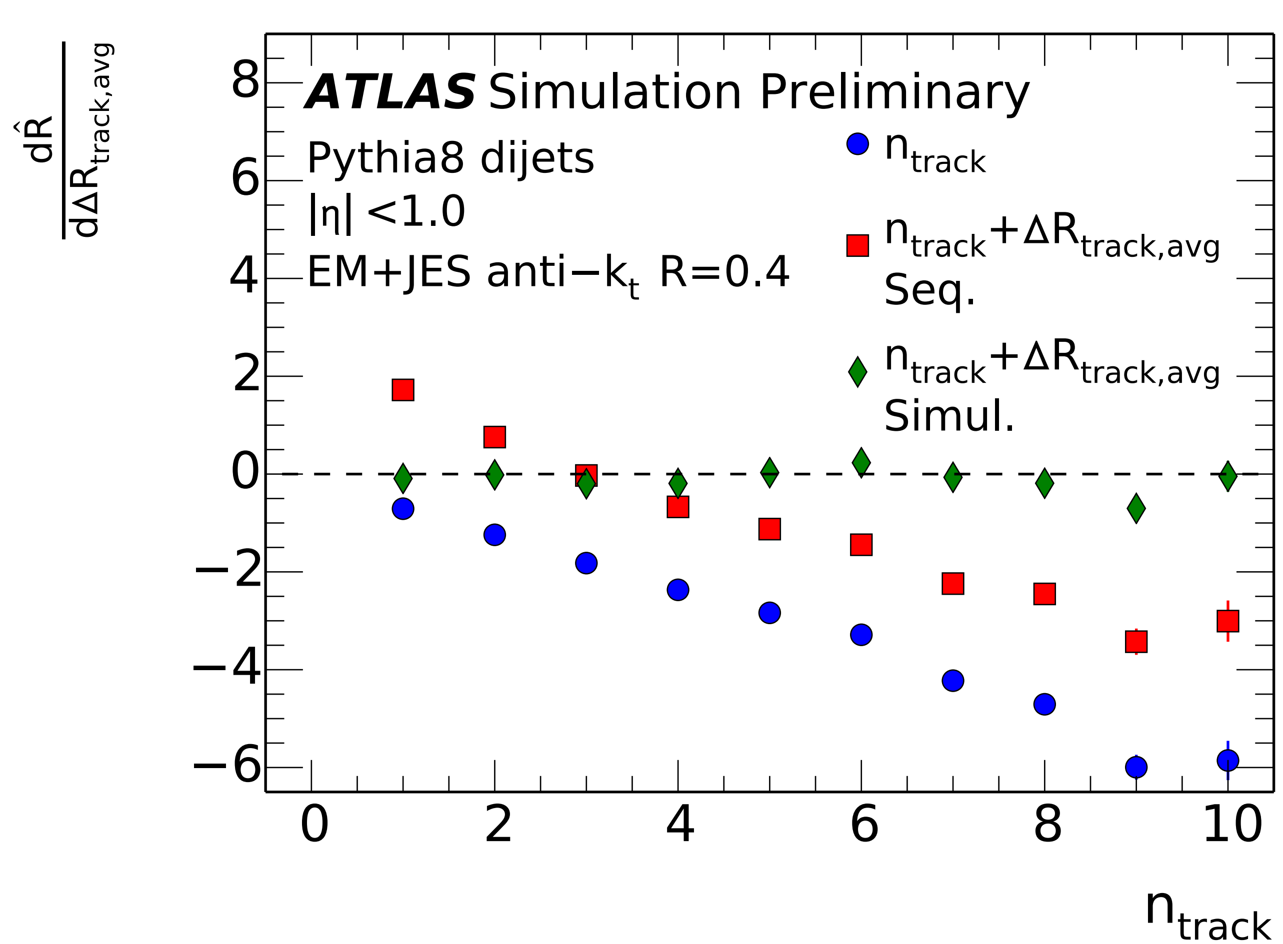


average track p_T-weighted distance from jet center

GNI in action

$\hat{R}$ is the calibrated E[x|y] / y





Only the **simultaneous approach** removes the full residual dependence!

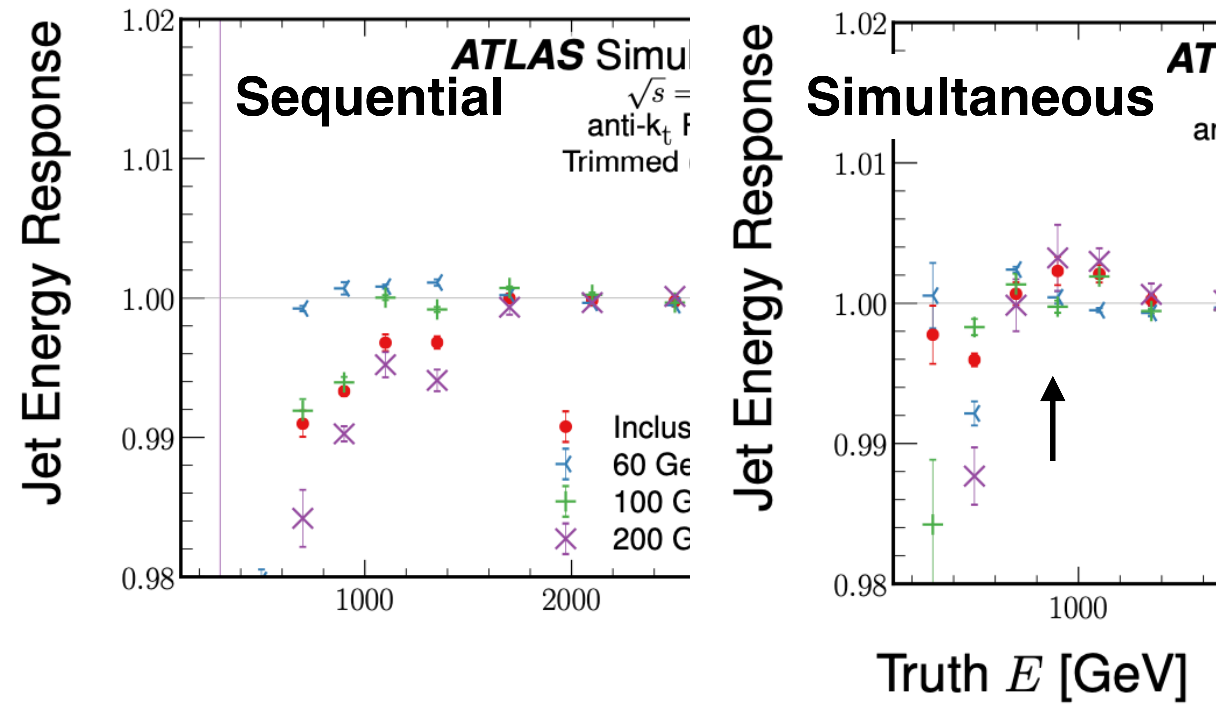
Further generalizations

Can also simultaneously calibrate a subset of the θ's (e.g. jet energy and mass)

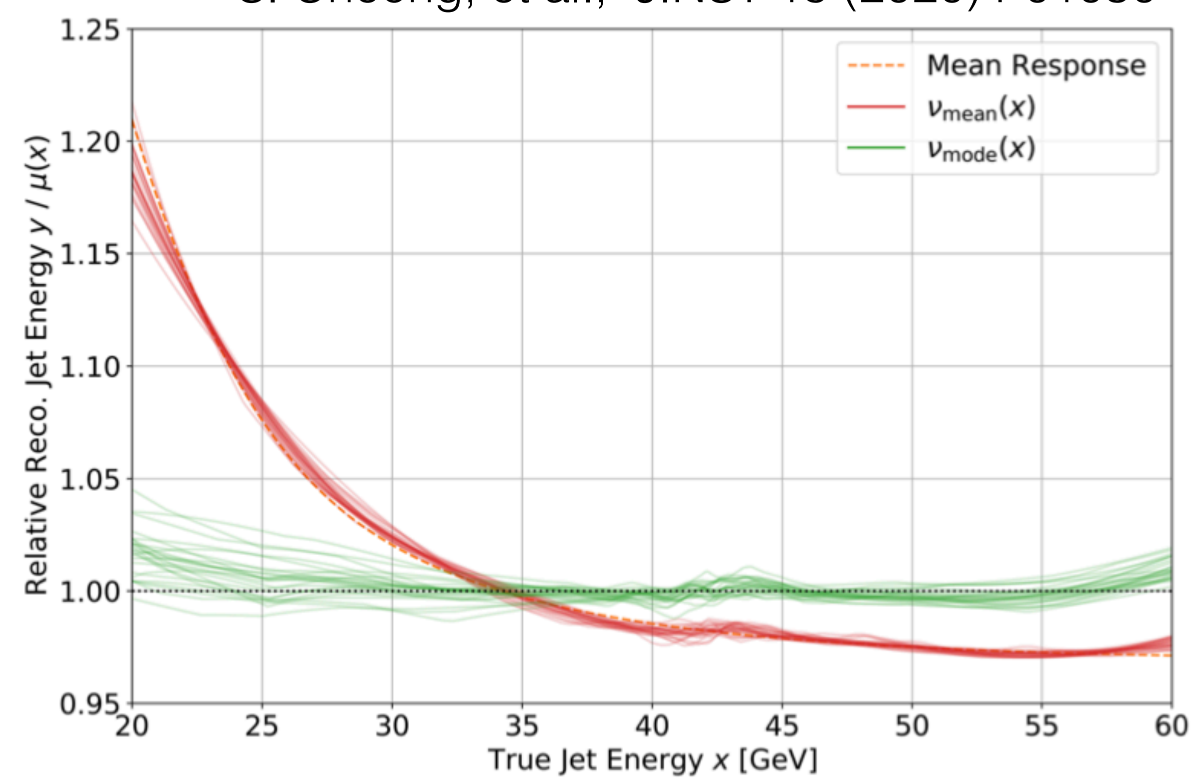
In many cases, it is desirable to calibrate the mode and not the mean since p(X|Y) is asymmetric.

Can achieve this with modified loss function!

ATL-PHYS-PUB-2020-001



S. Cheong, et al., JINST 15 (2020) P01030

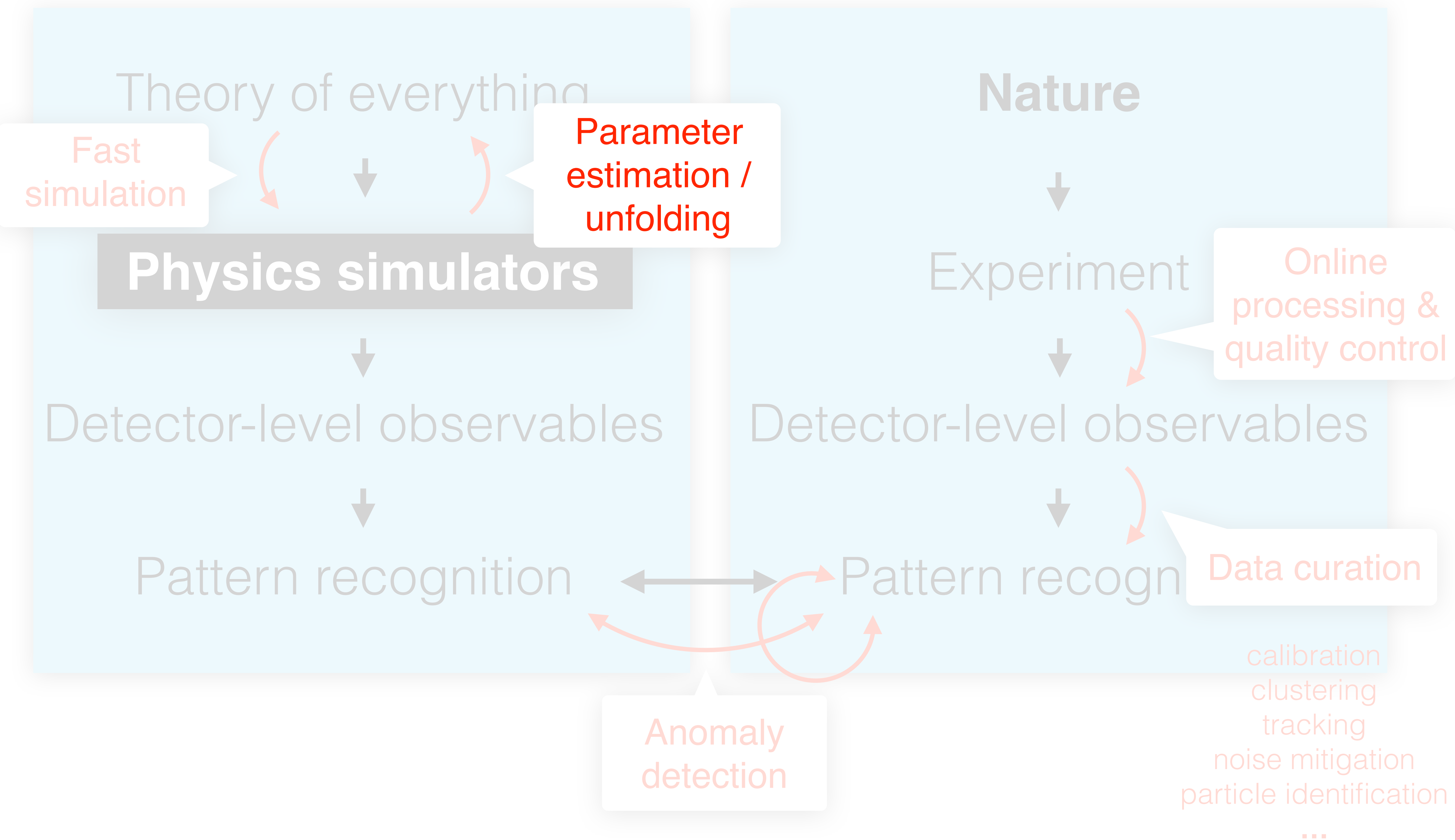


Partial Conclusions for Regression

There are many more applications of regression in HEP, but calibration is a prototypical example.

When building a regression model, it is critical to be wary of prior dependence and to pick the loss function based on what you actually want to learn (mean/median/mode/IQR/etc.)

Data analysis in HEP + Machine Learning



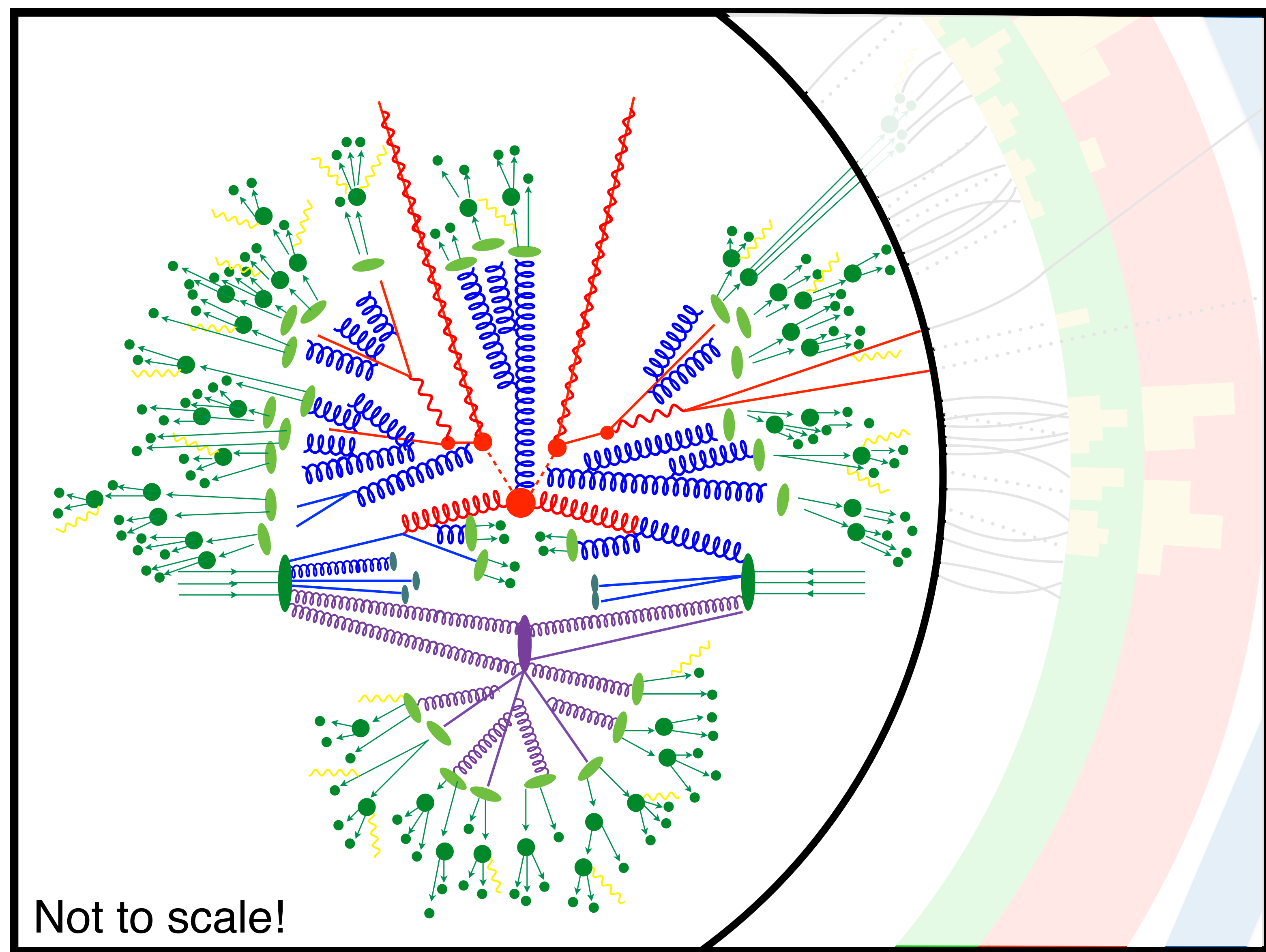
A *hyper* challenge for inference

Key **challenge** and **opportunity**: *hypervariate phase space*
& *hyper spectral data*

Typical collision events at the LHC produce **O(1000+)** particles

We detect these particles with **O(100 M)** readout channels

Image inspired by JHEP 02 (2009) 007



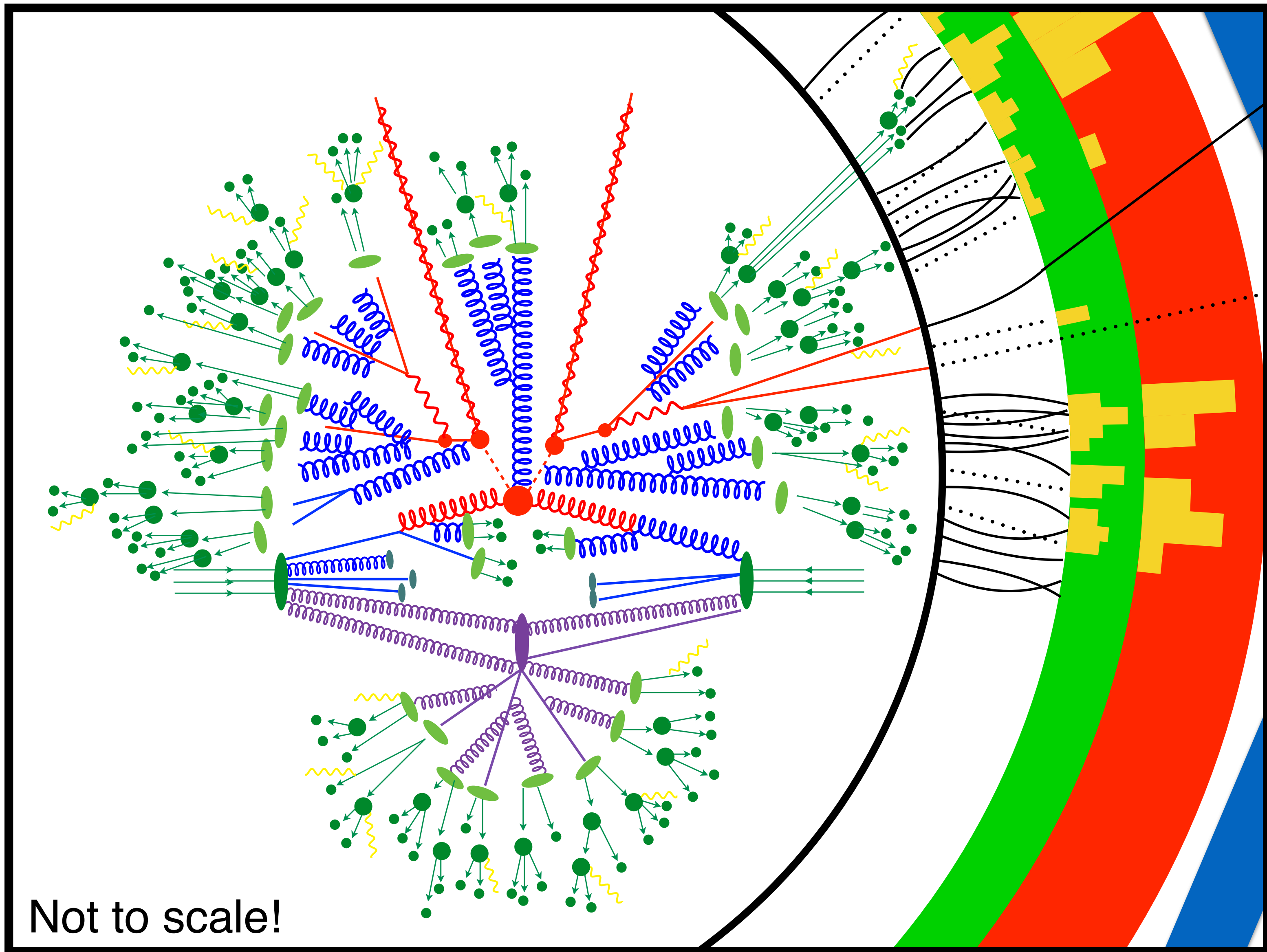
A *hyper* challenge for inference

Key **challenge** and **opportunity**: *hypervariate phase space* & *hyper spectral data*

Typical collision events at the LHC produce **O(1000+)** particles

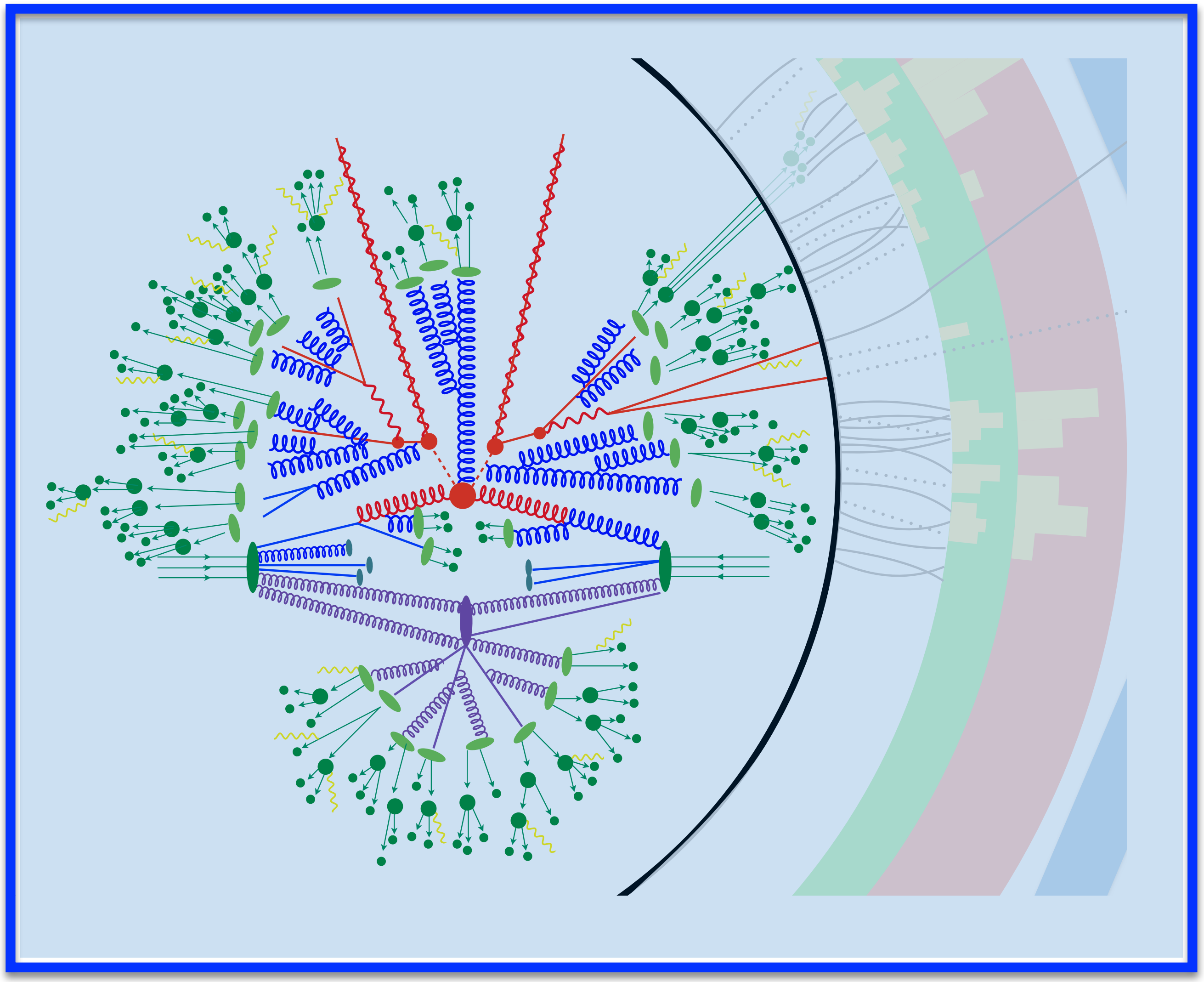
We detect these particles with **O(100 M)** readout channels

Image inspired by JHEP 02 (2009) 007

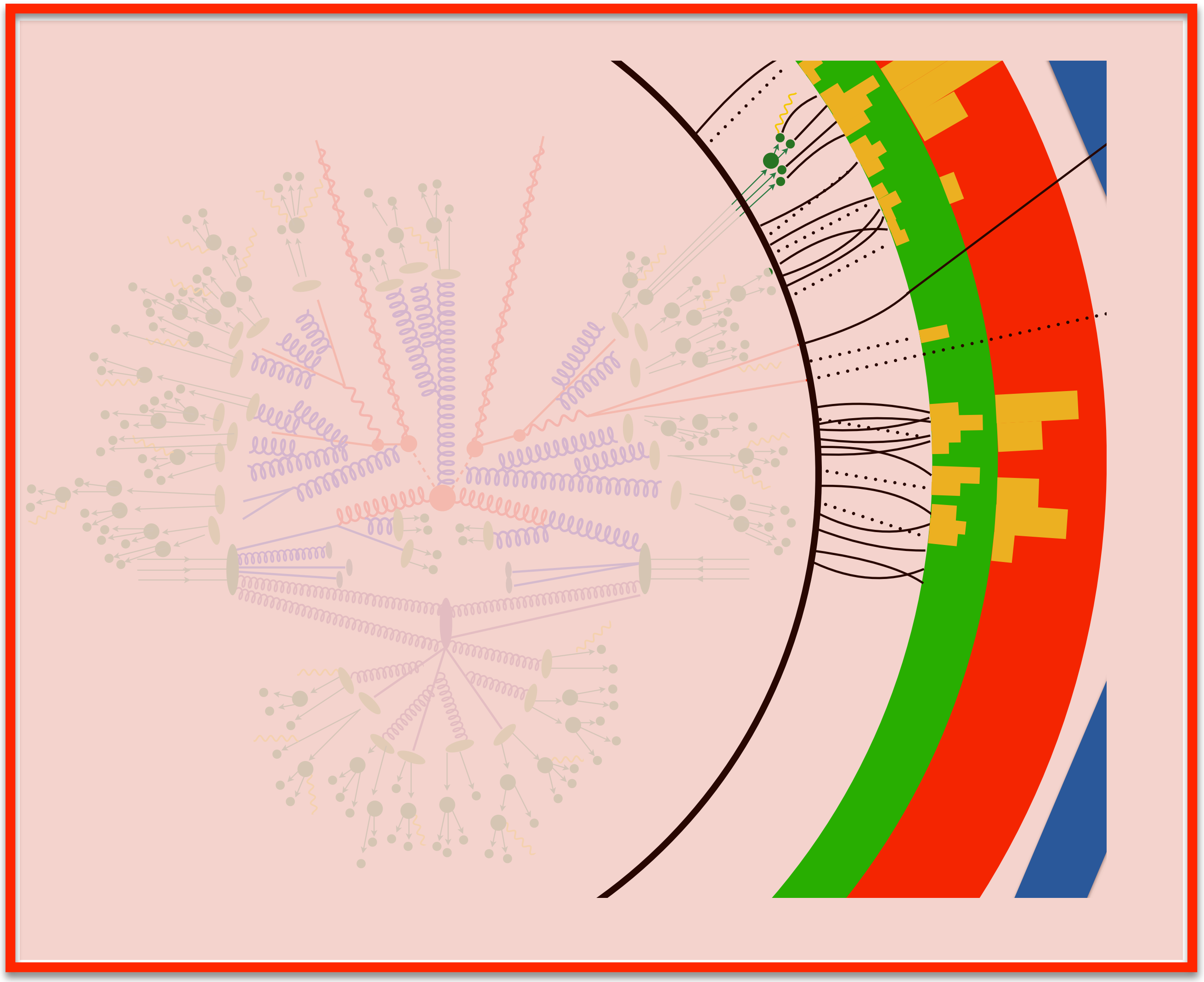


Example: Unfolding (Deconvolution)

Want this ↓



Measure this ↓

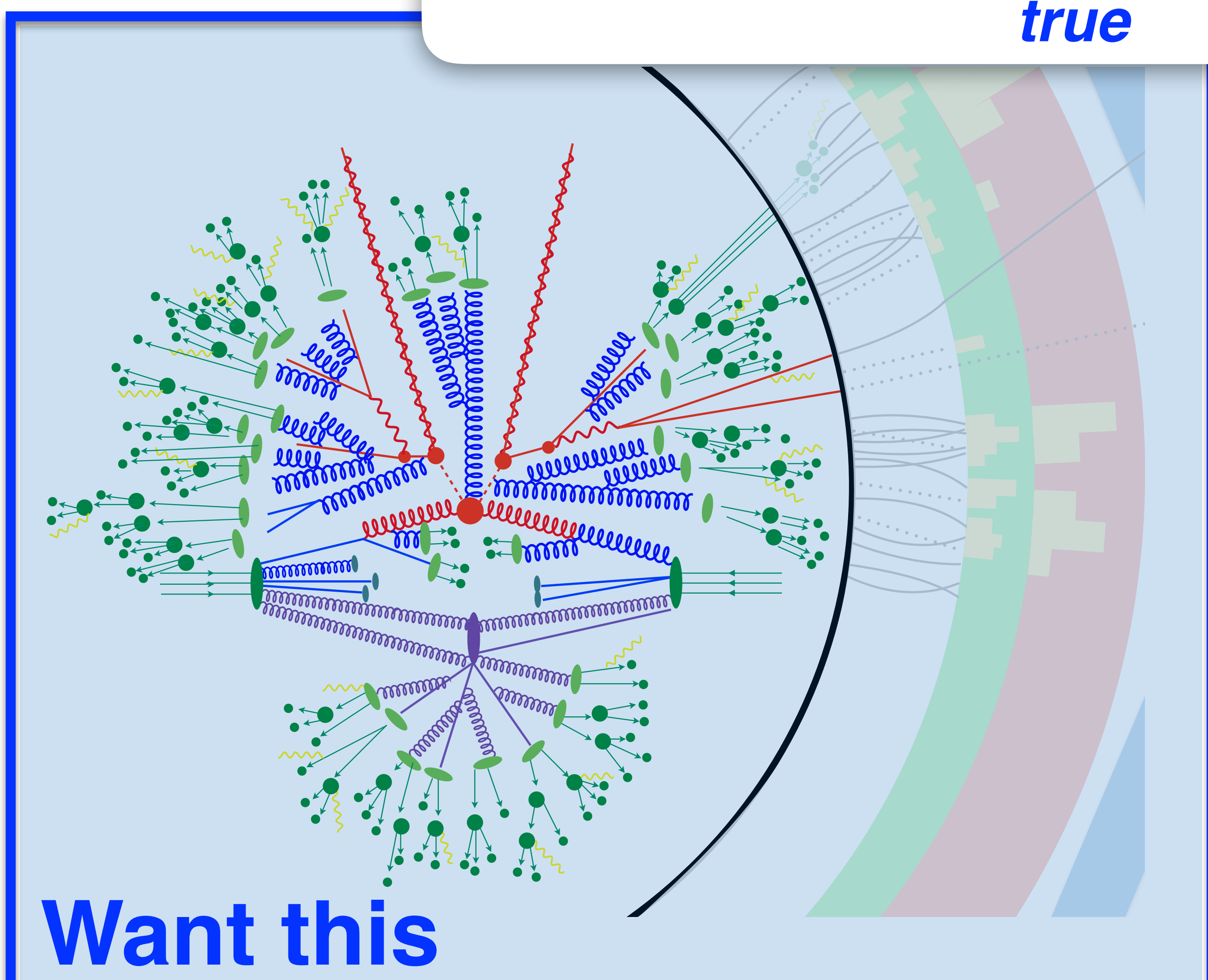


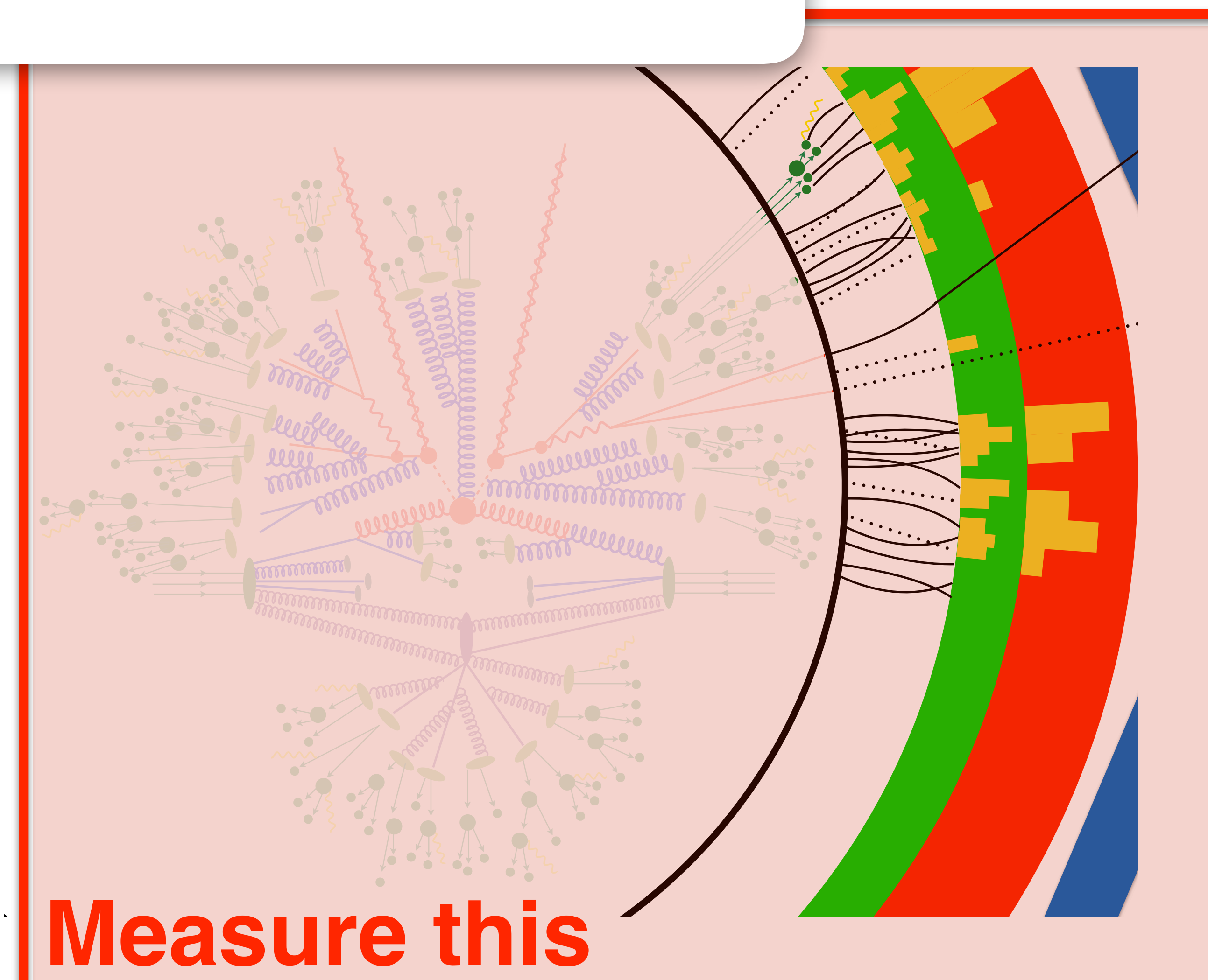
i.e. remove detector distortions

Example: Unfolding (Deconvolution)

If you know *p(**meas.** / **true**)*, could do maximum likelihood, i.e.

$$\text{unfolded} = \underset{\text{true}}{\text{argmax}}\ p(\text{measured} \,/\, \text{true})$$





*p(**meas.** / **true**)* = “response matrix” or “point spread function”

Example: Unfolding (Deconvolution)

If you know *p(**meas.** | **true**)*, could do maximum likelihood, i.e.

$$\text{unfolded} = \underset{\text{true}}{\operatorname{argmax}}\; p(\text{measured} \mid \text{true})$$

Challenge: **measured** is hyperspectral and **true** is hypervariate … *p(**meas.** | **true**) is **intractable** !*

p(meas. | true) = “response matrix” or “point spread function”

Example: Unfolding (Deconvolution)

If you know *p(**meas.** / **true**)*, could do maximum likelihood, i.e.

$$\text{unfolded} = \underset{\text{true}}{\operatorname{argmax}}\ p(\text{measured} \,/\, \text{true})$$

Challenge: **measured** is hyperspectral and **true** is hypervariate … *p(**meas.** | **true**) is **intractable** !*

However: we have **simulators** that we can use to sample from *p(**meas.** | **true**)*

→ **Simulation-based (likelihood-free) inference**

*p(**meas.** / **true**)* = "response matrix" or "point spread function"

I'll briefly show you one solution to give you a sense of the power of likelihood-free inference.

Reweighting

I'll briefly show you one solution to give you a sense of the power of likelihood-free inference.

The solution will be built on ***reweighting***

dataset 1: sampled from ***p(x)***
dataset 2: sampled from ***q(x)***

Create weights ***w(x) = q(x)/p(x)*** so that when dataset 1 is weighted by ***w***, it is statistically identical to dataset 2.

Reweighting

I'll briefly show you one solution to give you a sense of the power of likelihood-free inference.

The solution will be built on ***reweighting***

dataset 1: sampled from $p(x)$
dataset 2: sampled from $q(x)$

Create weights $w(x) = q(x)/p(x)$ so that when dataset 1 is weighted by w, it is statistically identical to dataset 2.

What if we don't (and can't easily) know q and p?

Classification for reweighting

Fact: Neutral networks learn to approximate the likelihood ratio $= q(x)/p(x)$

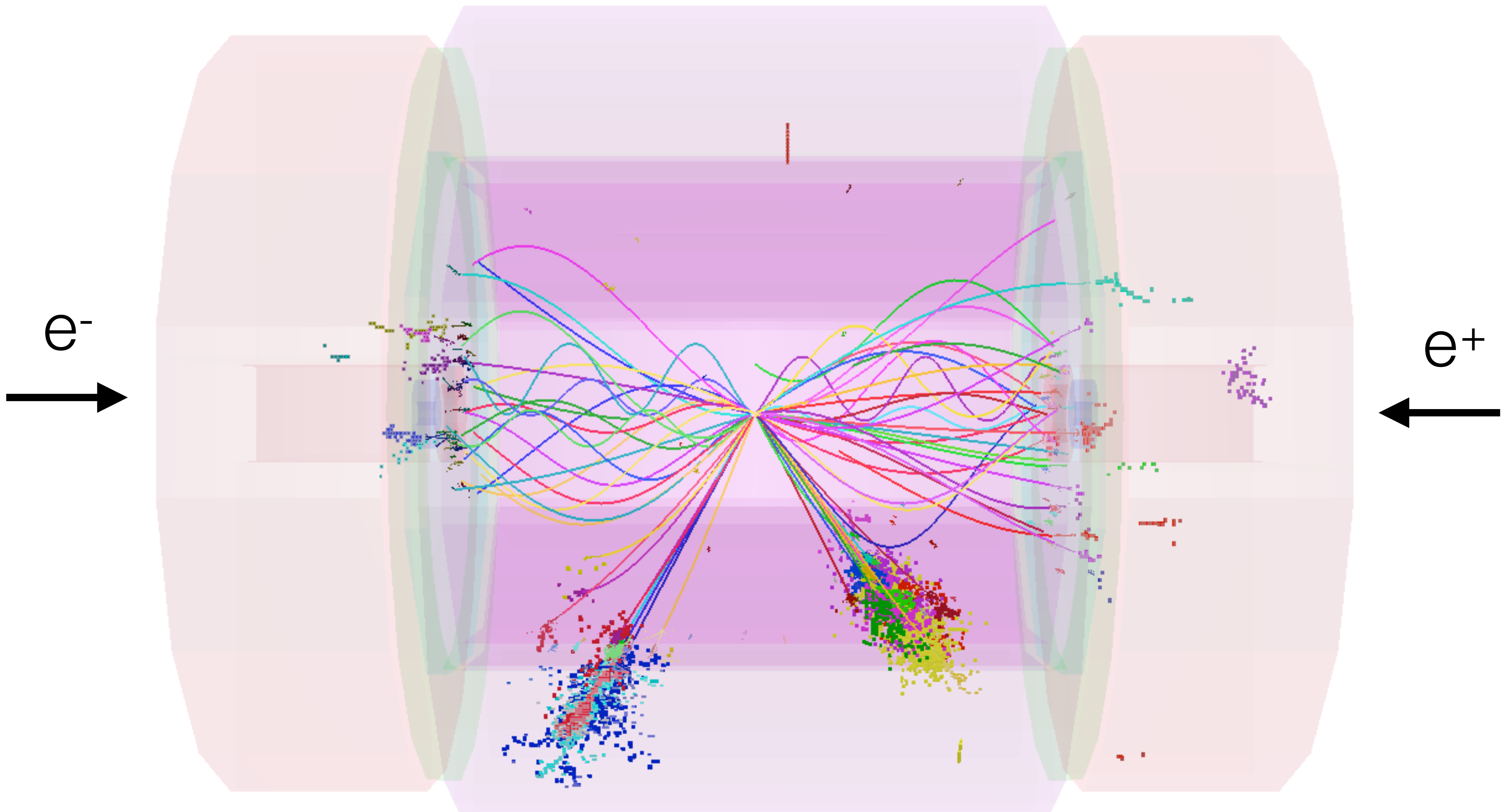
(see previous lecture! Can you derive the monotonic relation?)

Solution: train a neural network to distinguish the two datasets!

This turns the problem of **density estimation** (**hard**) into a problem of **classification** (**easy**)

Classification for reweighting

Particularly useful for particle physics, where collisions may produce a variable # of particles which are interchangeable*

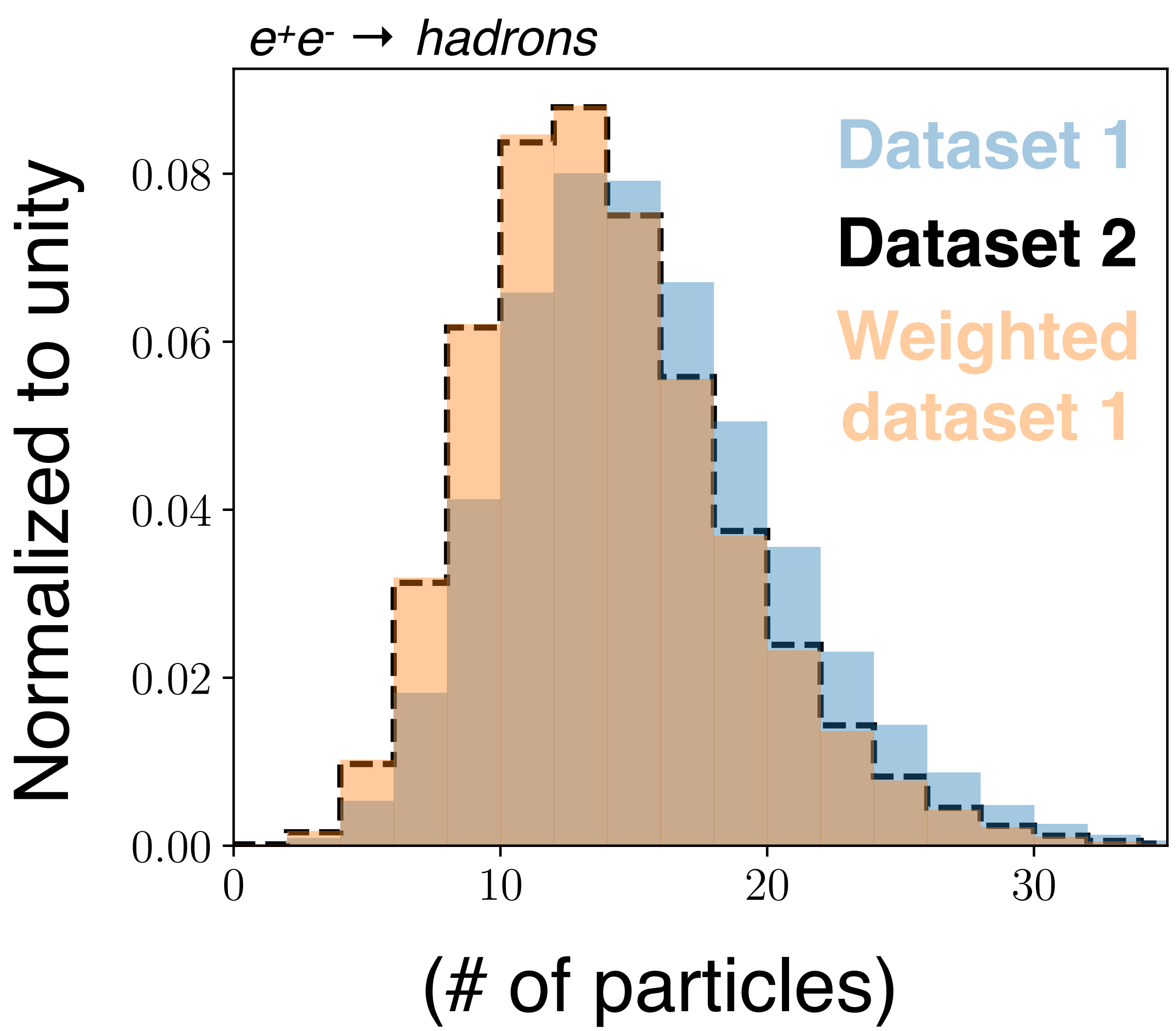


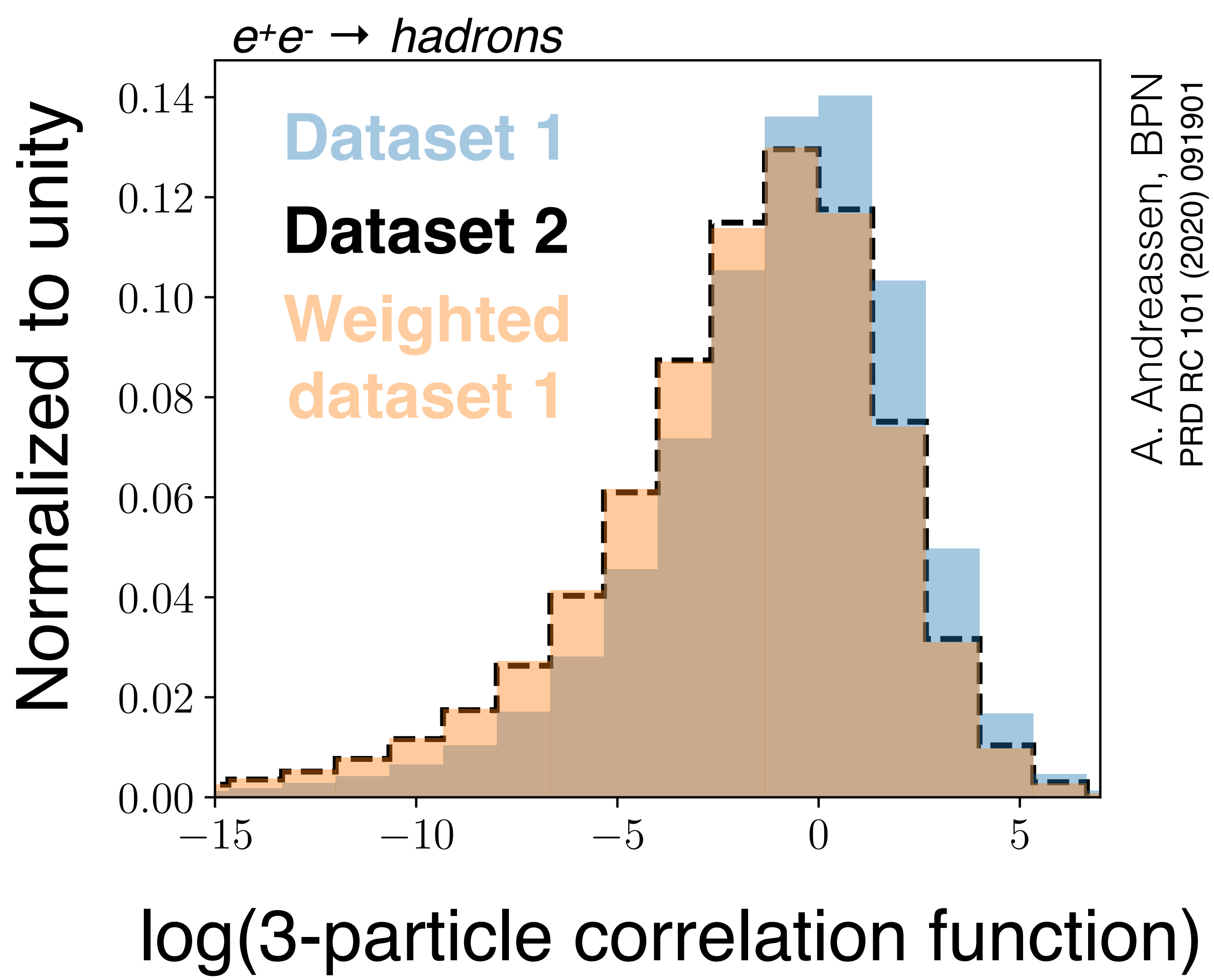
*deep learning architecture: deep sets, Zaheer et al., NIPS 2017, Komiske, Metodiev, Thaler, JHEP 01 (2019) 121

Image: Linear Collider Detector Project

Classification for reweighting

Reweight the **full phase space** and then check for various binned 1D observables.

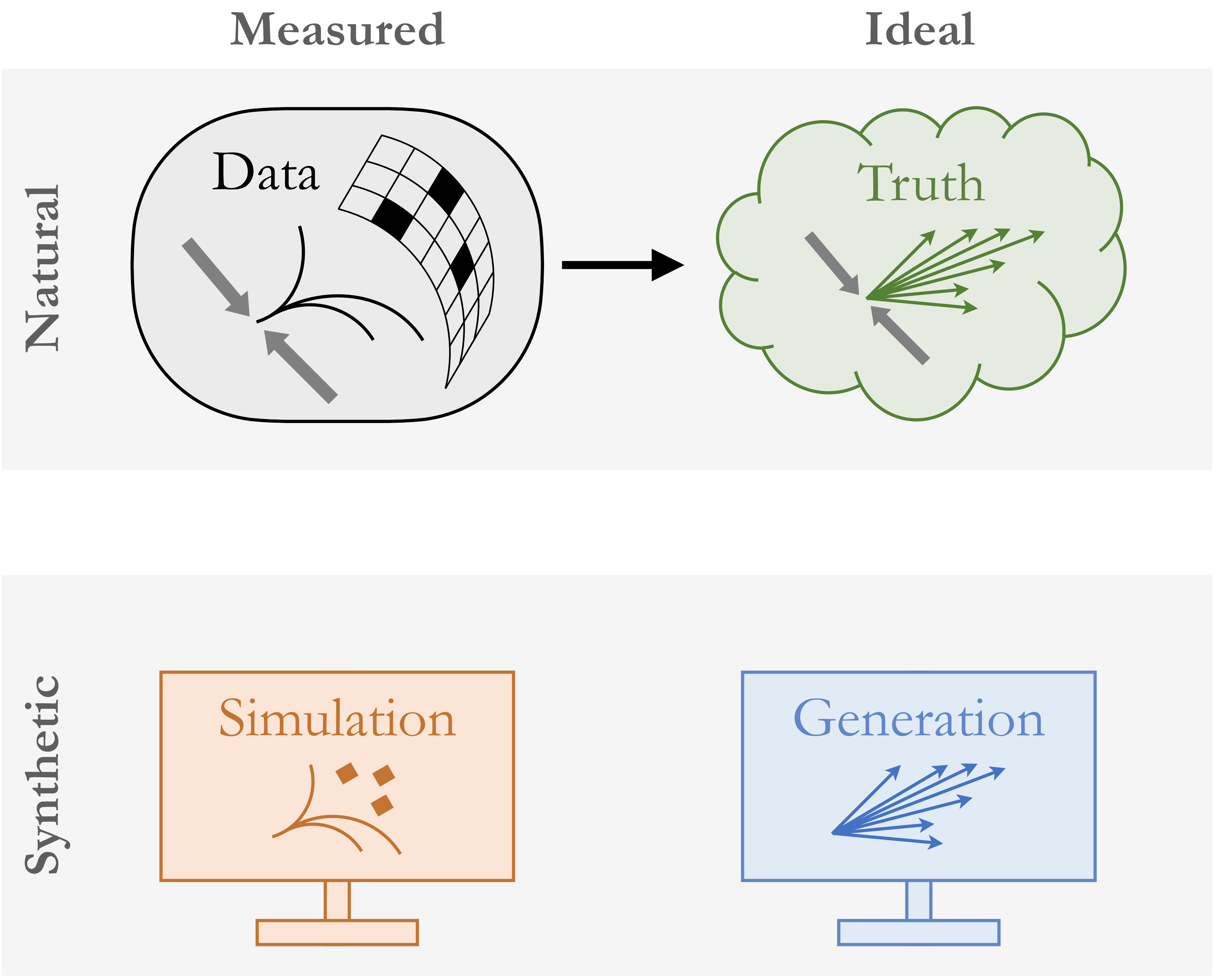




A. Andreassen, BPN
PRD RC 101 (2020) 091901

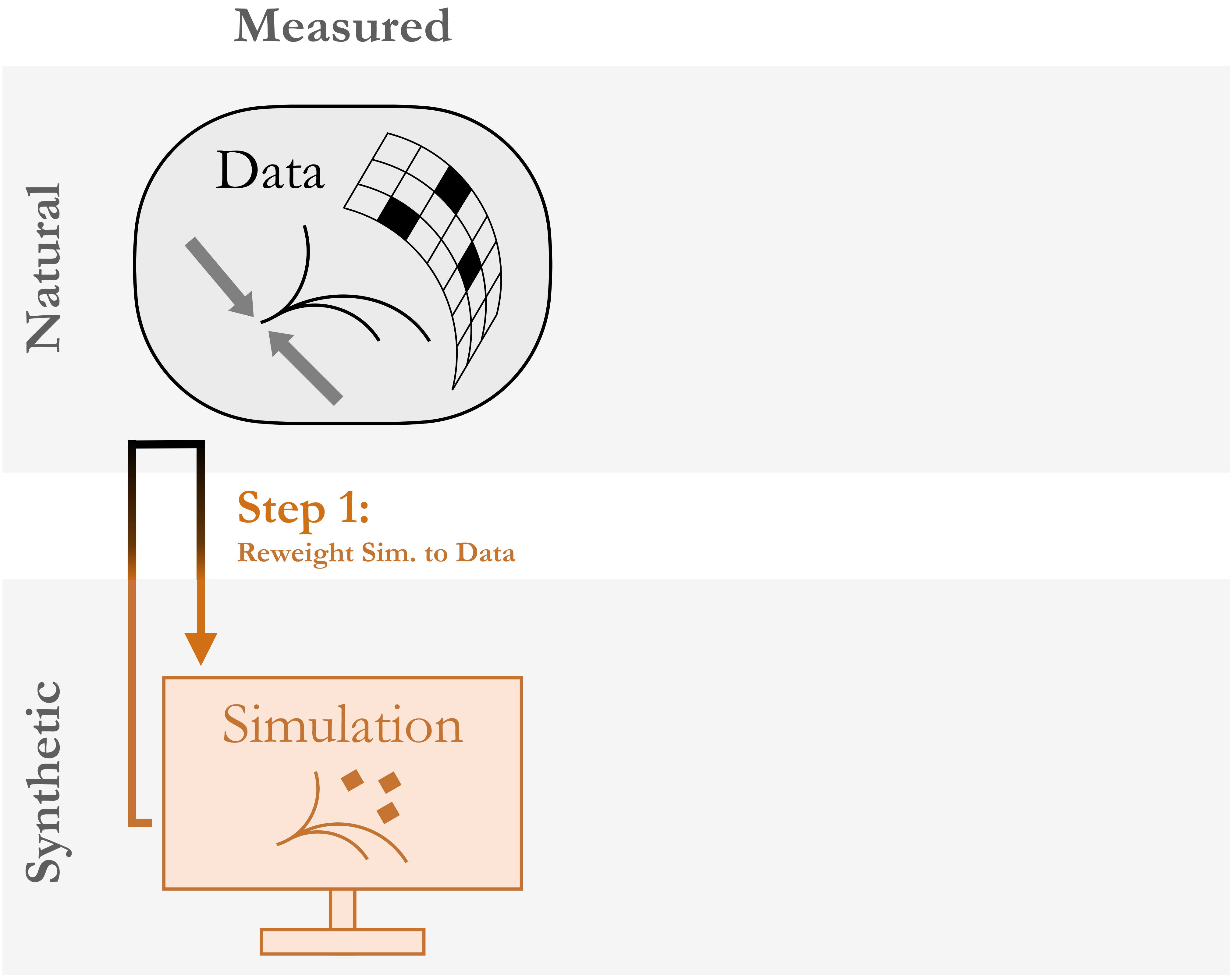
We call this Deep neural networks using Classification for Tuning and Reweighting or "**DCTR**"

Unfold by iterating: OmniFold



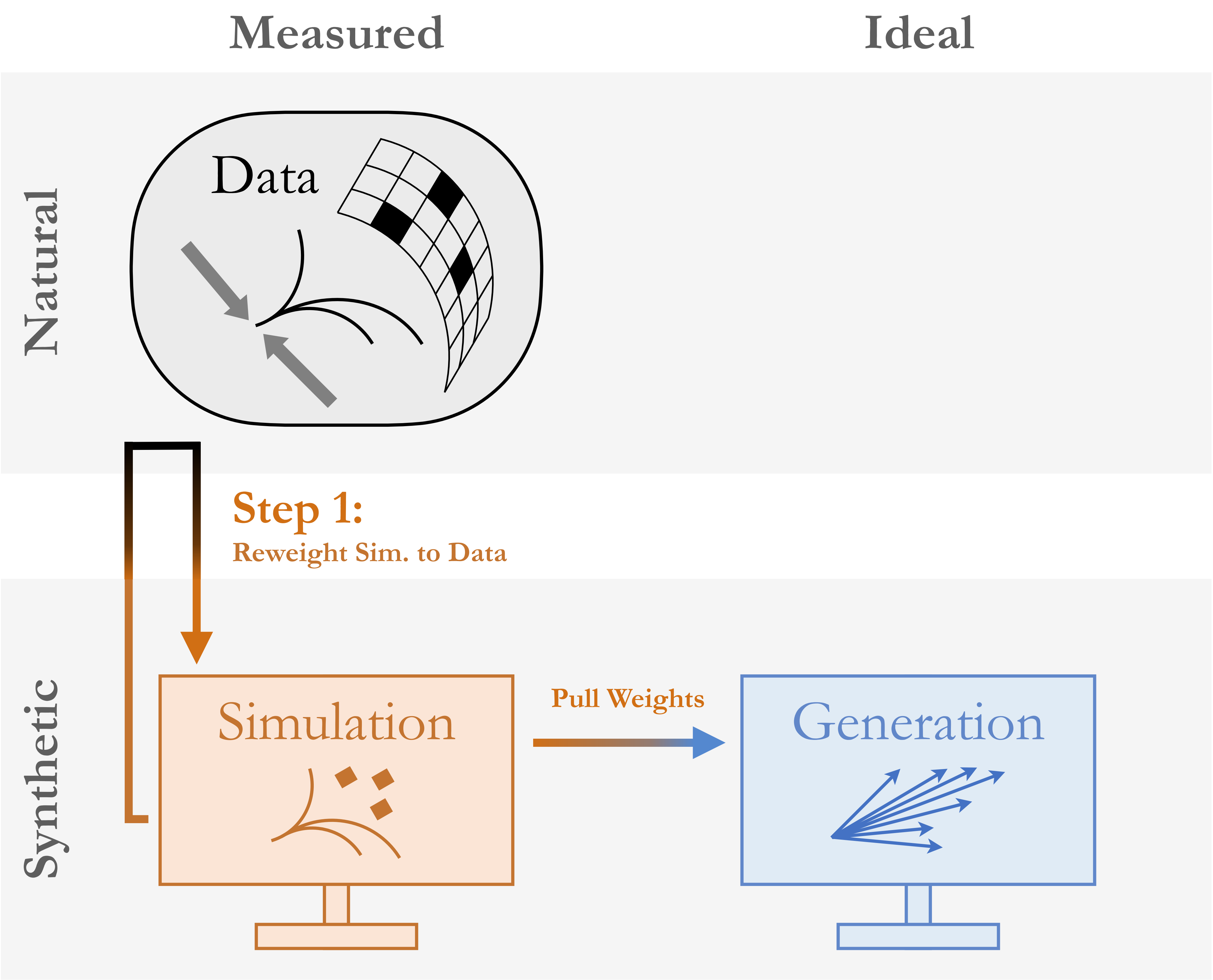
A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001

Unfold by iterating: OmniFold



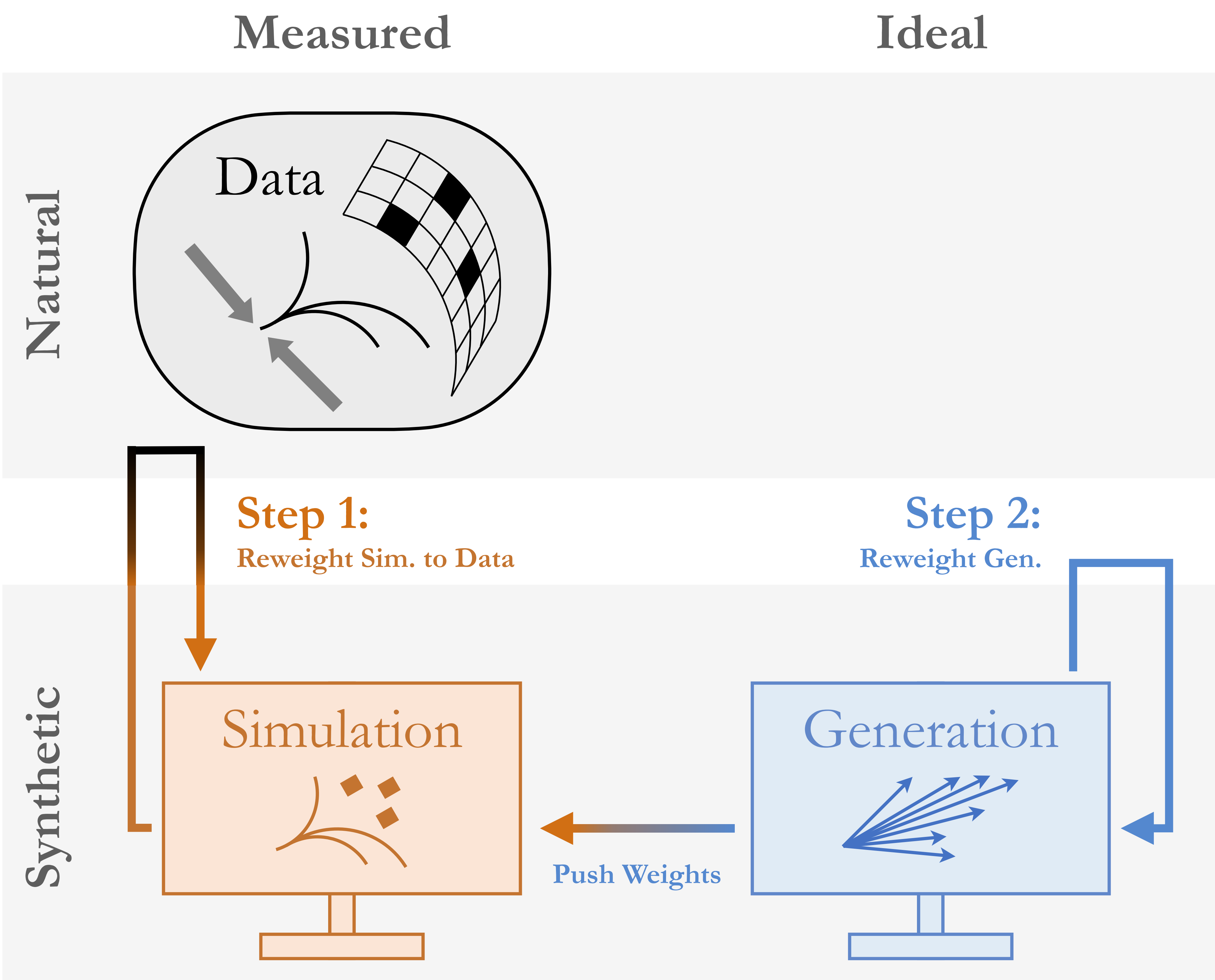
A. Andreassen, P. Komiske, E. Metodiev,
BPN, J. Thaler, PRL 124 (2020) 182001

Unfold by iterating: OmniFold



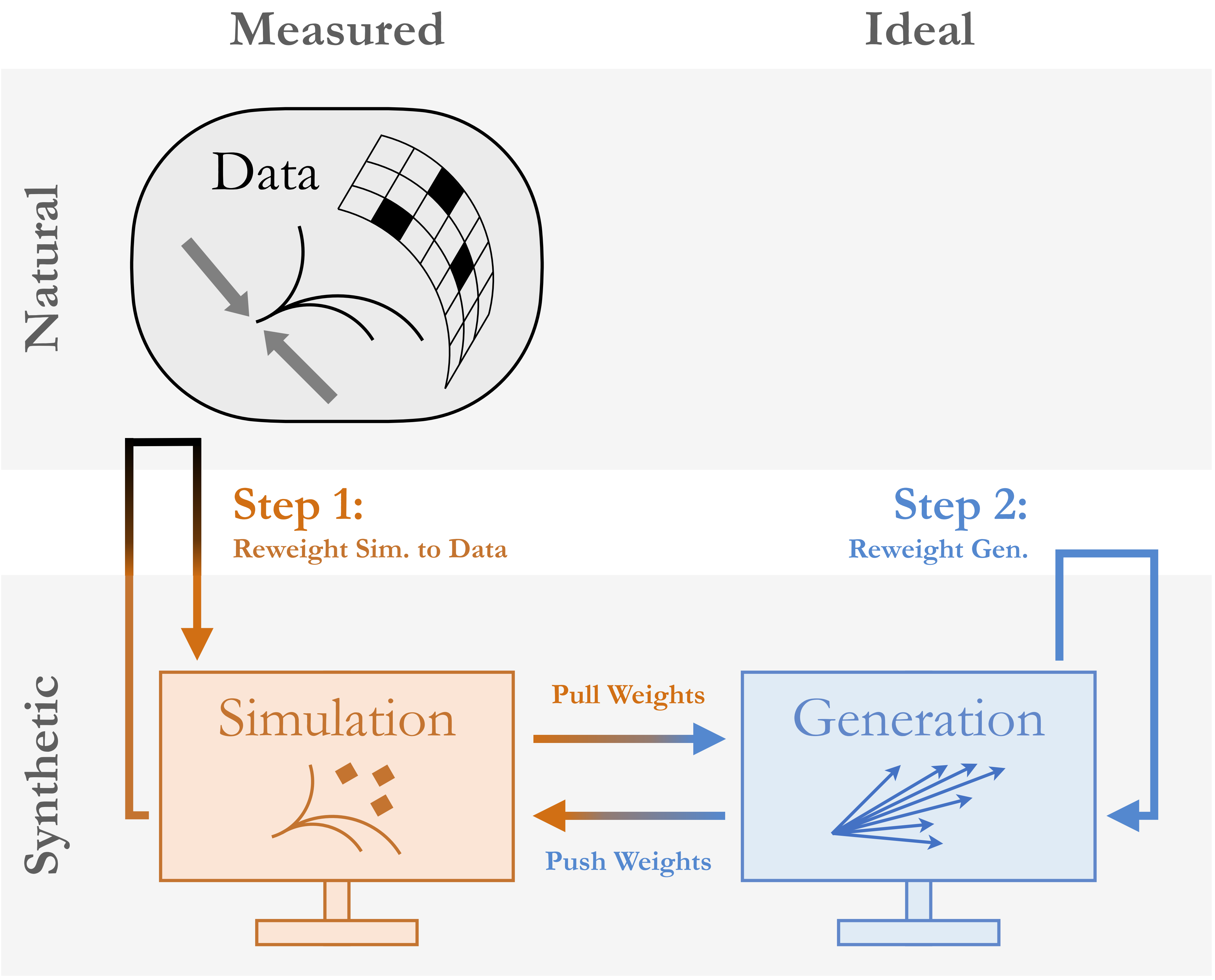
A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001

Unfold by iterating: OmniFold



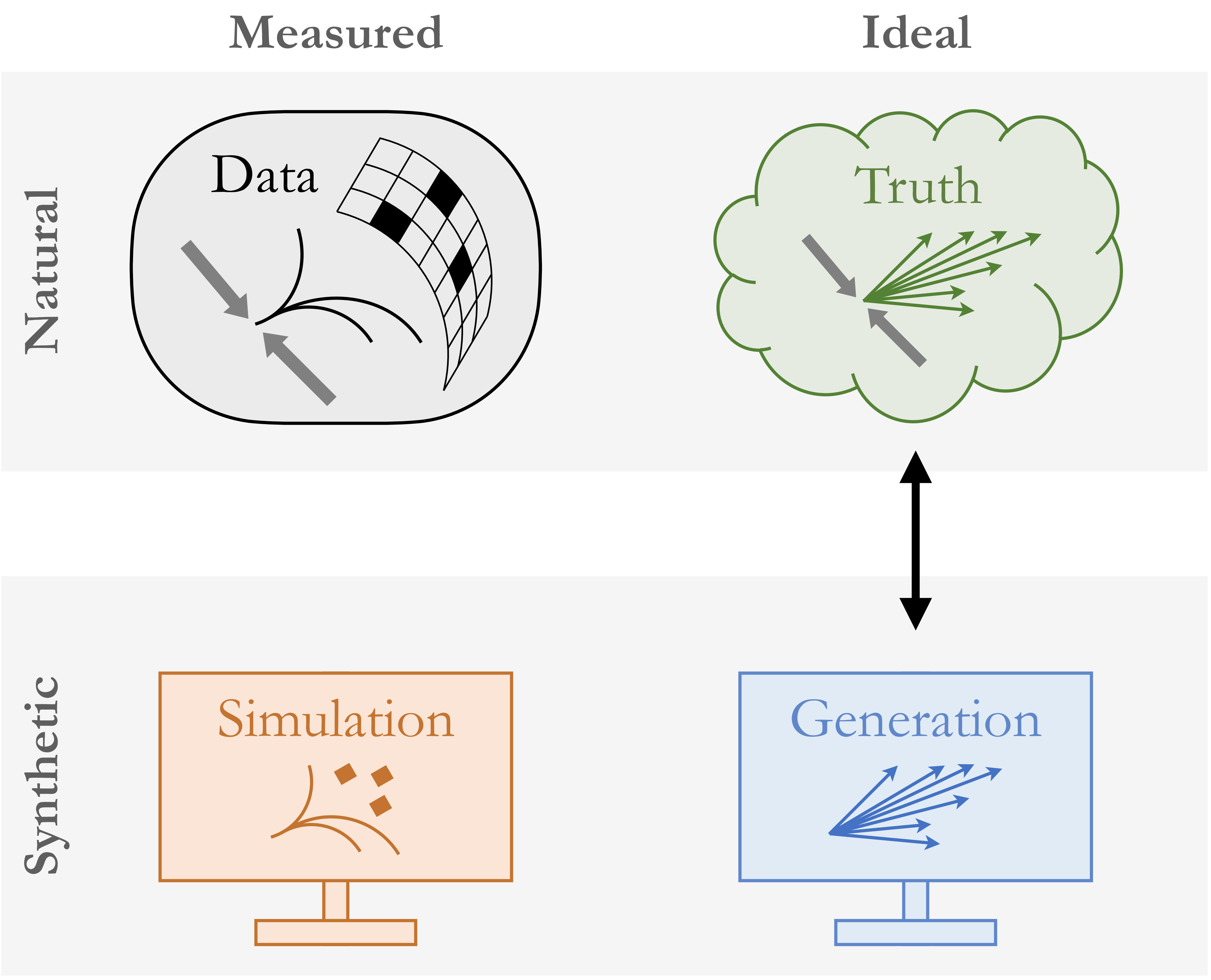
A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001

Unfold by iterating: OmniFold



A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001

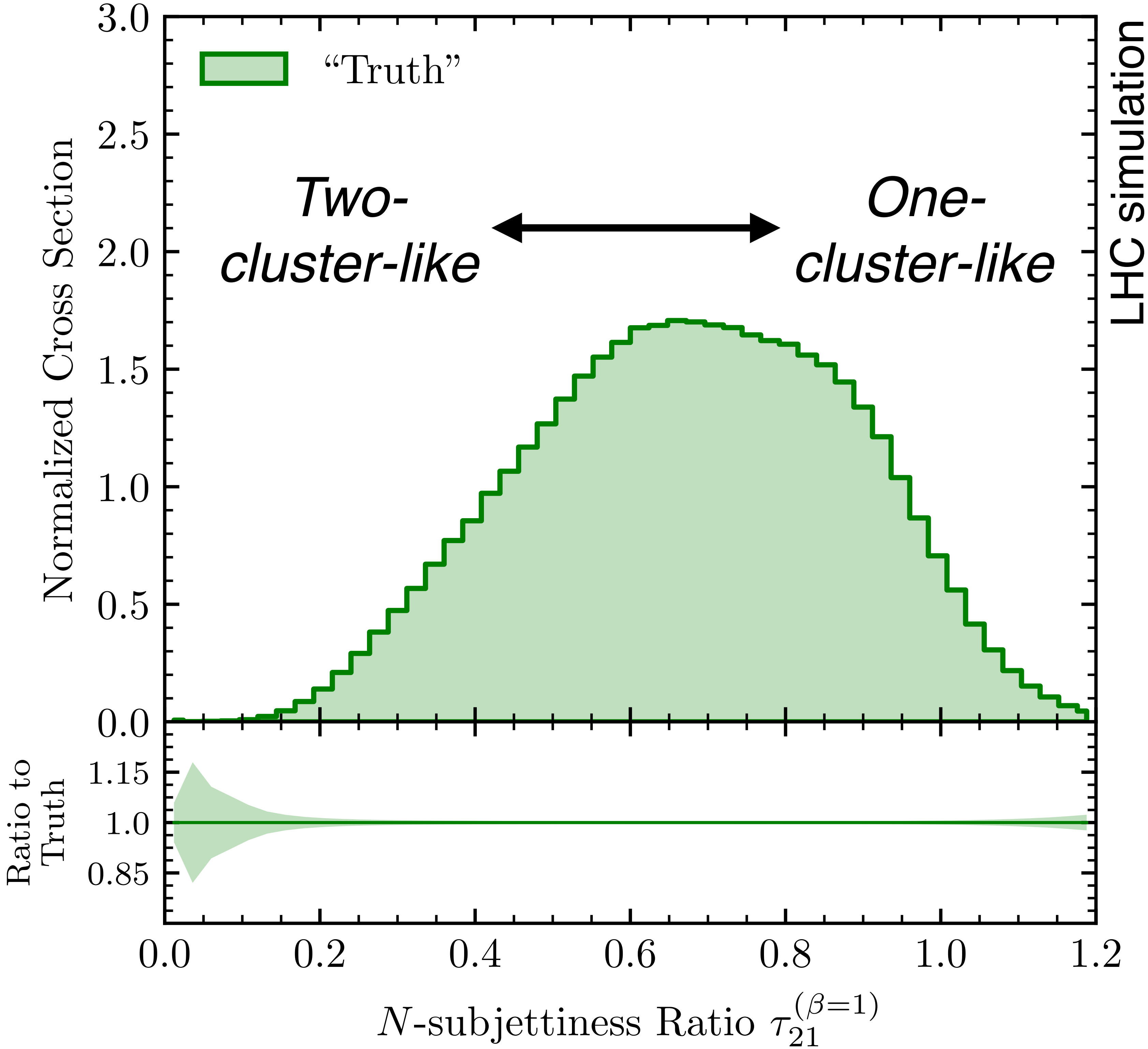
Unfold by iterating: OmniFold



A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001

Results

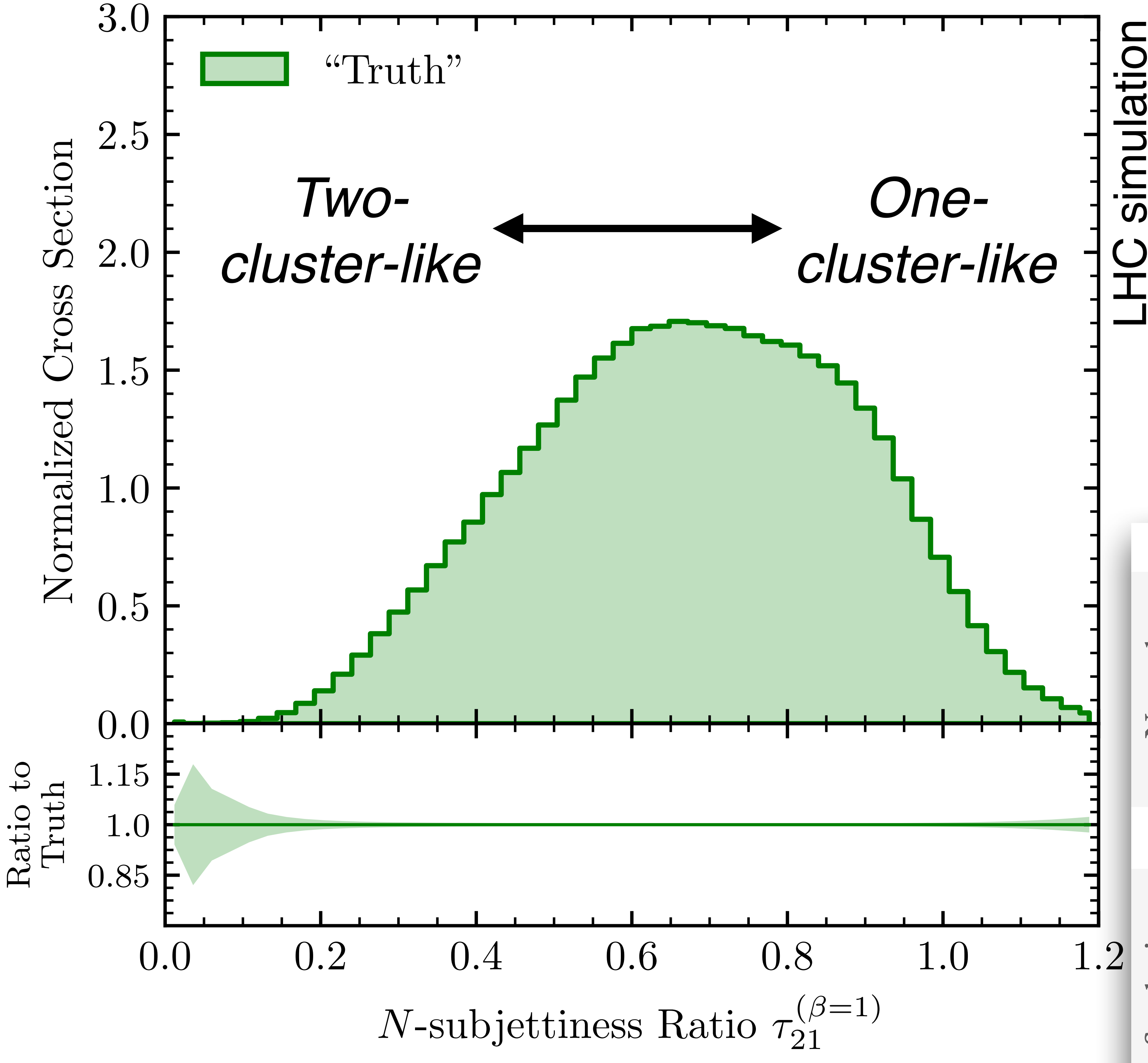
A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001



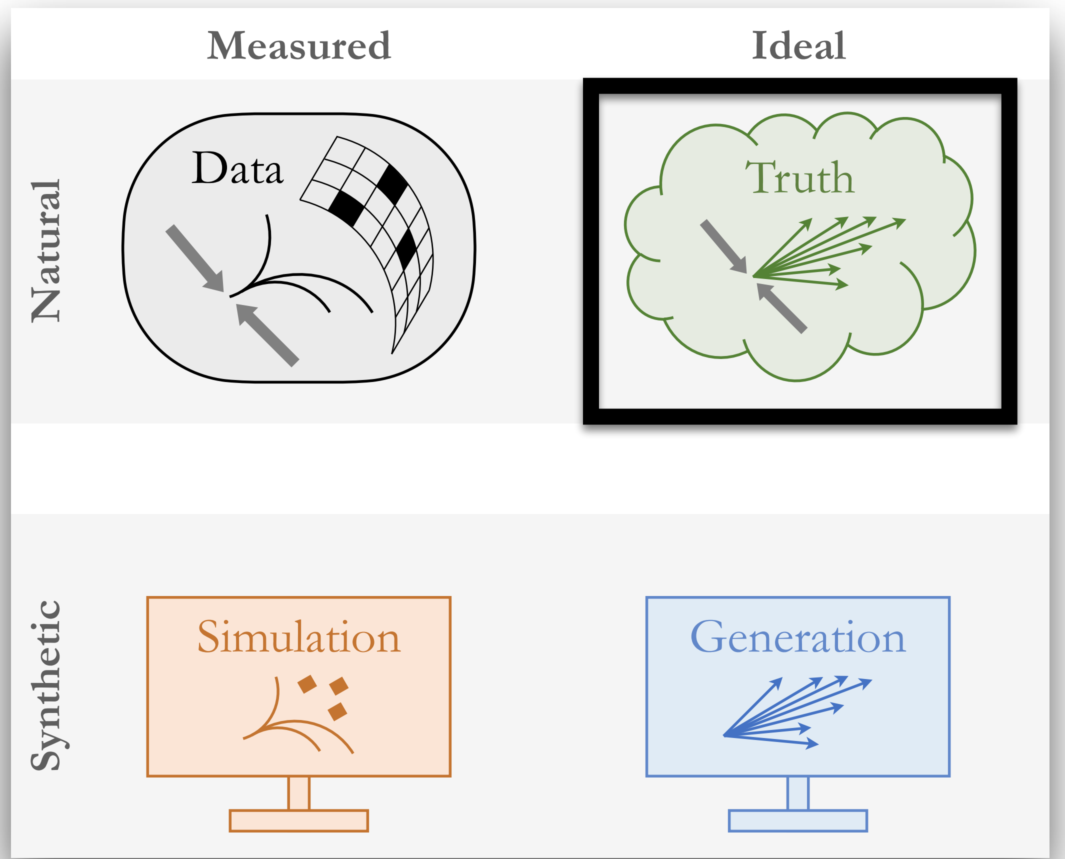
Consider this observable, which characterizes the substructure

Results

A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001

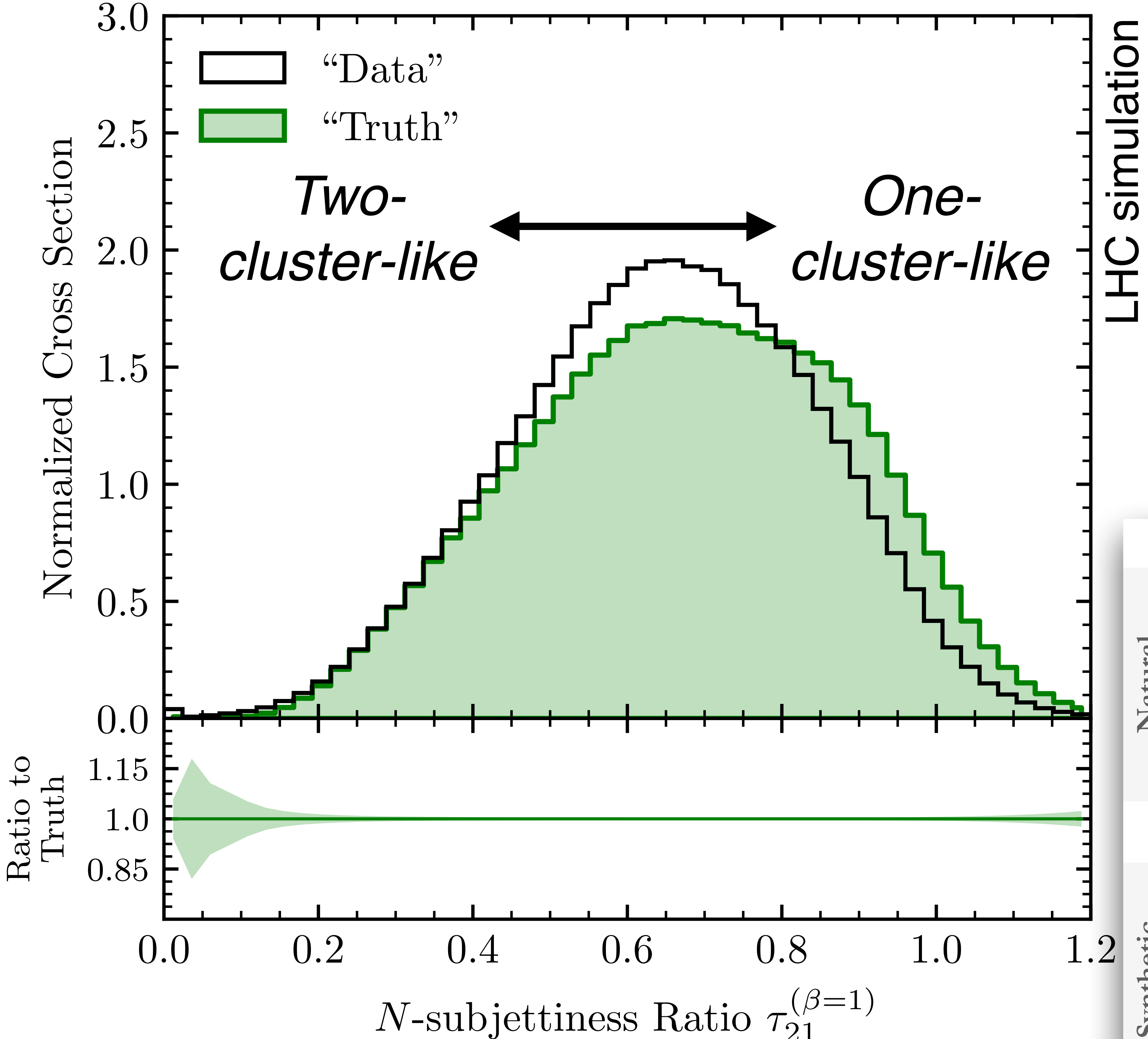


Consider this observable, which characterizes the substructure

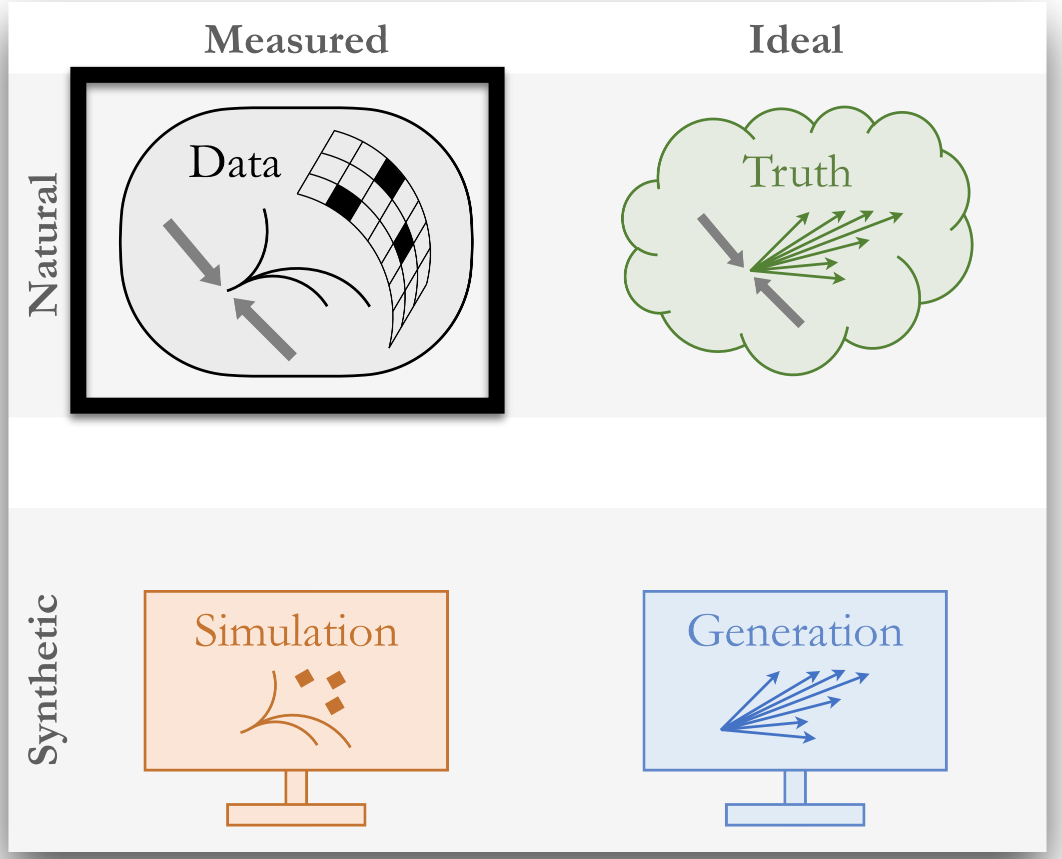


Results

A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001

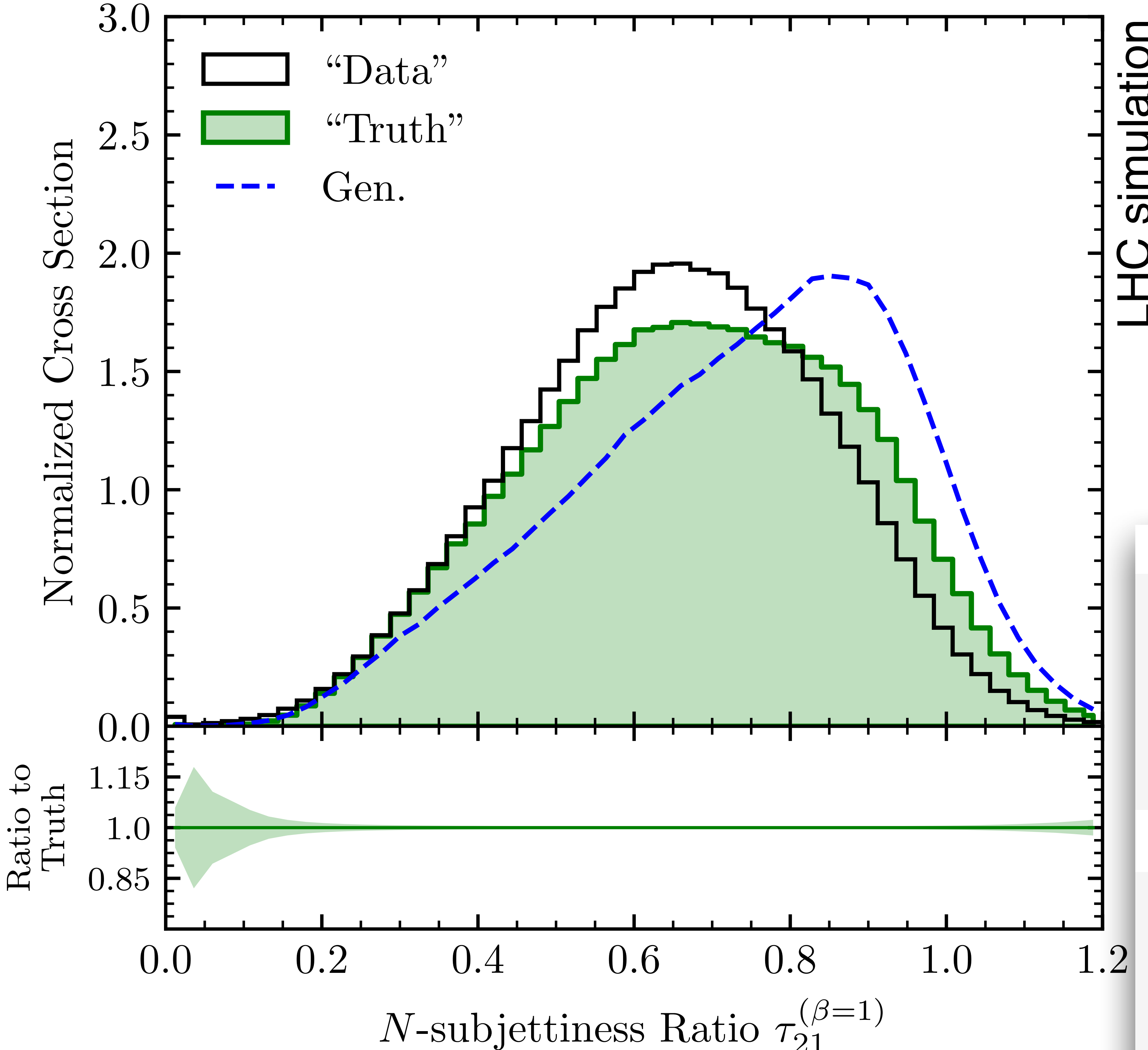


Consider this observable, which characterizes the substructure

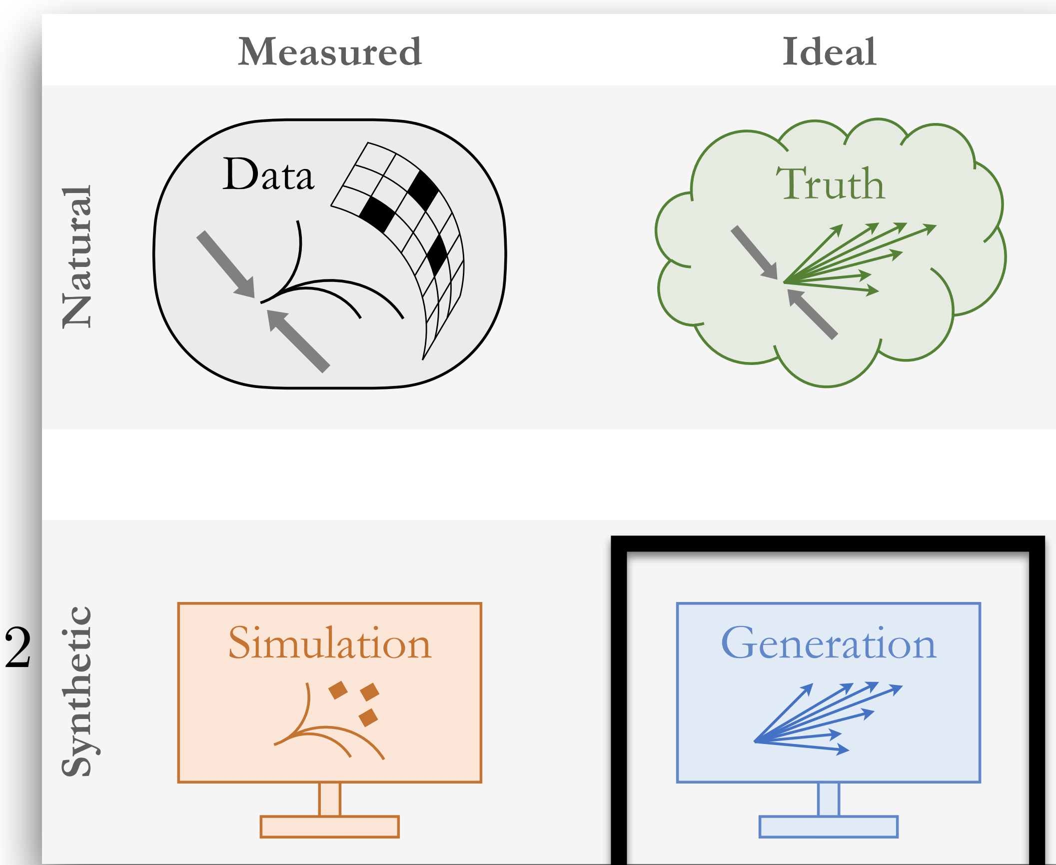


Results

A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001

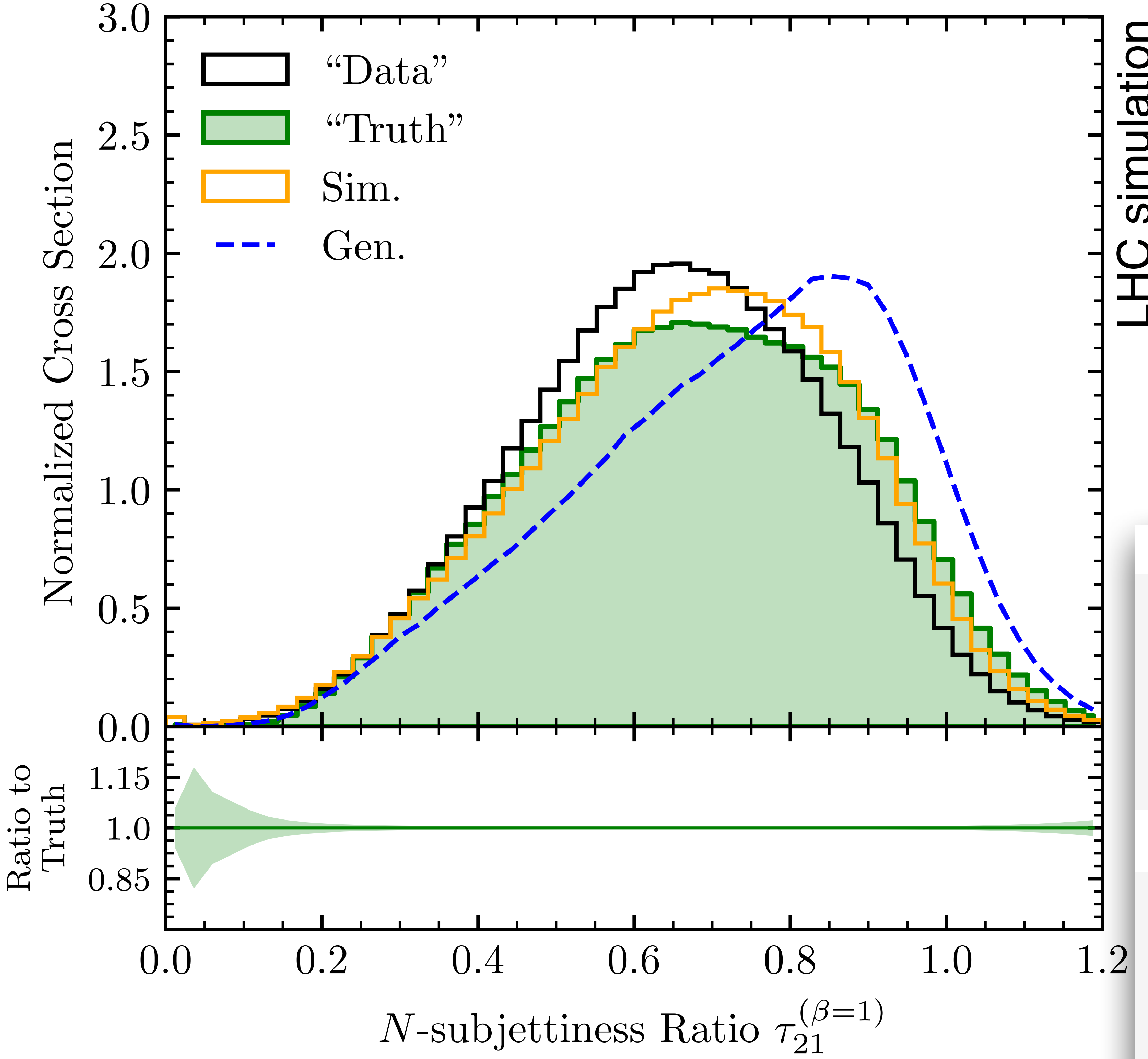


Consider this observable, which characterizes the substructure

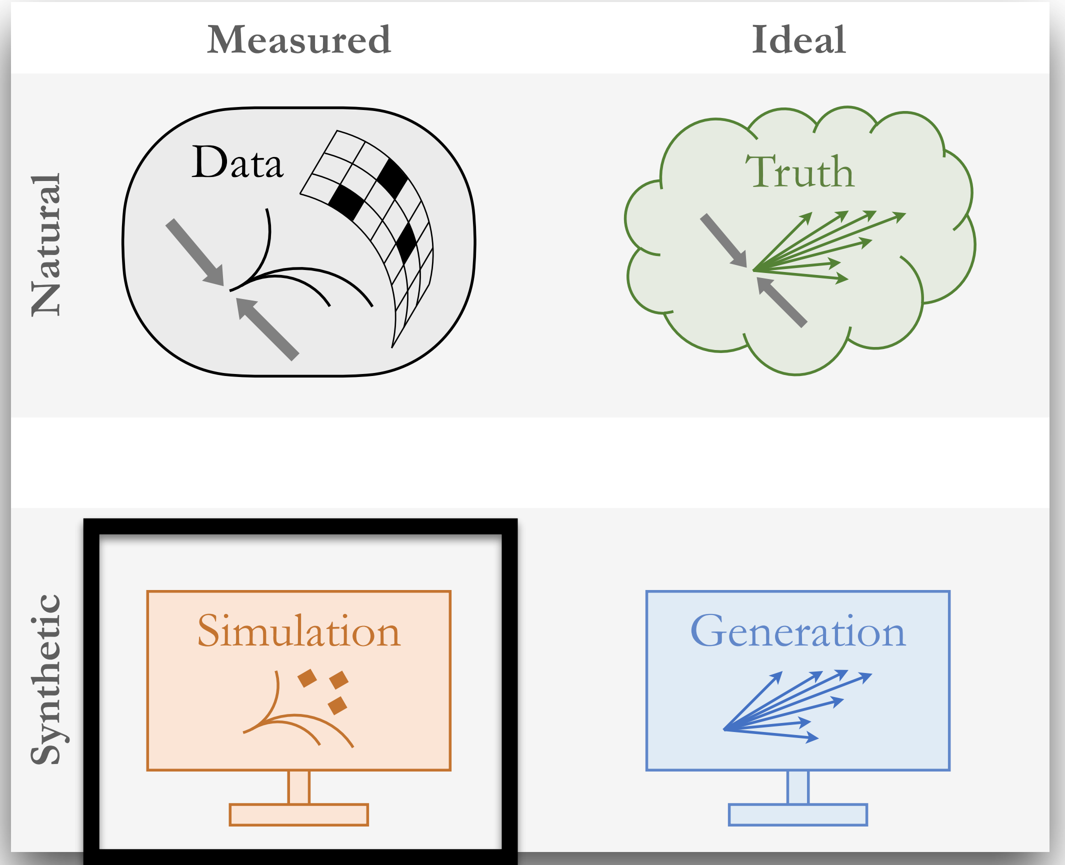


Results

A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001

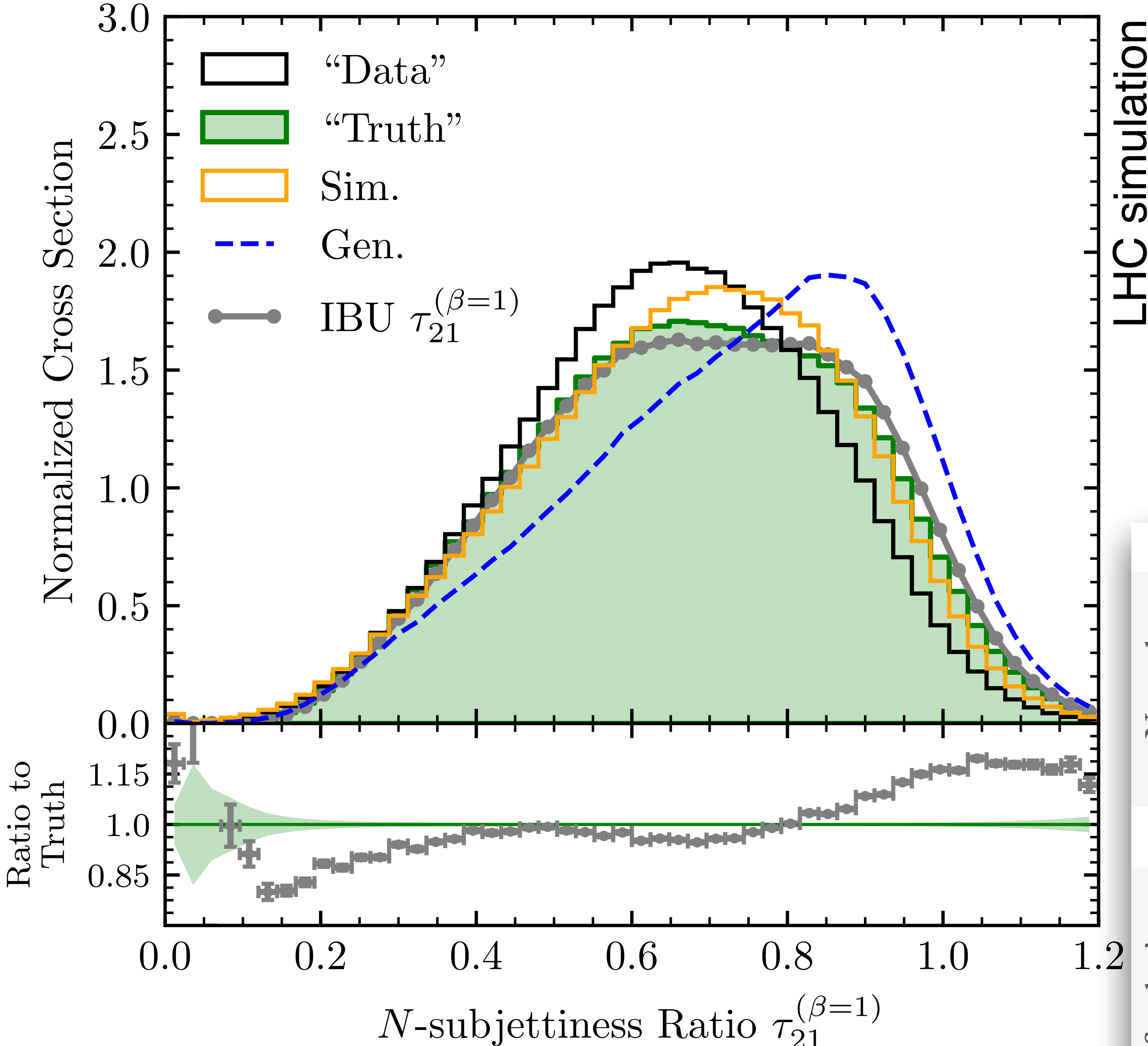


Consider this observable, which characterizes the substructure

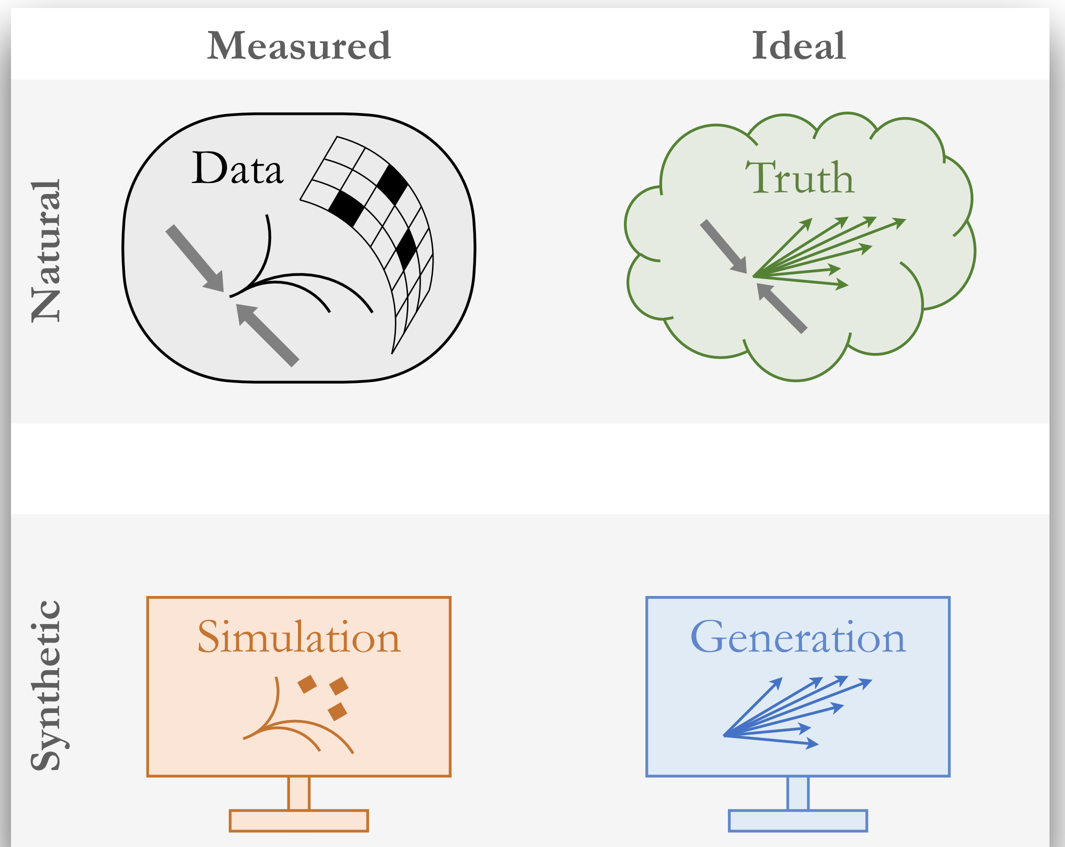


Results

A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001

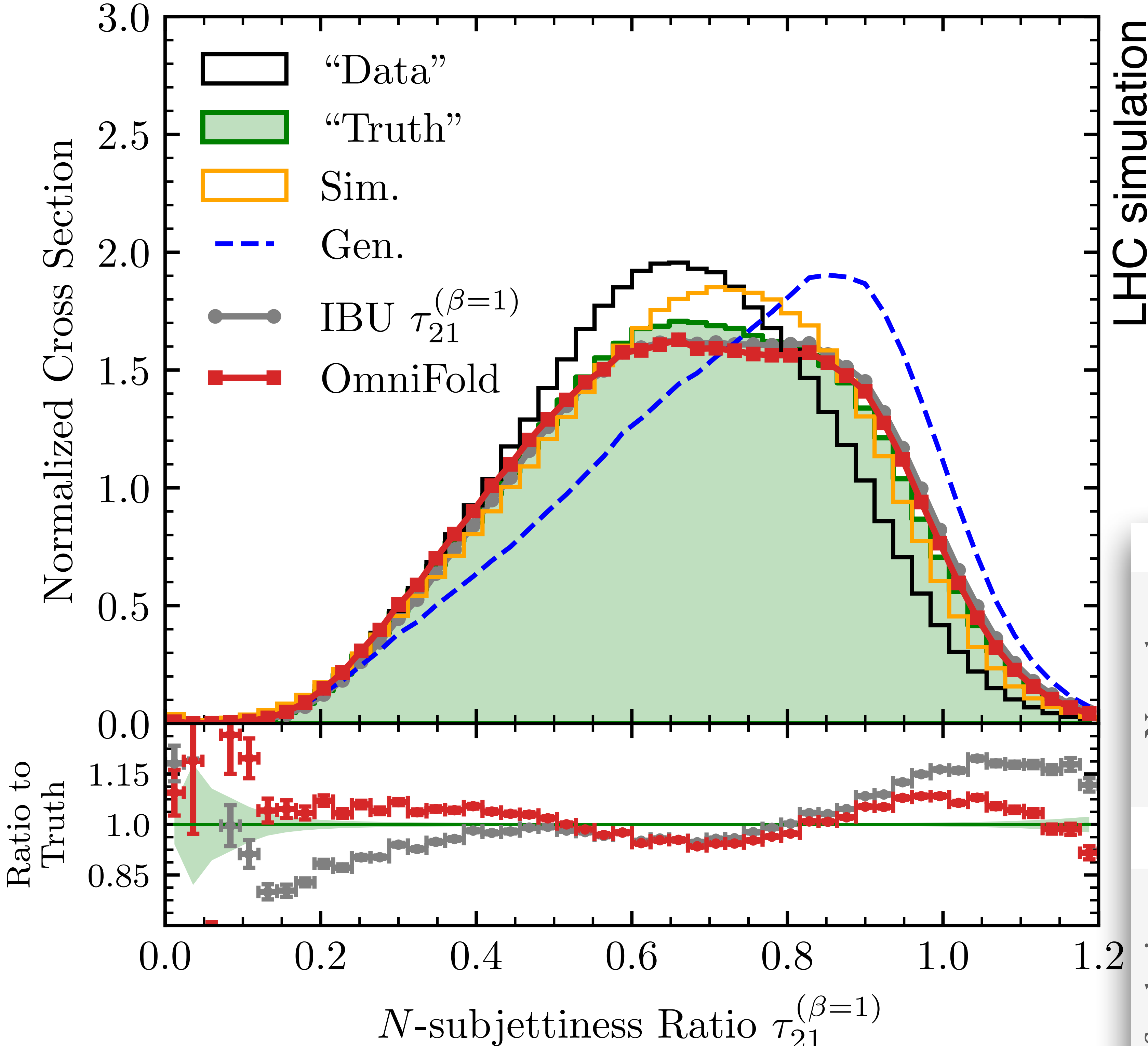


IBU is the current standard. It is a1D binned and iterative approach.

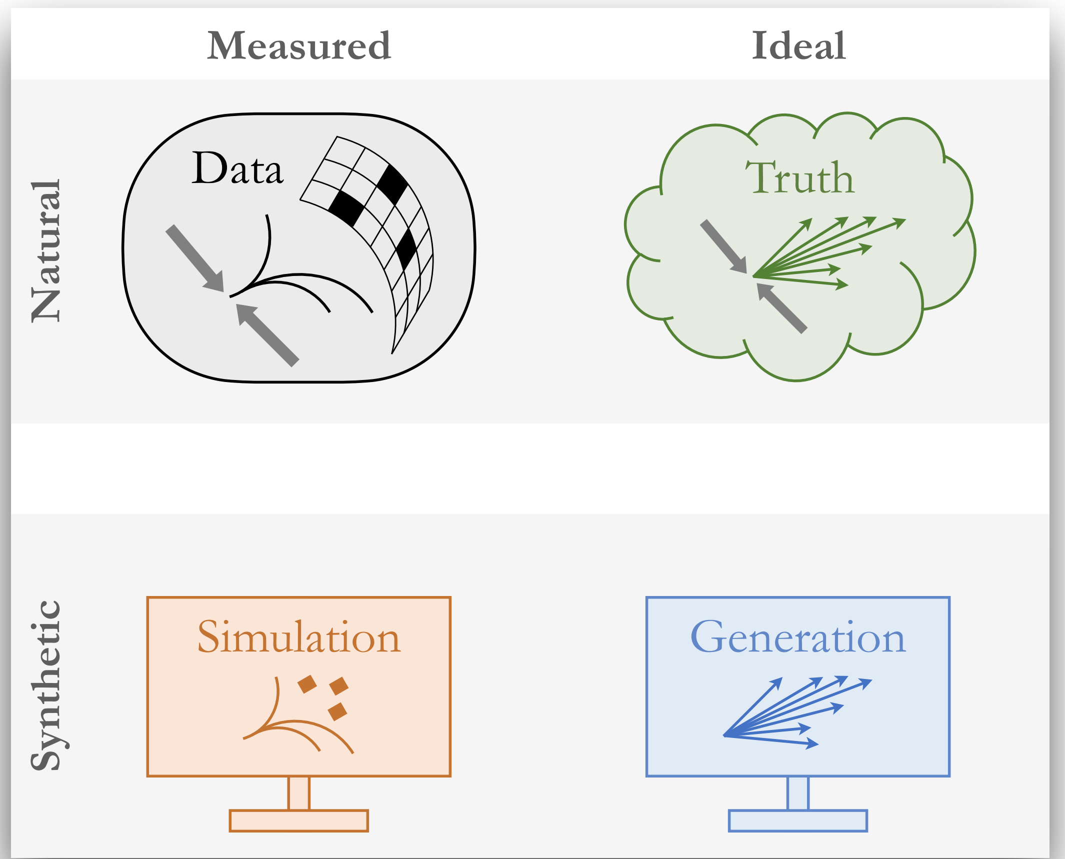


Results

A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001

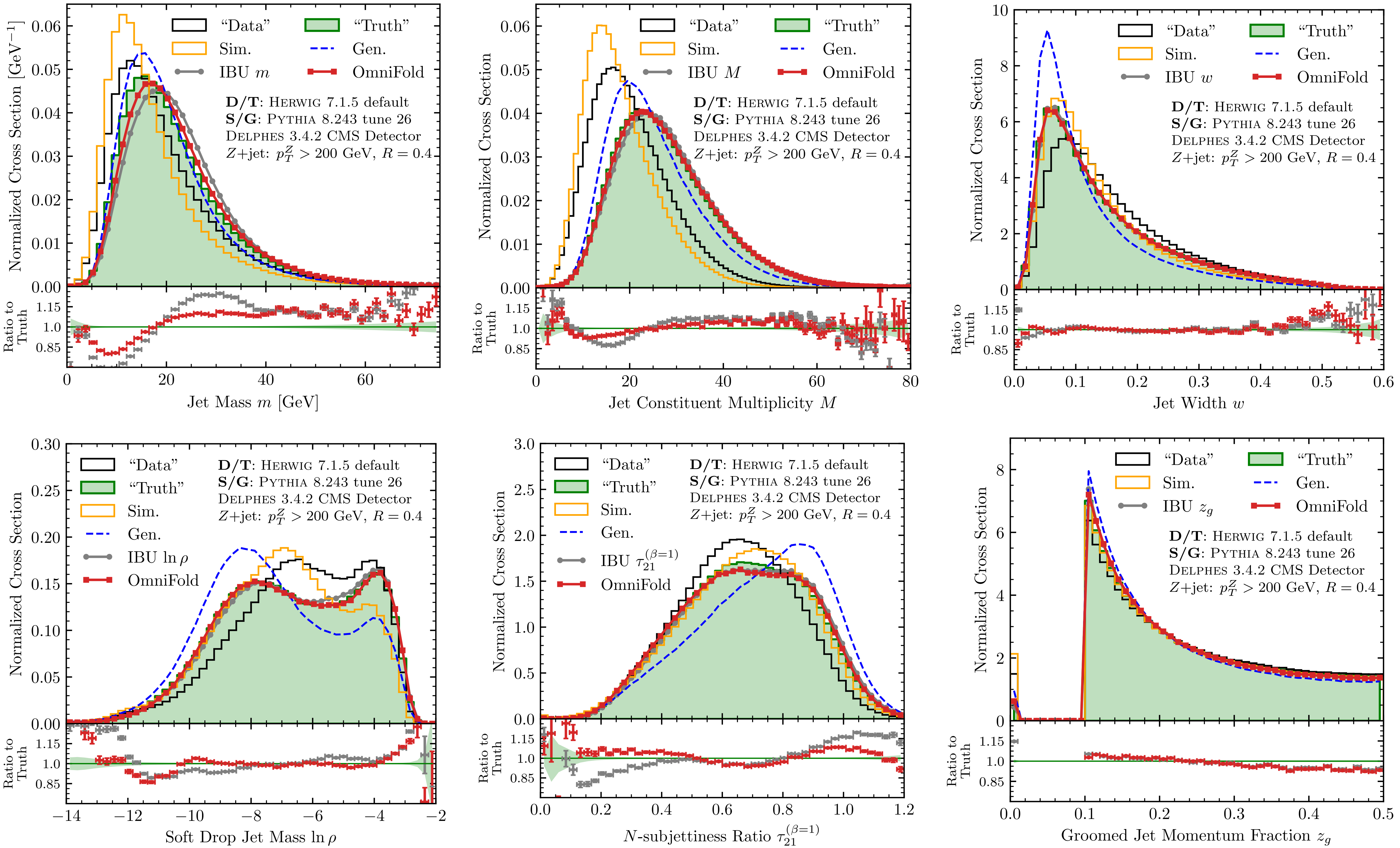


OmniFold outperforms IBU even though it is not tailored to this observable



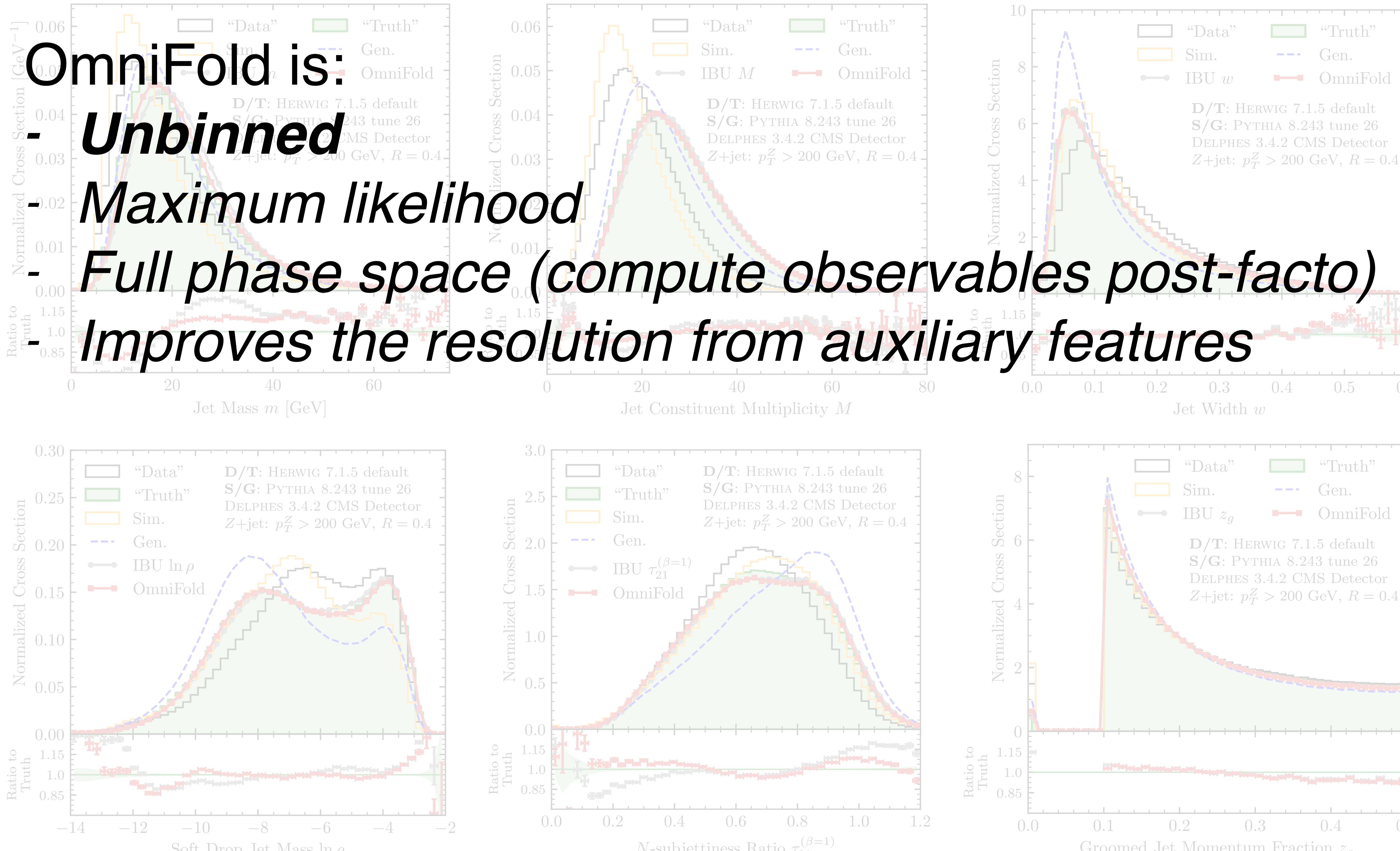
Results

A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, PRL 124 (2020) 182001



Results

[A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, 1907.08209]



OmniFold is:

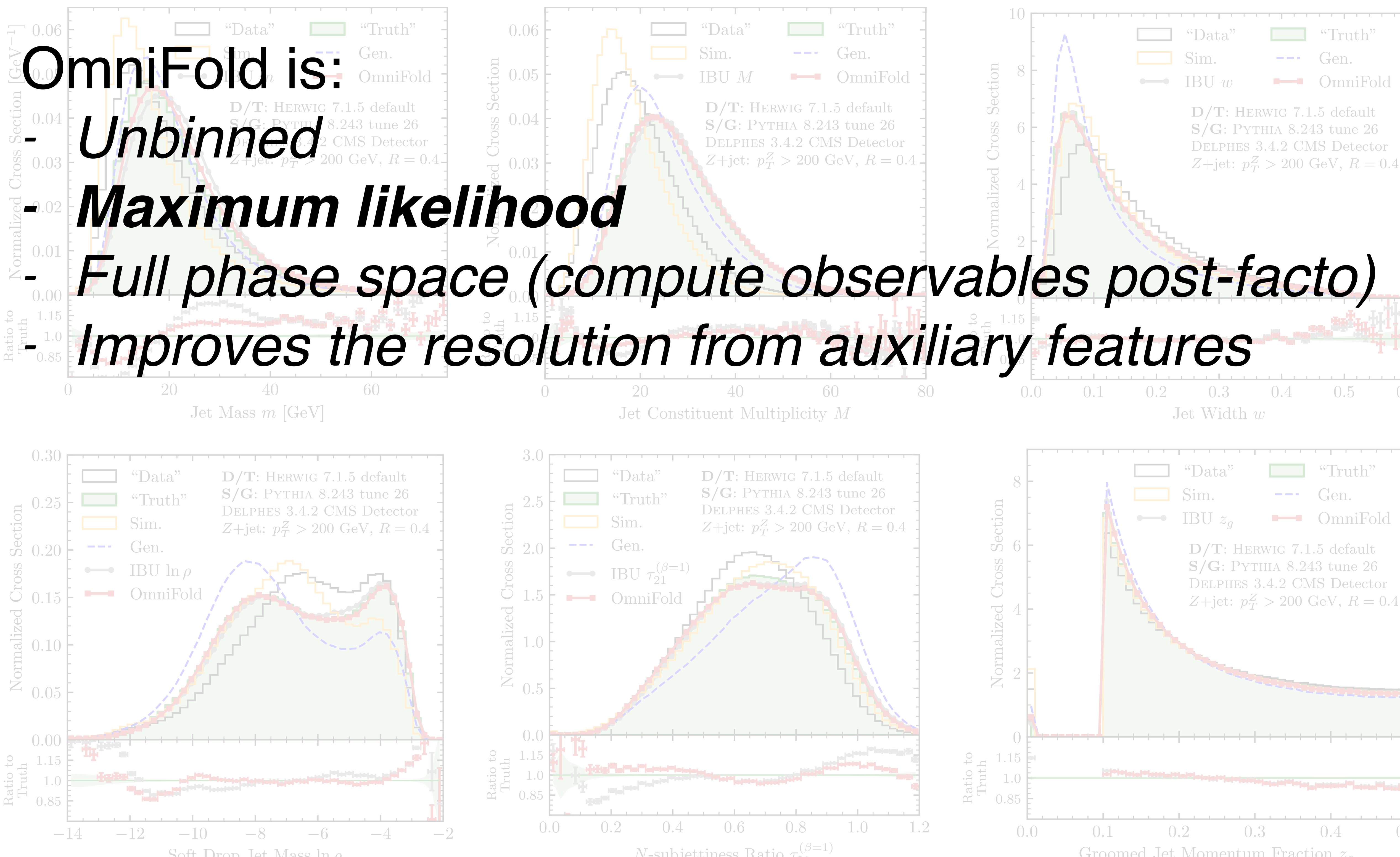
- ***Unbinned***
- *Maximum likelihood*
- *Full phase space (compute observables post-facto)*
- *Improves the resolution from auxiliary features*

Results

[A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, 1907.08209]

OmniFold is:

- *Unbinned*
- ***Maximum likelihood***
- *Full phase space (compute observables post-facto)*
- *Improves the resolution from auxiliary features*

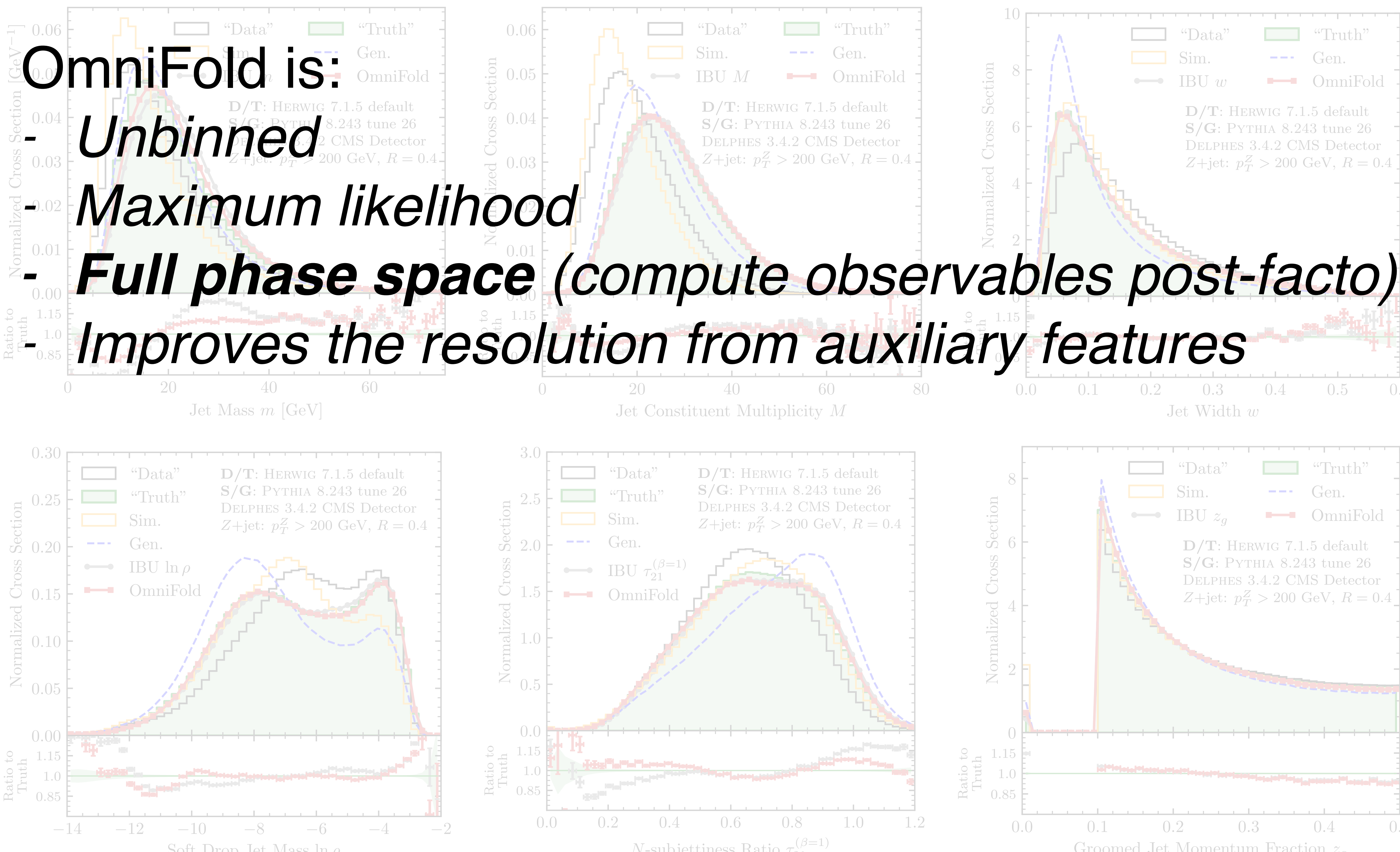


Results

[A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, 1907.08209]

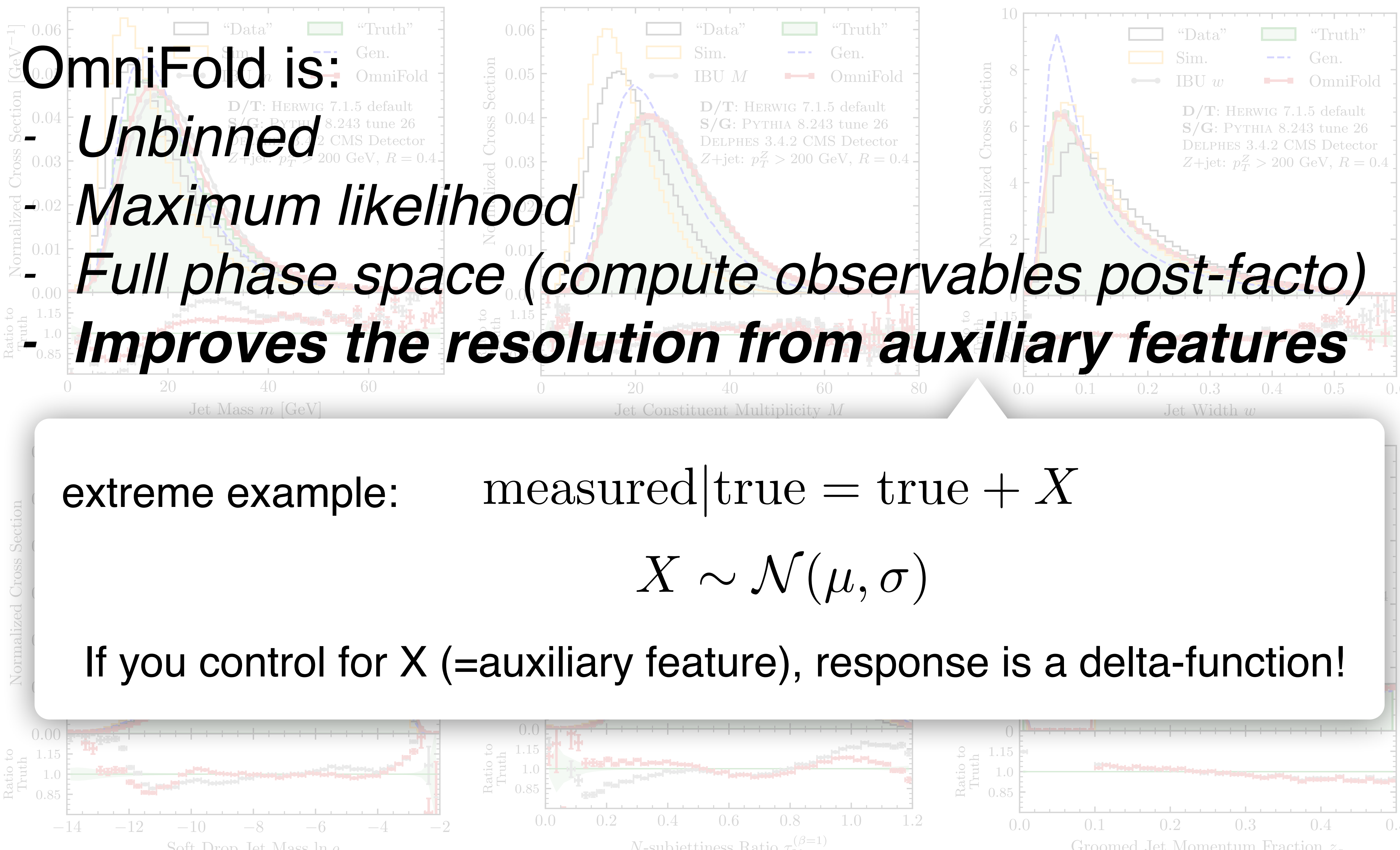
OmniFold is:

- *Unbinned*
- *Maximum likelihood*
- ***Full phase space*** *(compute observables post-facto)*
- *Improves the resolution from auxiliary features*



Results

[A. Andreassen, P. Komiske, E. Metodiev, BPN, J. Thaler, 1907.08209]



OmniFold is:

- *Unbinned*
- *Maximum likelihood*
- *Full phase space (compute observables post-facto)*
- ***Improves the resolution from auxiliary features***

extreme example: $$\text{measured}|\text{true} = \text{true} + X$$

$$X \sim \mathcal{N}(\mu, \sigma)$$

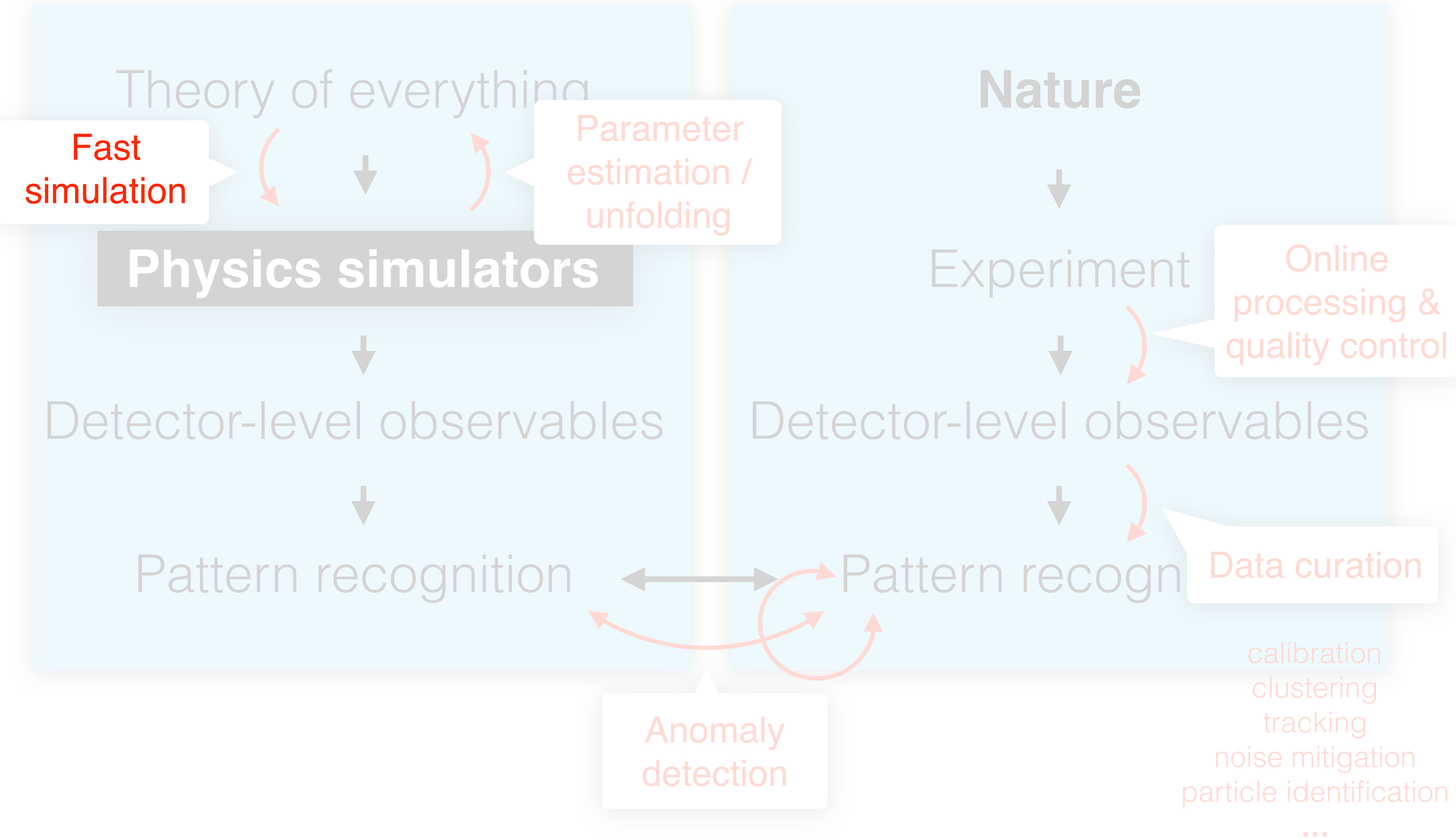
If you control for X (=auxiliary feature), response is a delta-function!

Partial Conclusions for Inference

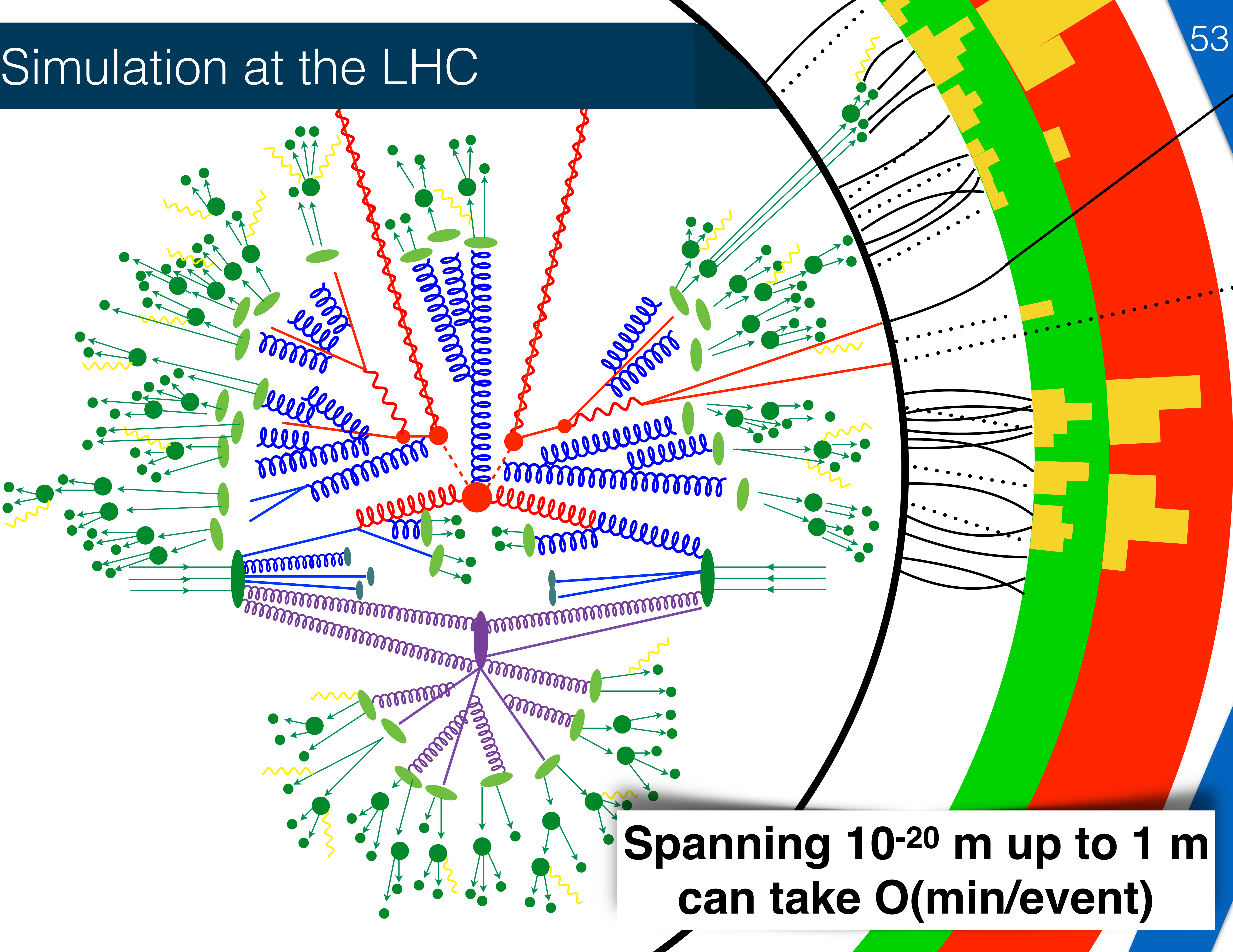
One of the features of HEP that distinguishes it from other fields is the availability of a high-fidelity simulation (thanks to MCNet collaborators!)

These simulations are usually expensive and non-differentiable. A variety of ML methods can scaffold on top of our simulators to allow us to use all their physics to extract the most information from our data.

Data analysis in HEP + Machine Learning

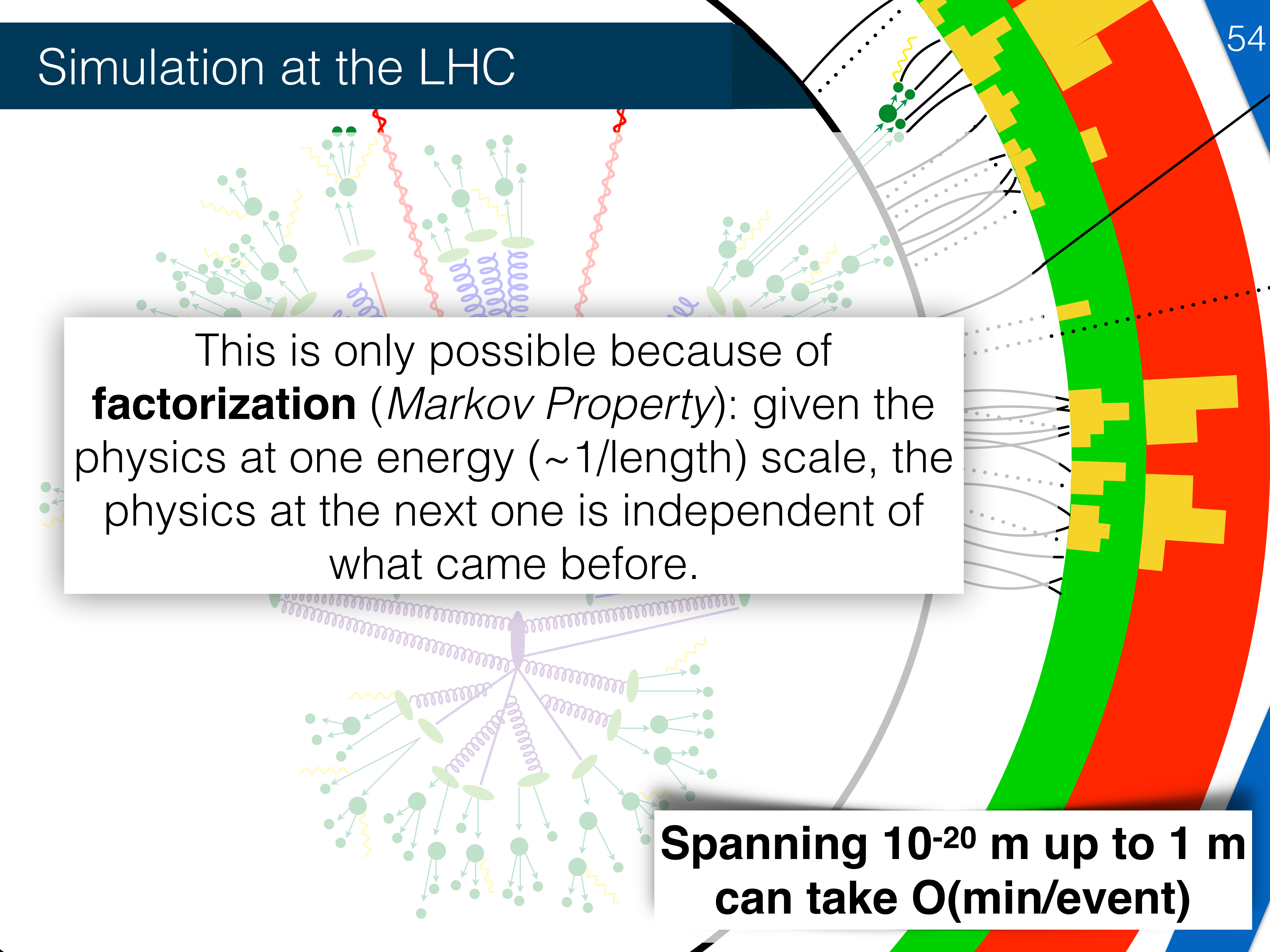


Simulation at the LHC



Spanning 10^{-20} m up to 1 m can take O(min/event)

Simulation at the LHC



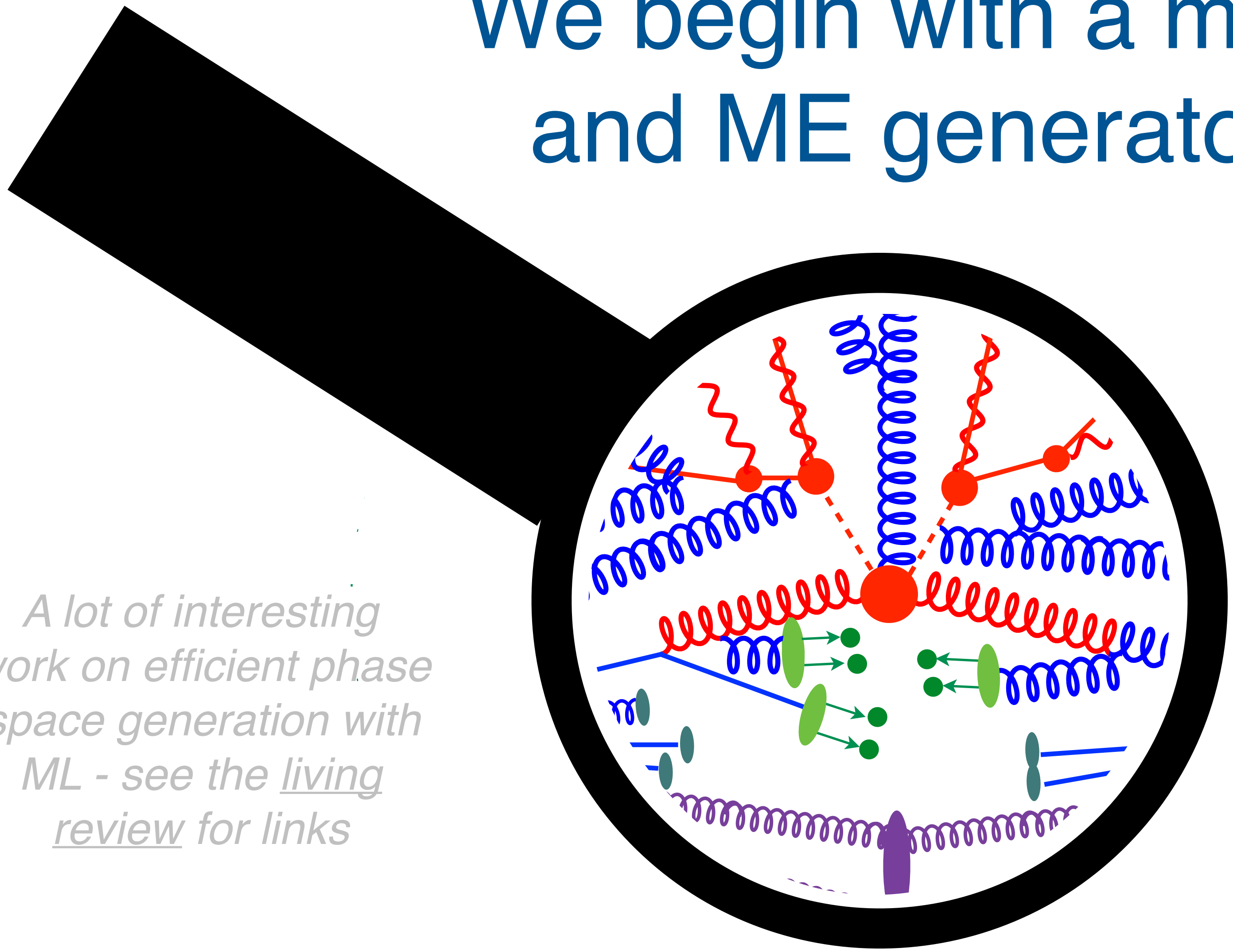
This is only possible because of **factorization** (*Markov Property*): given the physics at one energy (~1/length) scale, the physics at the next one is independent of what came before.

Spanning 10^{-20} m up to 1 m can take O(min/event)

Part I: Hard-scatter

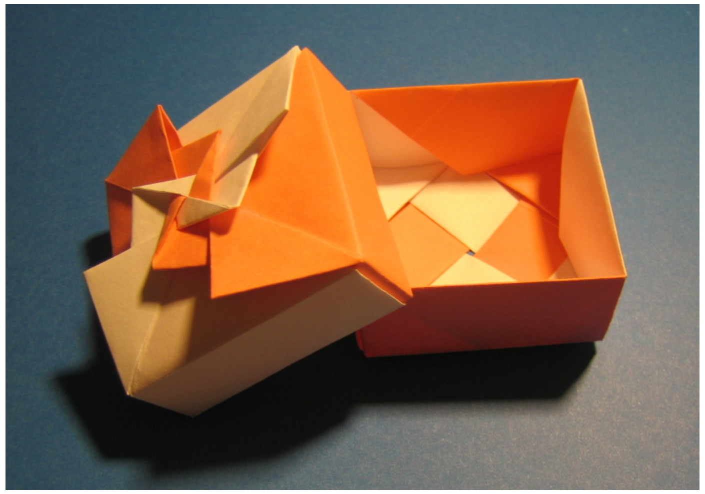
We begin with a model and ME generators.

$$
\begin{aligned}
\mathcal{L} = &-\frac{1}{4}F_{\mu\nu}F^{\mu\nu} \\
&+ i\bar{\psi}\not{D}\psi \\
&+ \psi_i y_{ij}\psi_j\phi + \text{h.c.} \\
&+ |D_\mu\phi|^2 - V(\phi) \\
&+ ???
\end{aligned}
$$



A lot of interesting work on efficient phase space generation with ML - see the living review for links

```
************************************************************
*                                                          *
*                    W E L C O M E  to                     *
*             M A D G R A P H 5 _ a M C @ N L O            *
*                                                          *
*                                                          *
*                 *                       *                *
*                   *        * *        *                  *
*                     * * * * 5 * * * *                    *
*                   *        * *        *                  *
*                 *                       *                *
*                                                          *
************************************************************
```

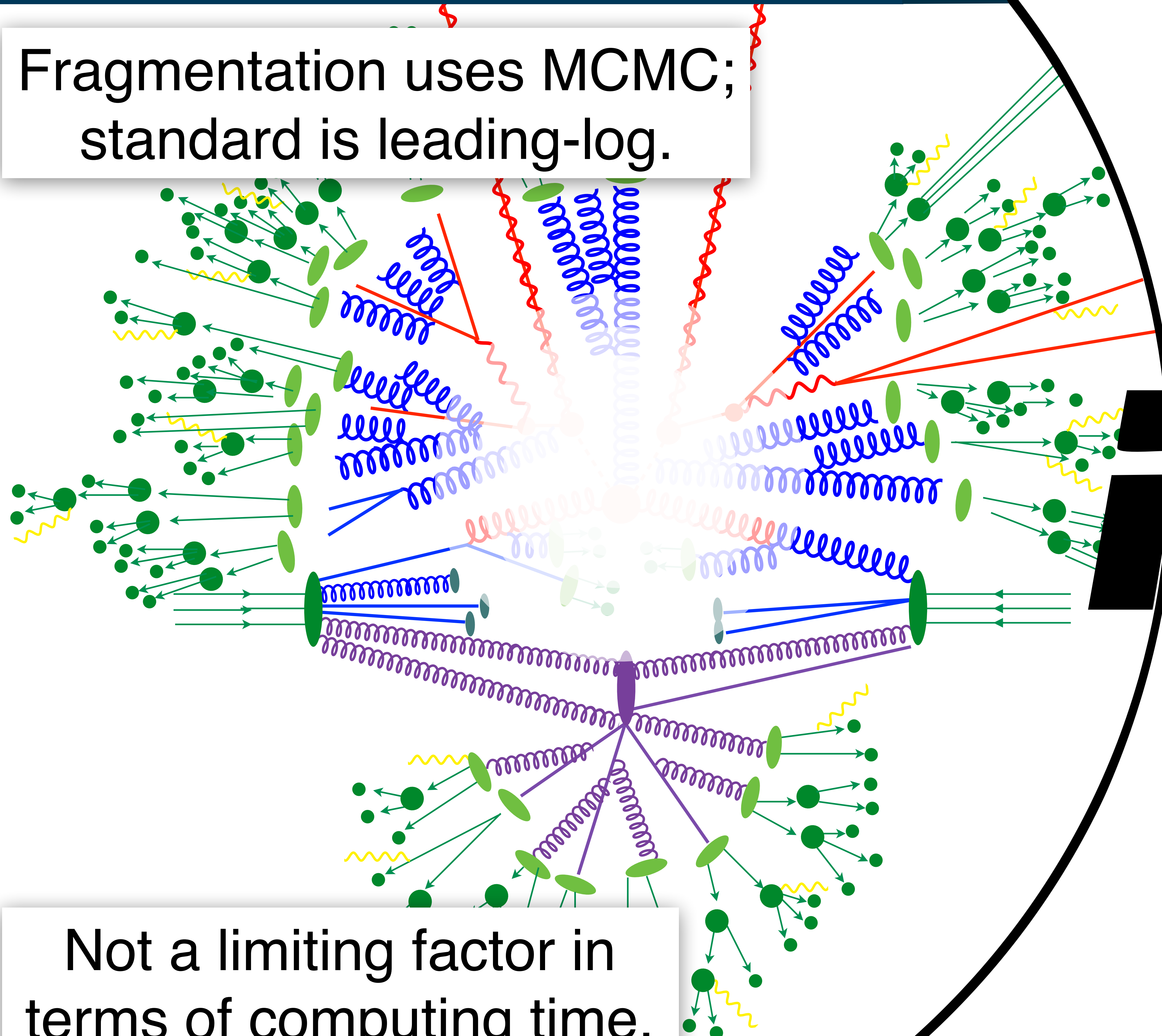


Standard is automated NLO or LO + matched

For many cases, this is slow but not limiting (yet)

Part II: Fragmentation

Fragmentation uses MCMC; standard is leading-log.



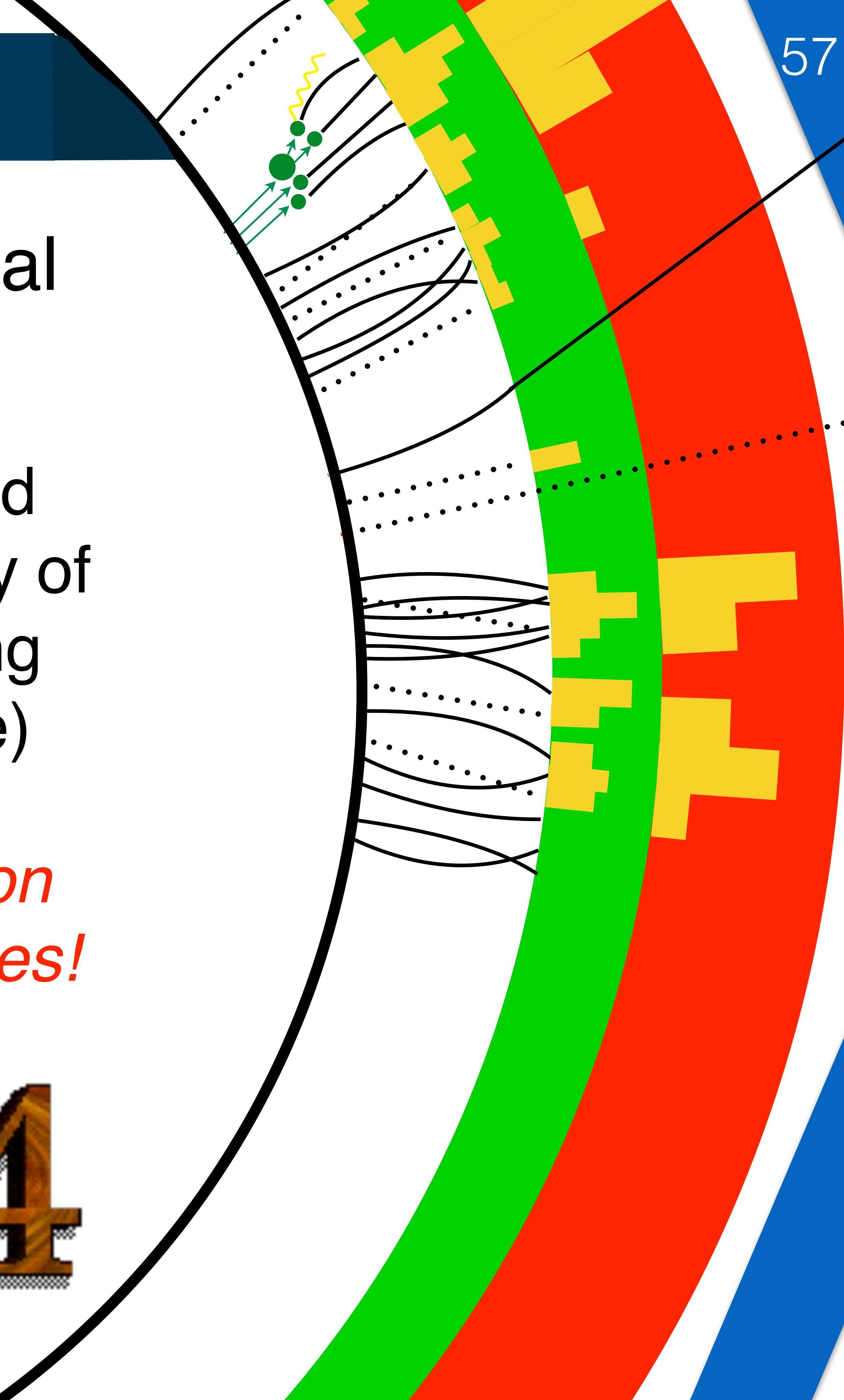
Not a limiting factor in terms of computing time.

State-of-the-art for material interactions is Geant4.

Includes electromagnetic and hadronic physics with a variety of lists for increasing/decreasing accuracy (at the cost of time)

This accounts for O(1) fraction of all HEP competing resources!

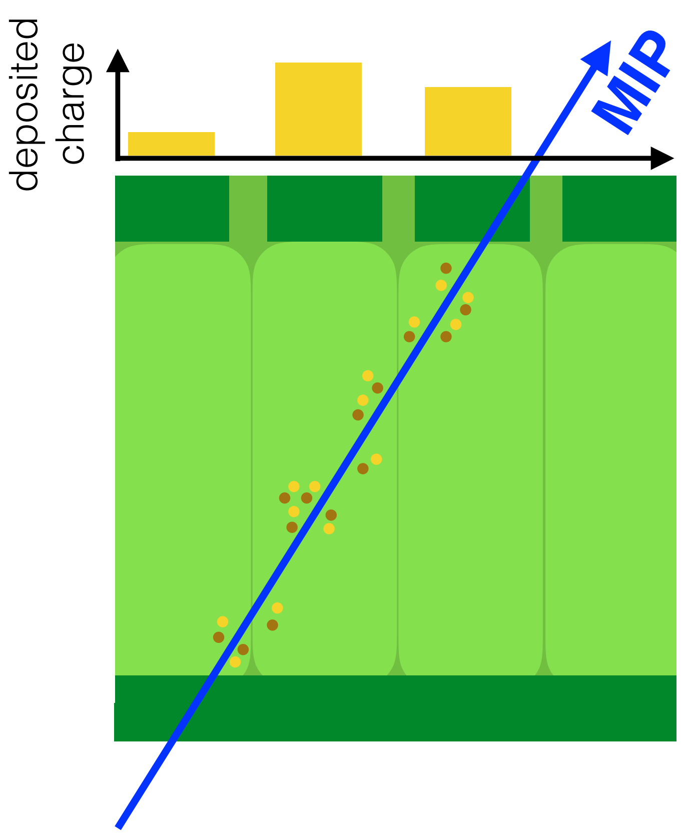
Geant4

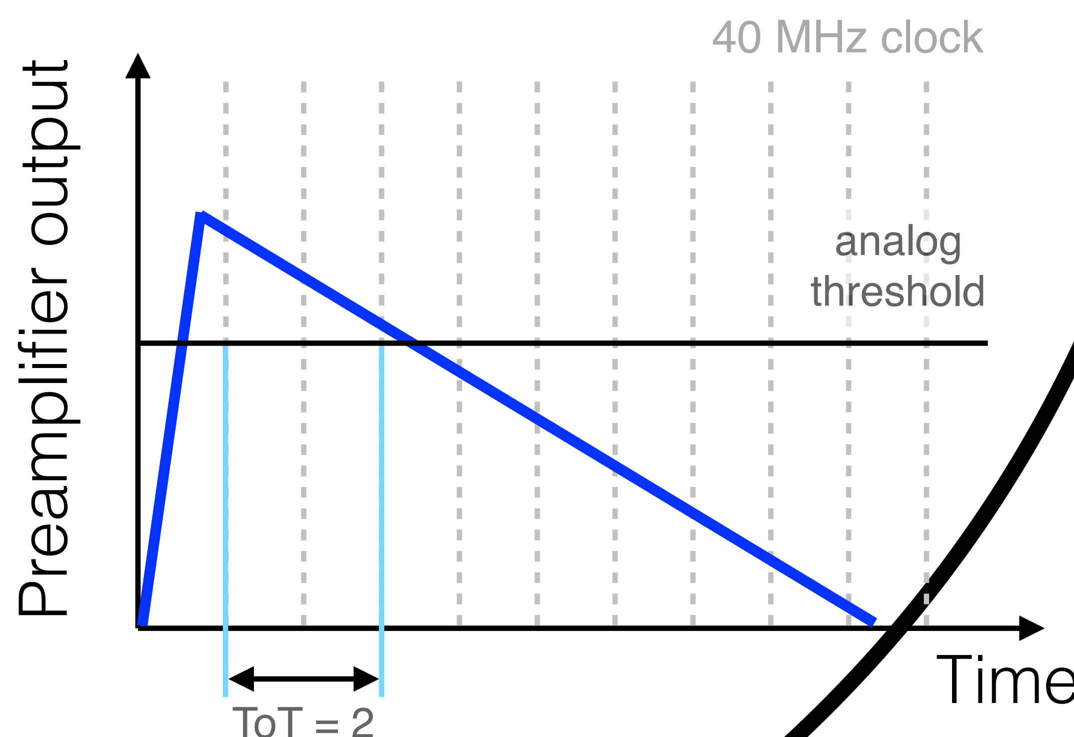


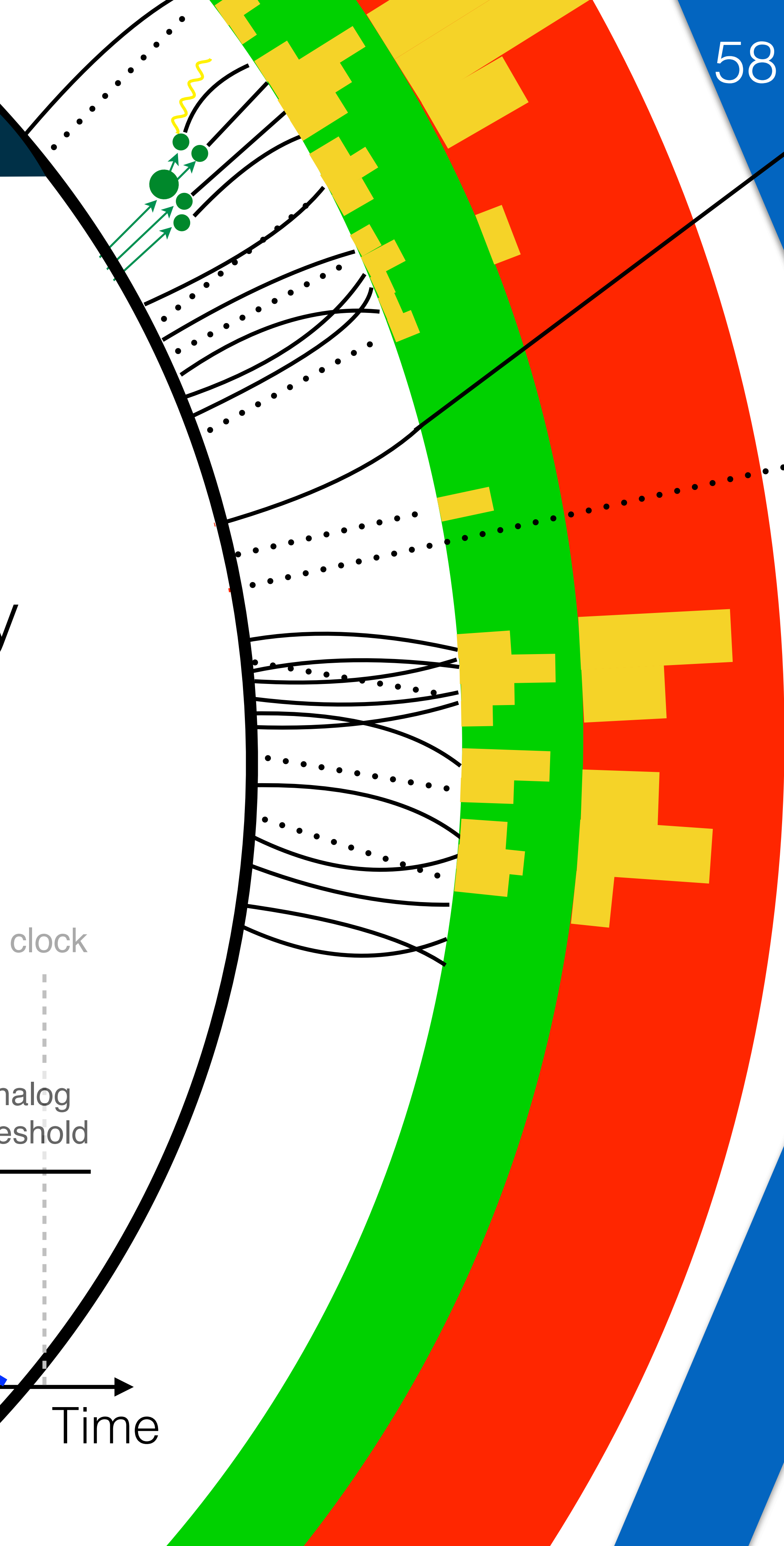
Part IV: Digitization

It is important to mention that **after** Geant4, each experiment has custom code for ***digitization***

this can also be slow; but is usually faster than G4 and reconstruction

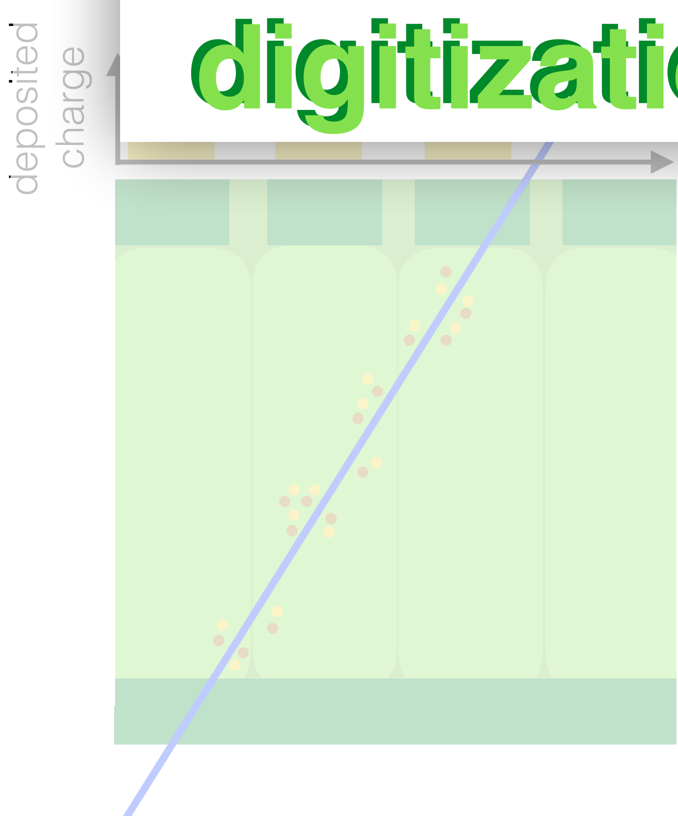


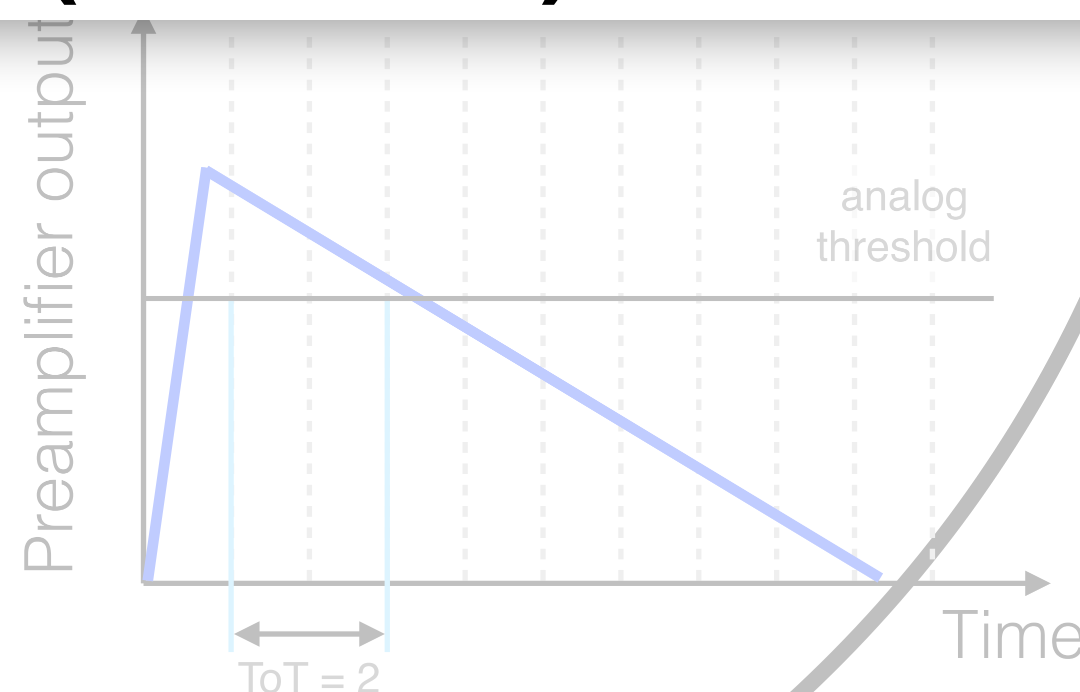


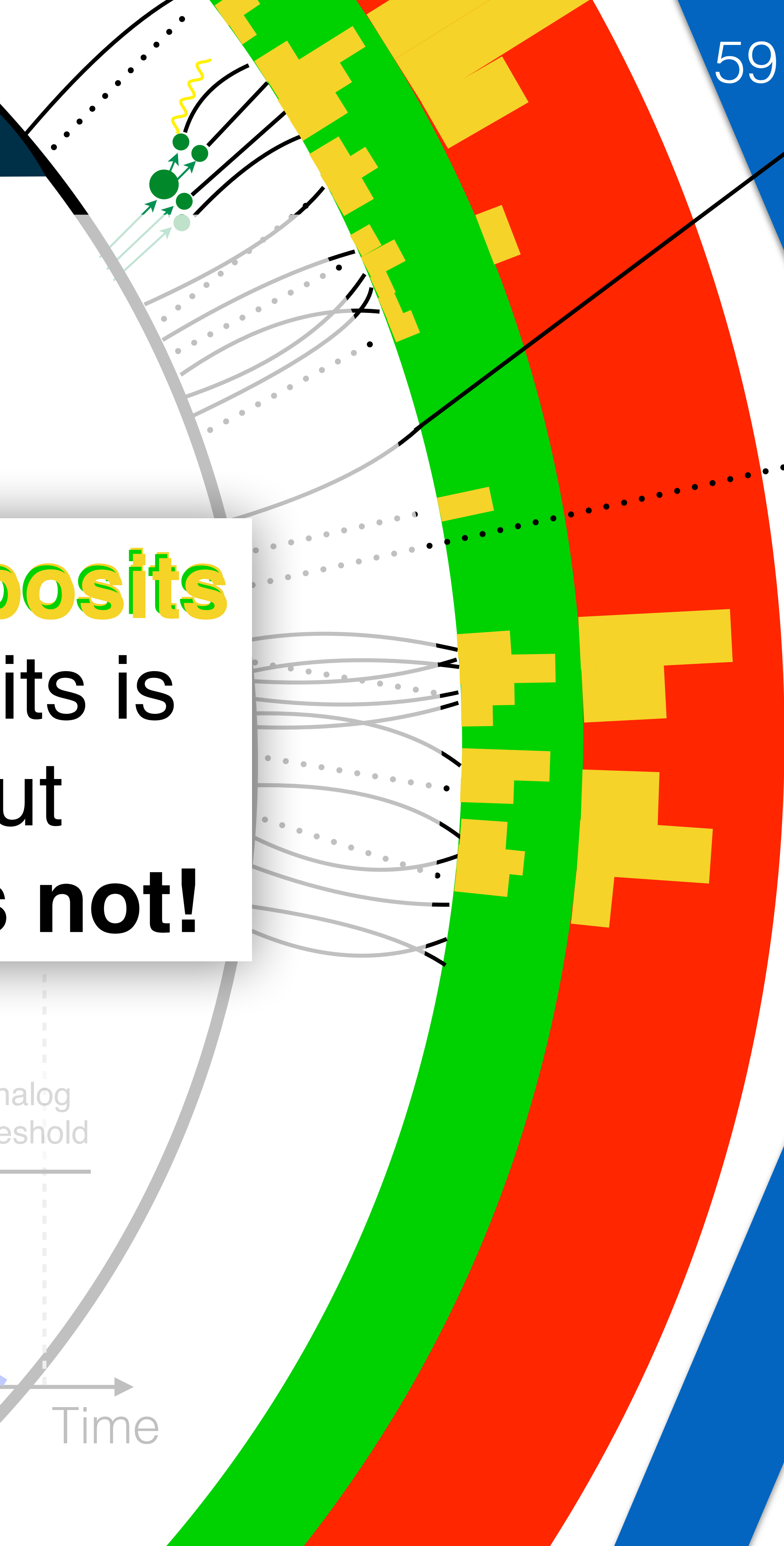


It is important to mention that **after** Geant4, each experiment has custom code for *digitization*

N.B. calorimeter energy deposits **factorize** (sum of the deposits is the deposit of the sum) but **digitization (w/ noise) does not!**

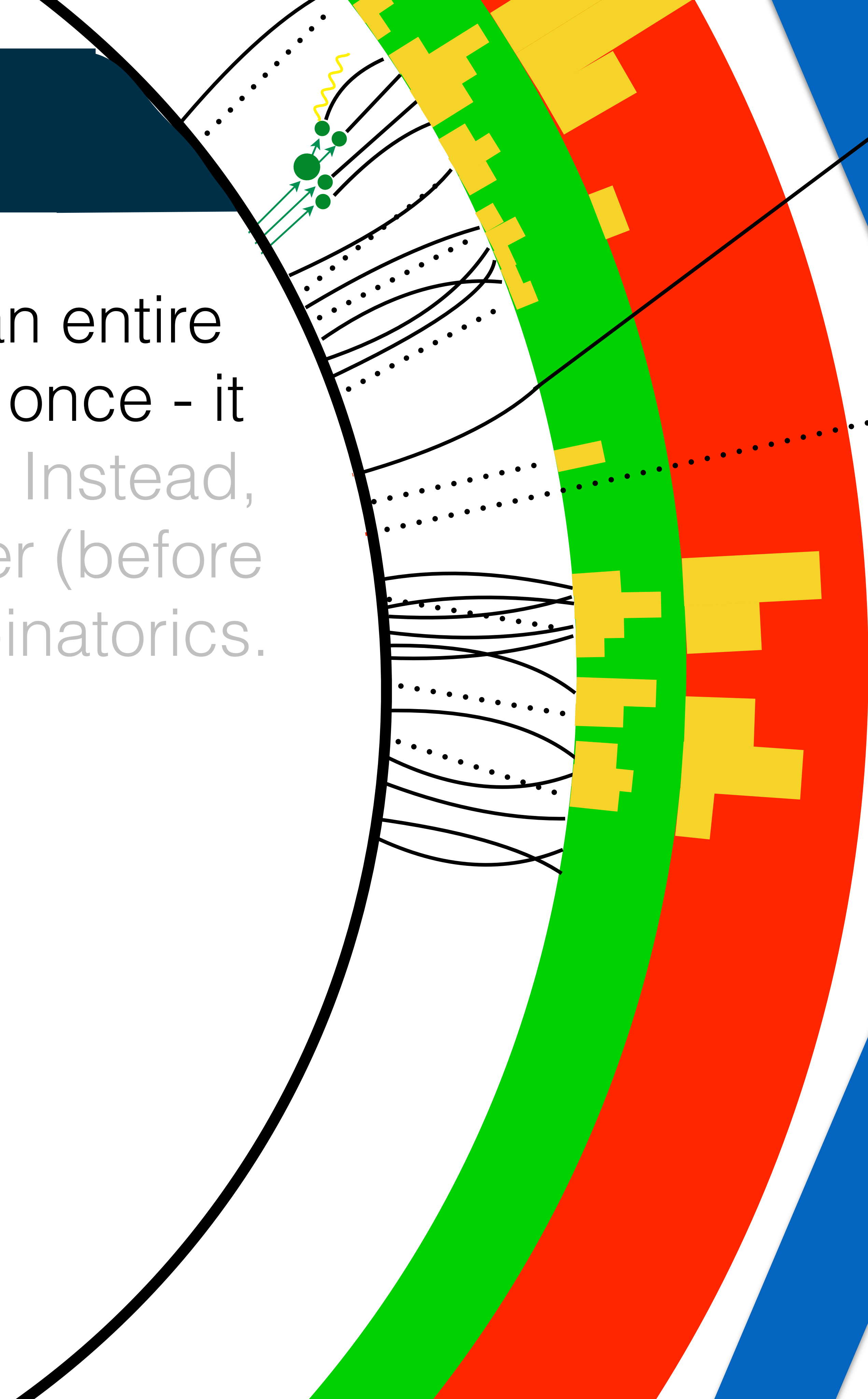






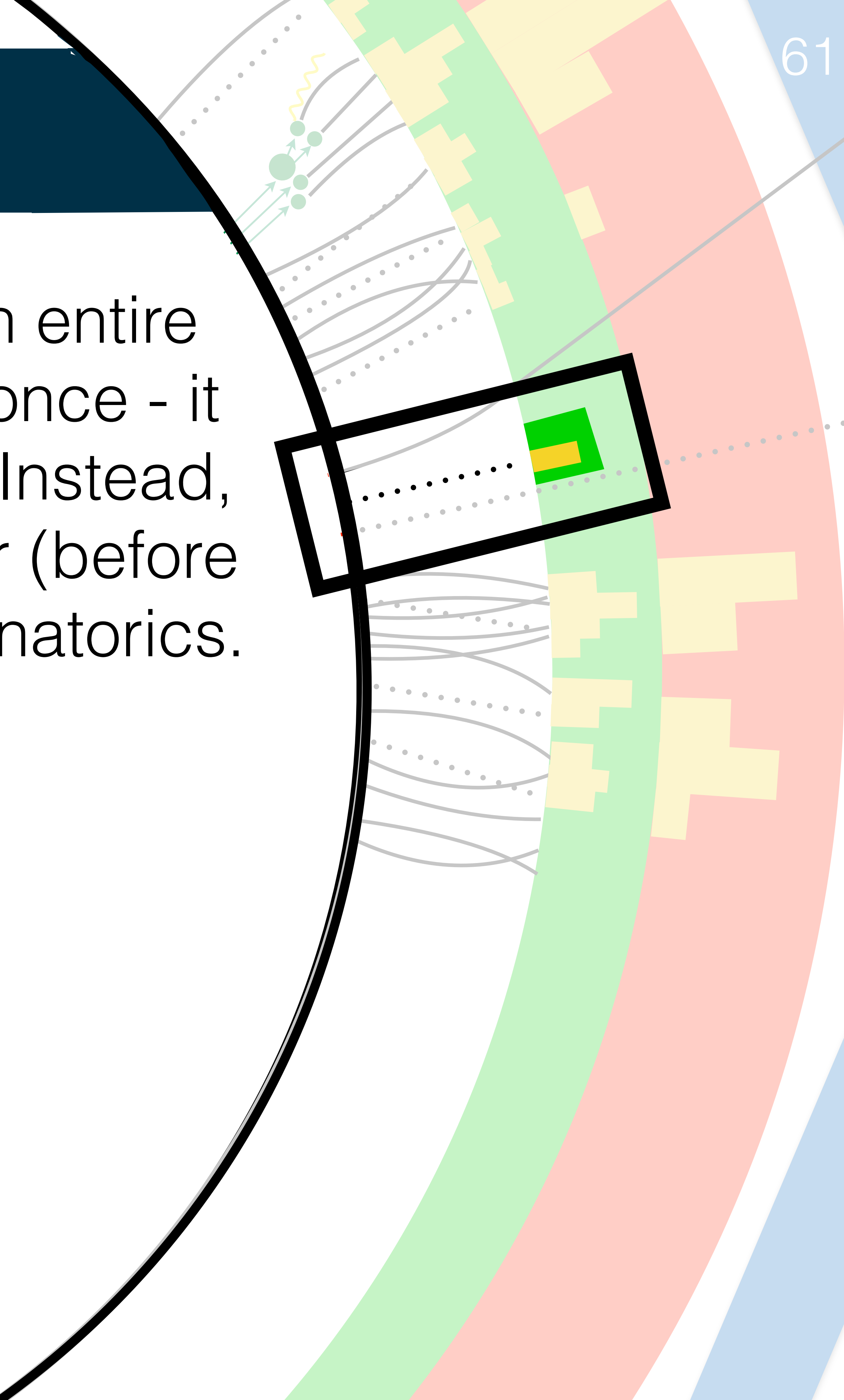
Factorization

We are not trying to generate an entire event (O(1000) particles)) all at once - it would be **very had to validate!** Instead, generate a single particle shower (before electronics) and appeal to combinatorics.



Factorization

We are not trying to generate an entire event (O(1000) particles)) all at once - it would be **very had to validate!** Instead, generate a single particle shower (before electronics) and appeal to combinatorics.

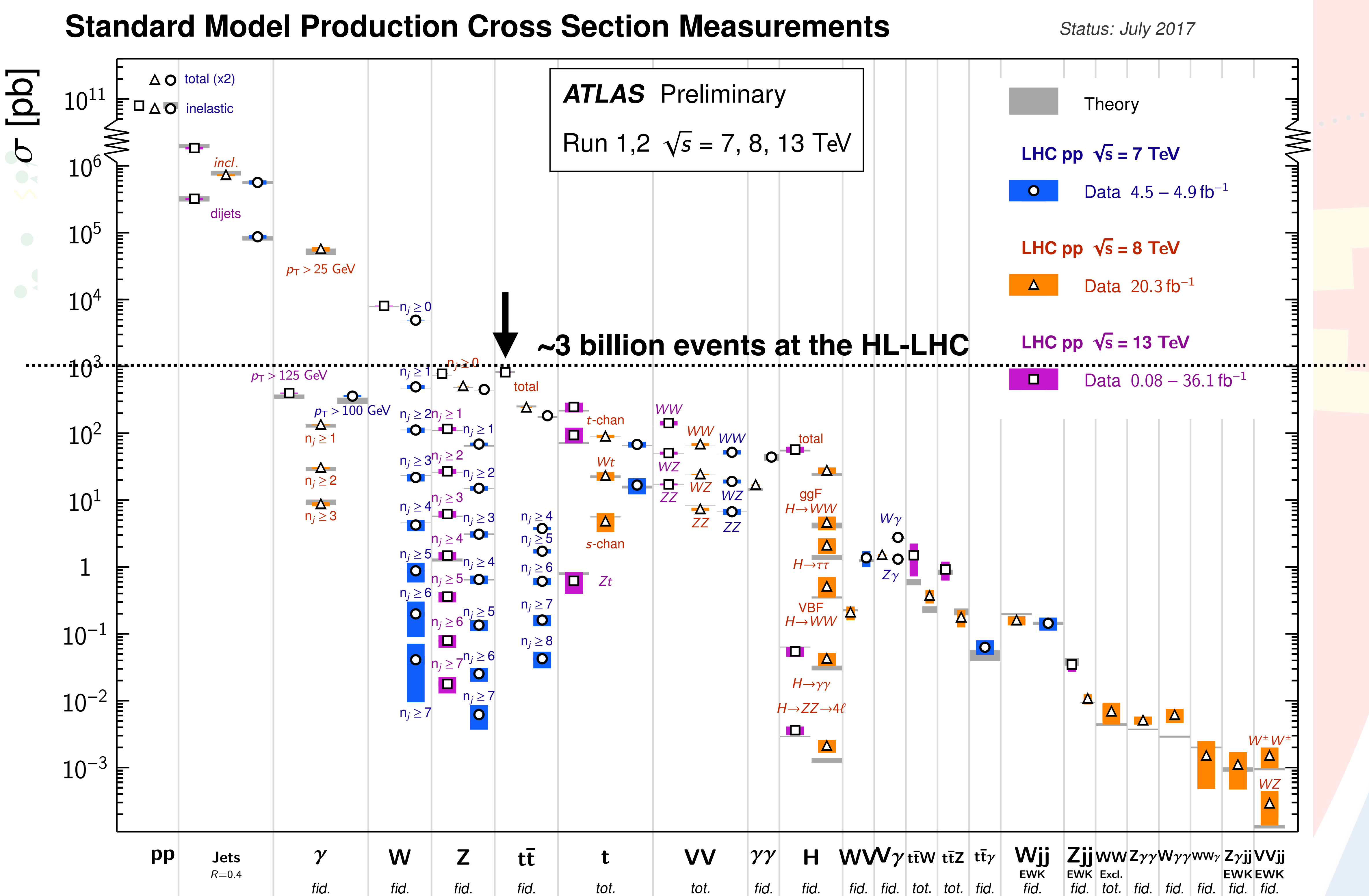


Goal: replace (or augment) simulation steps with a faster, powerful generator based on state-of-the-art machine learning techniques

This work: attack the most important part:
Calorimeter Simulation

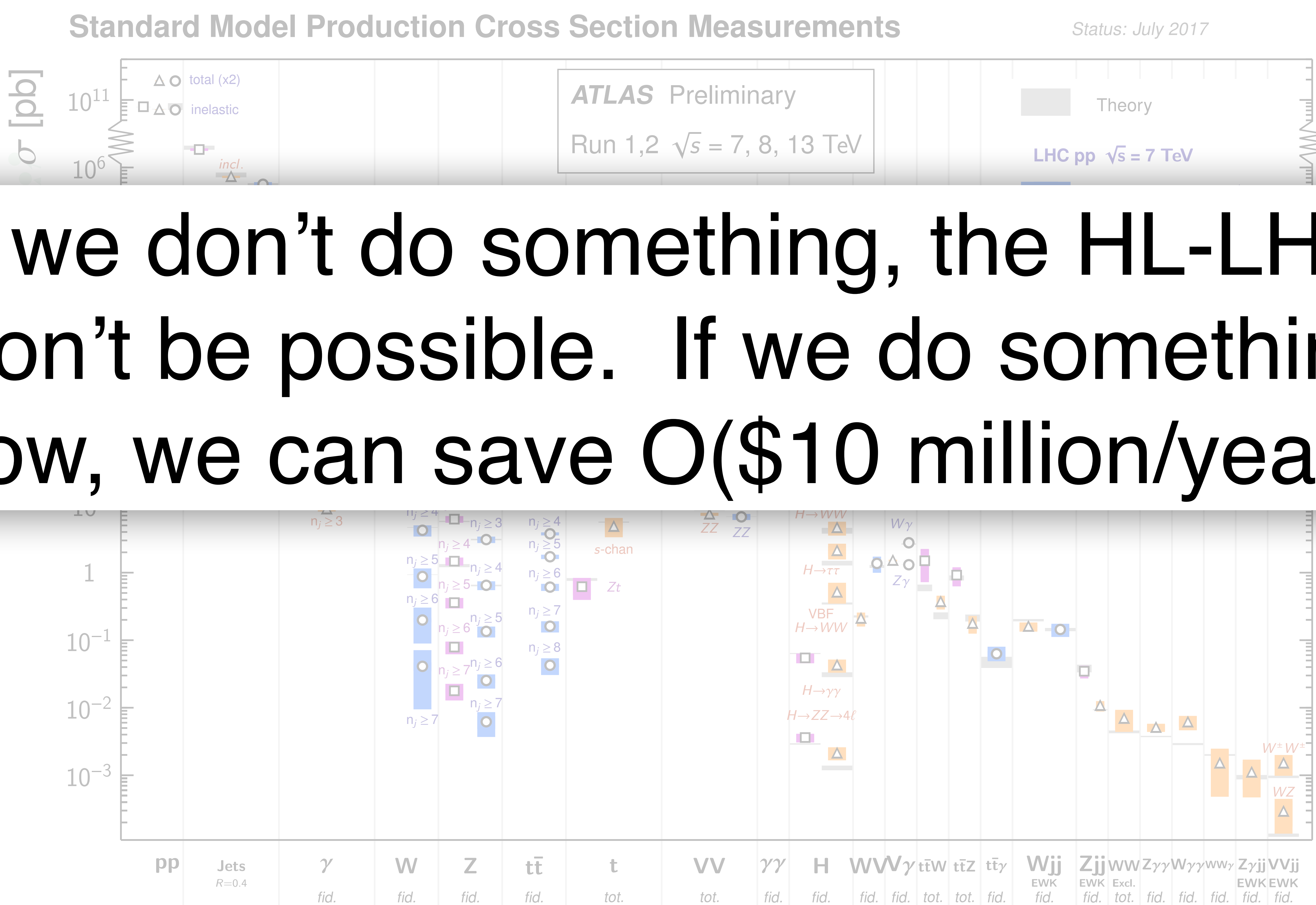
Why should **you** care?

N.B. ALL jet substructure analyses in ATLAS are forced to use full simulation as current fast sim. is not good enough.



Why should **you** care?

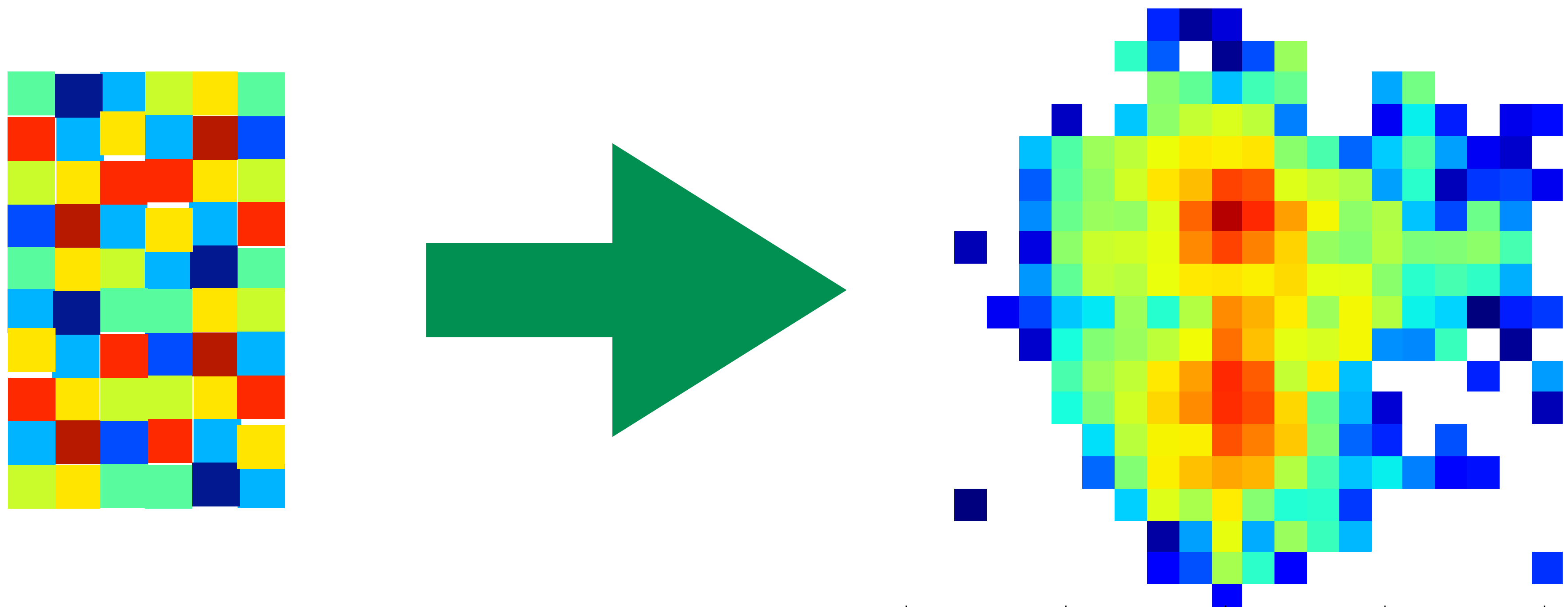
N.B. ALL jet substructure analyses in ATLAS are forced to use full simulation as current fast sim. is not good enough.



If we don't do something, the HL-LHC won't be possible. If we do something now, we can save O($10 million/year).

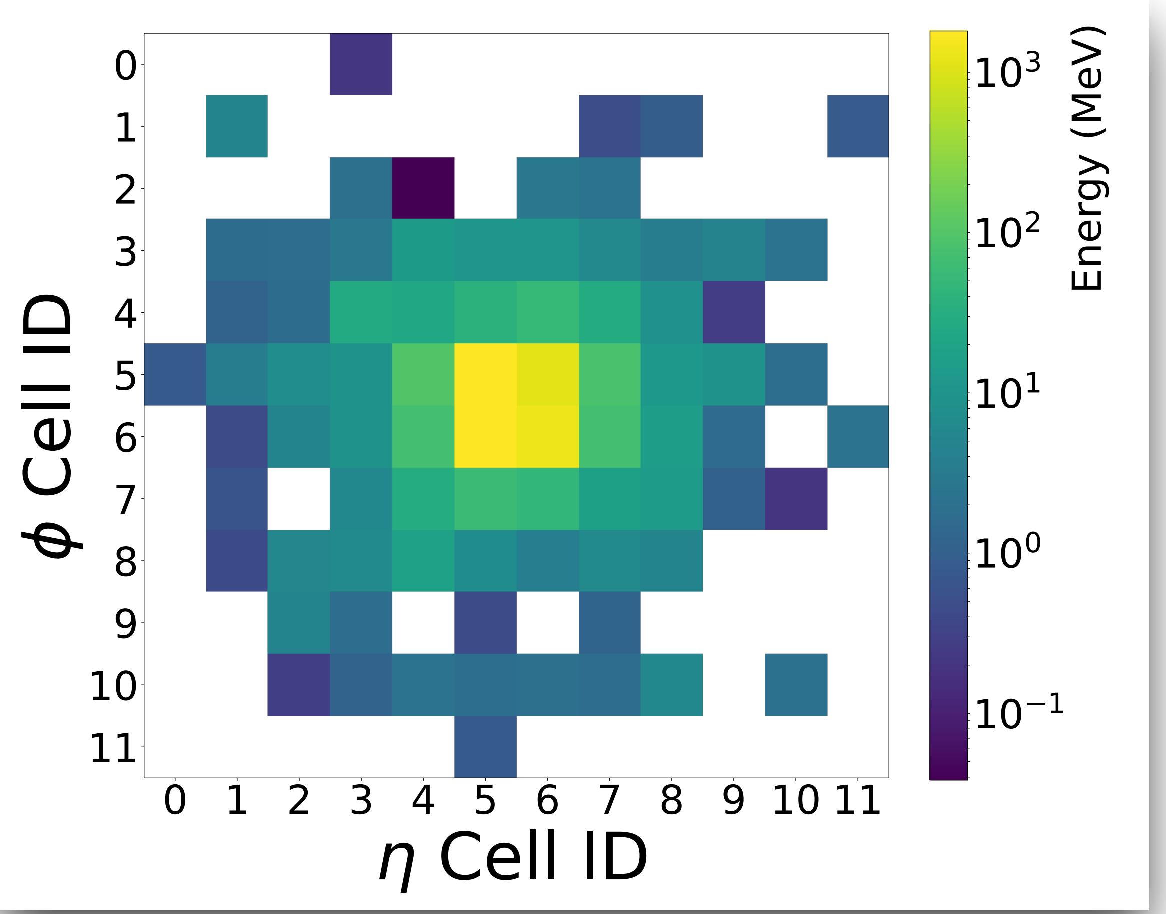
Now to the machine learning

A **generator** is nothing other than a function that maps random numbers to structure.

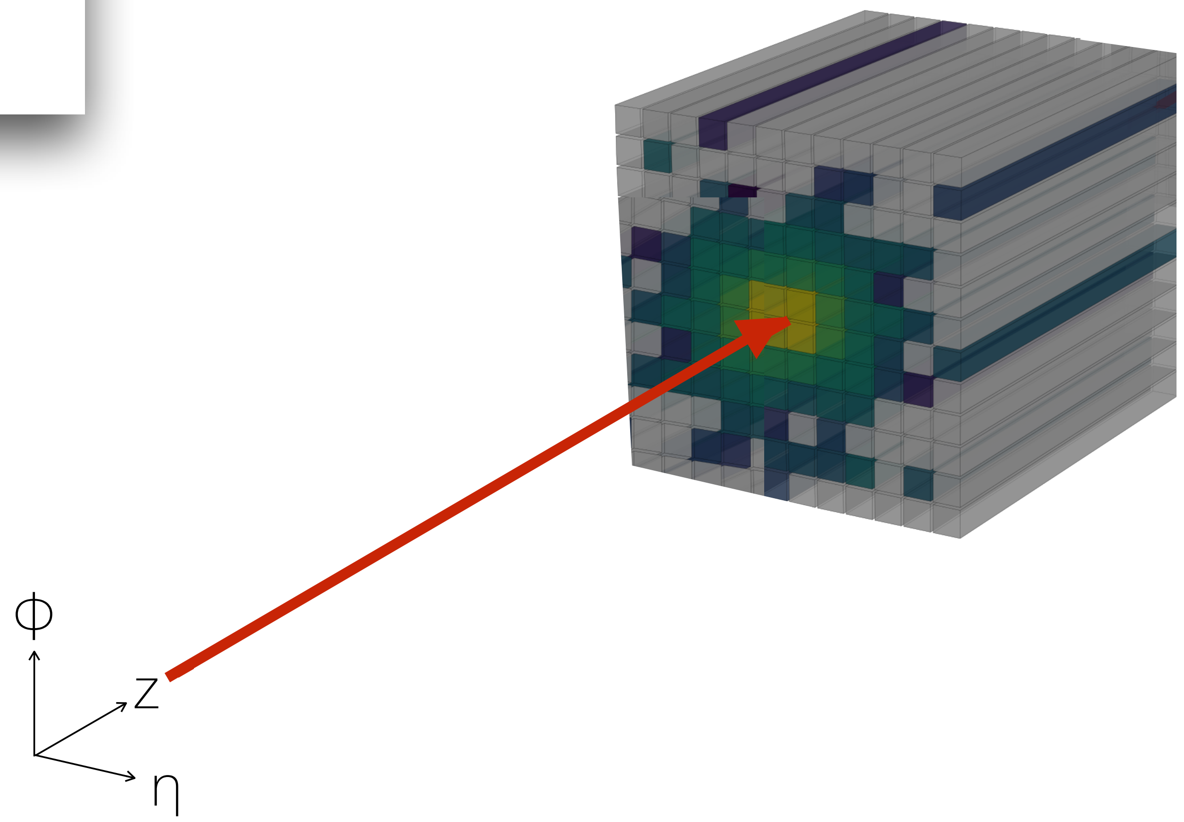


Our structure: **calorimeter images**

Calorimeter images



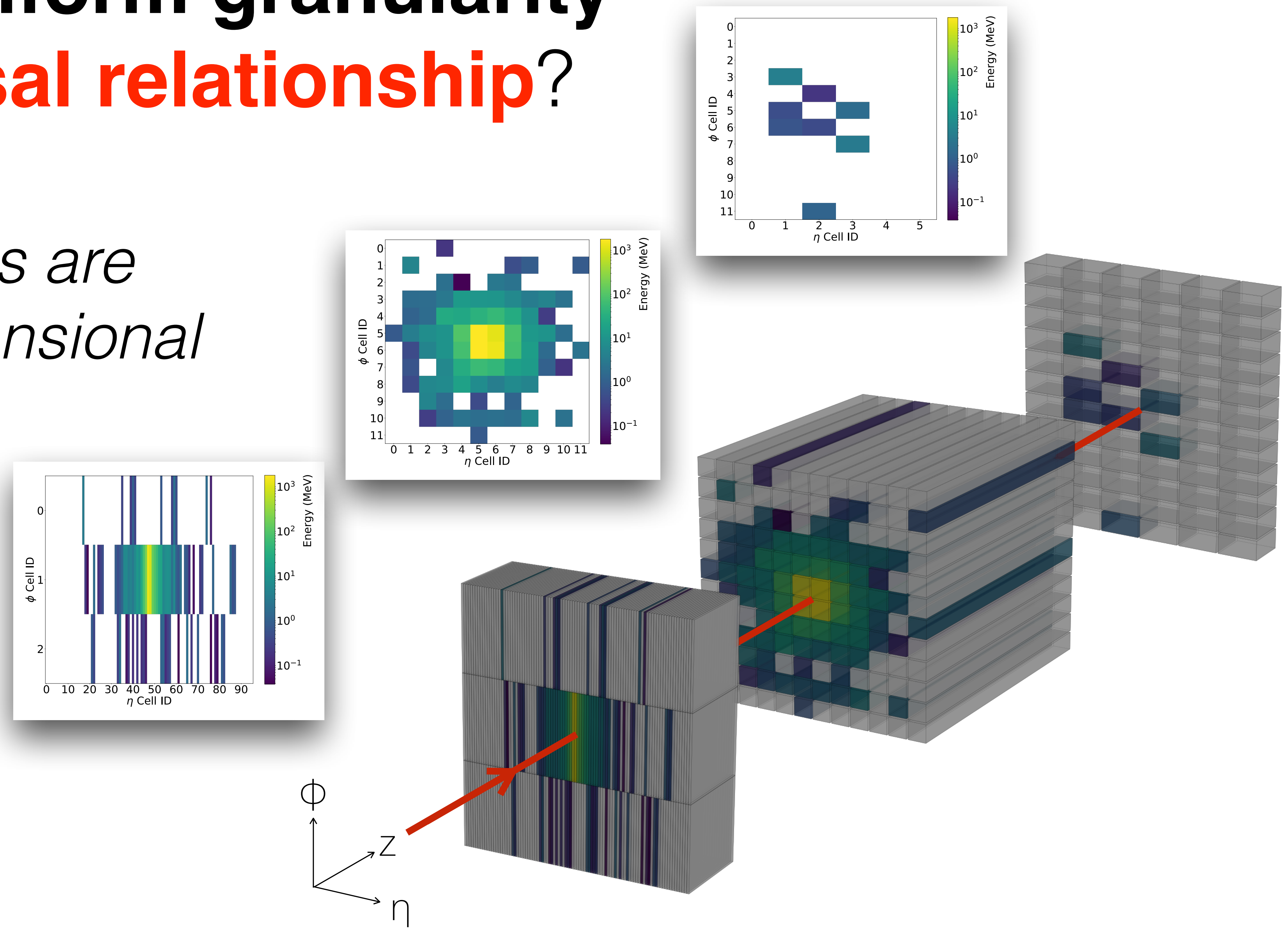
Grayscale images:
Pixel intensity =
energy deposited



Calorimeter images

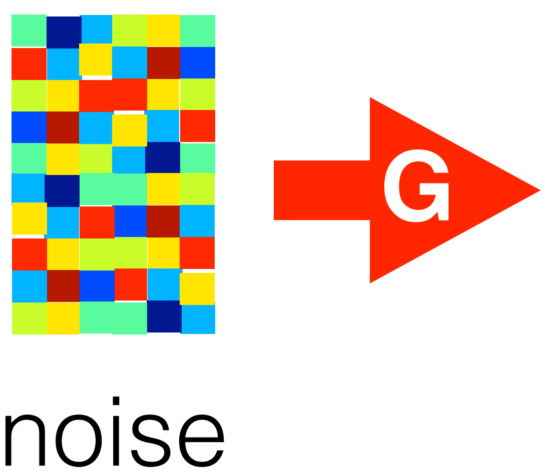
Challenge: **multiple layers** with **non-uniform granularity** and a **causal relationship**?

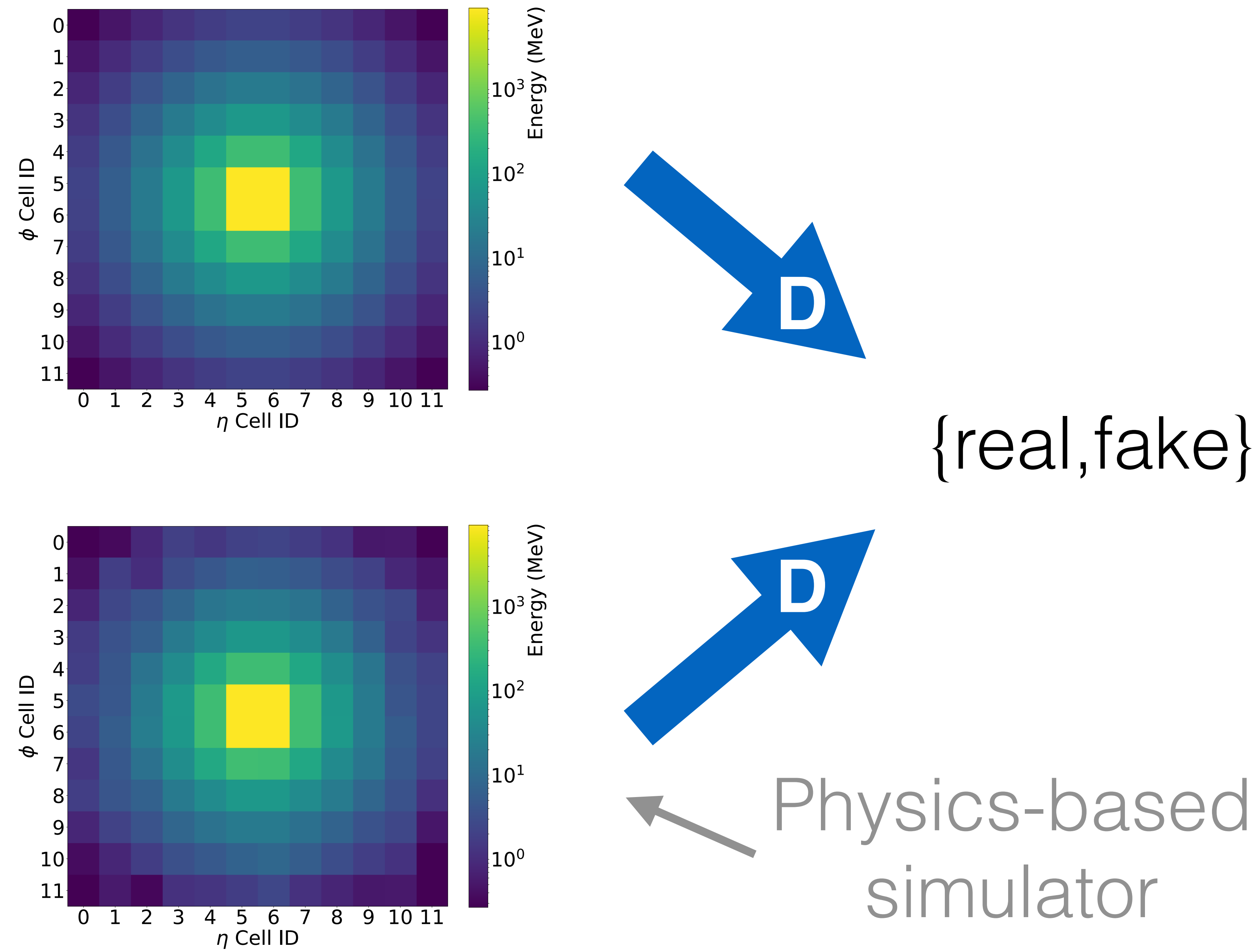
N.B. images are O(1000) dimensional



One popular approach: GANs

Generative Adversarial Networks (GAN):
*A two-network game where one **maps noise to images** and one **classifies images as fake or real**.*





When **D** is maximally confused, **G** will be a good generator

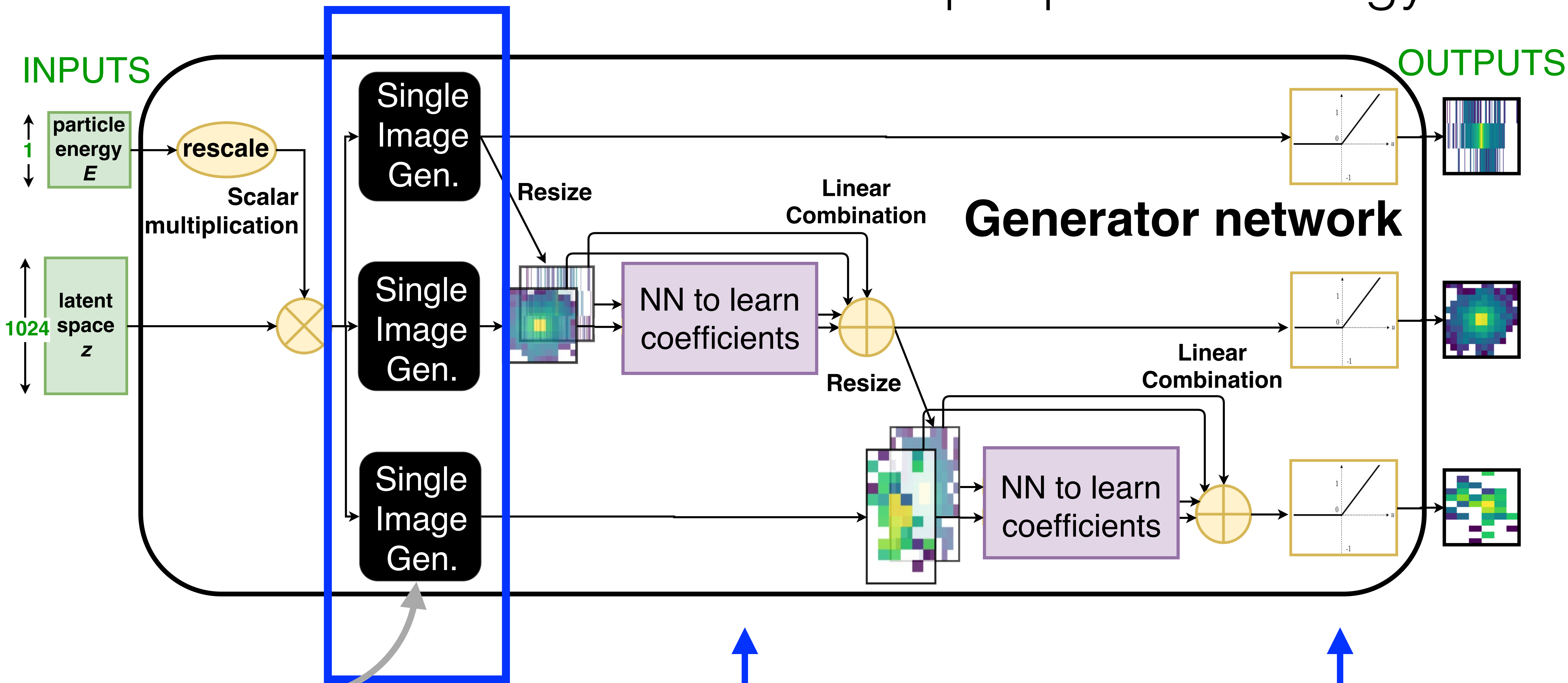
[I. Goodfellow et al., NIPS 2014]

Introducing CaloGAN

[L. de Oliveira, M. Paganini, BPN, PRL 120 (2018) 042003]

One network per particle type; input particle energy

One image per calo. layer



[L. de Oliveira, M. Paganini, BPN, CSBS 1 (2017) 4]

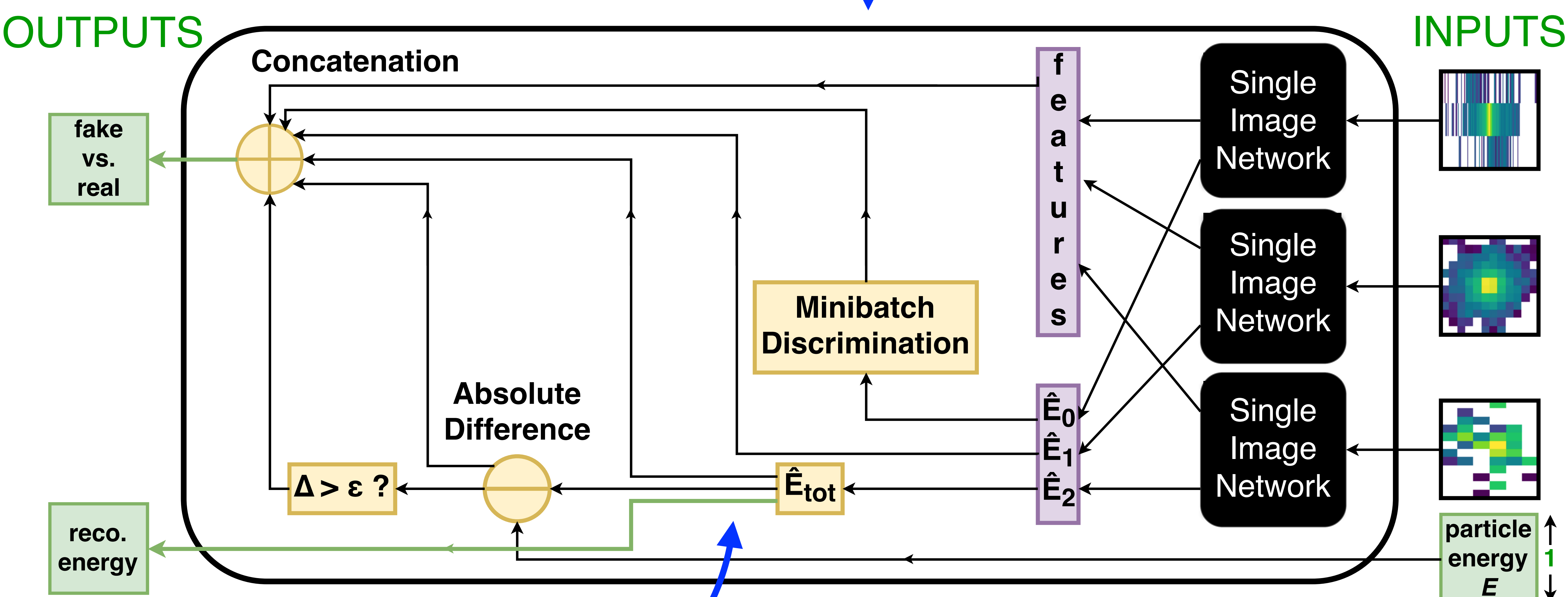
use layer i as input to layer i+1

ReLU to encourage sparsity

Building in physical constraints

Mode collapse: learns to generate one part of the distribution well, but leaves out other parts.

help avoid 'mode collapse'

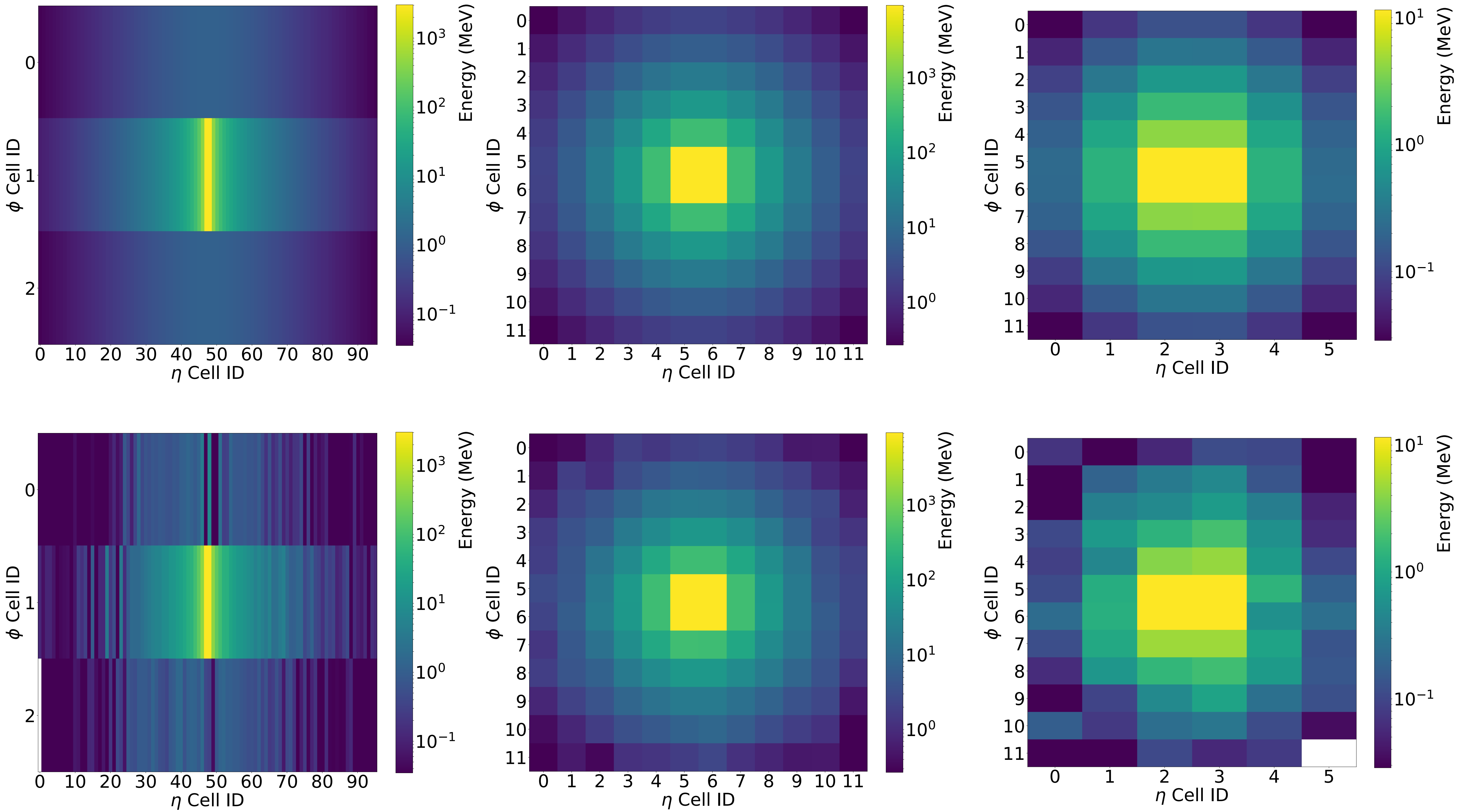


Encourage energy conservation

Discriminator network

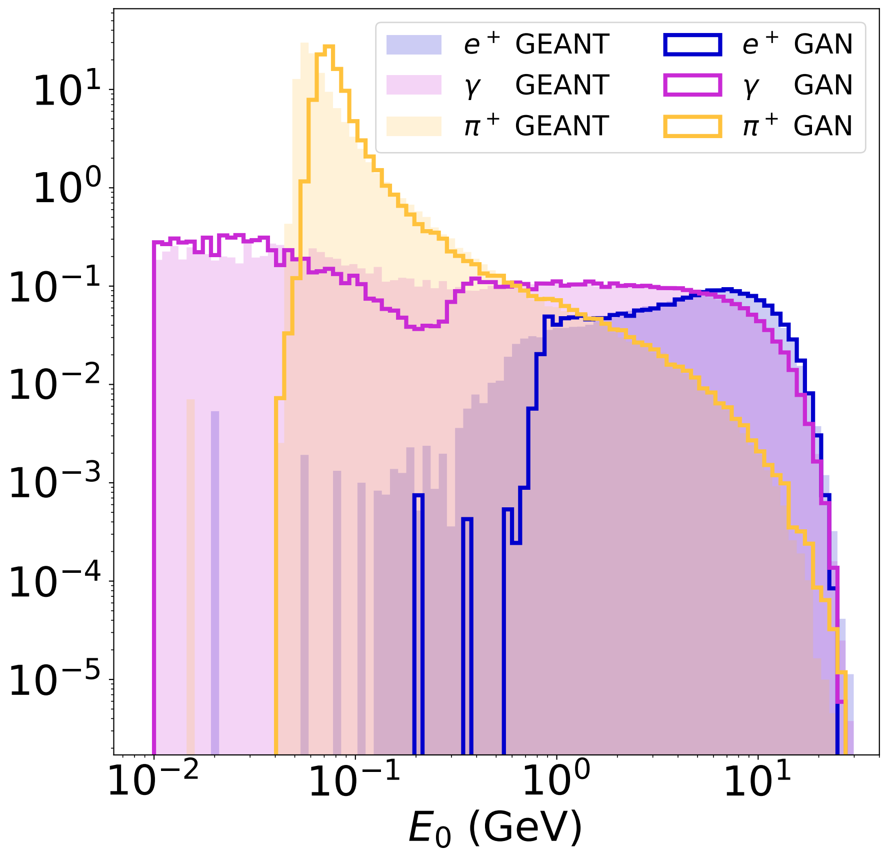
Results: average images

Full physics generator (Geant4)

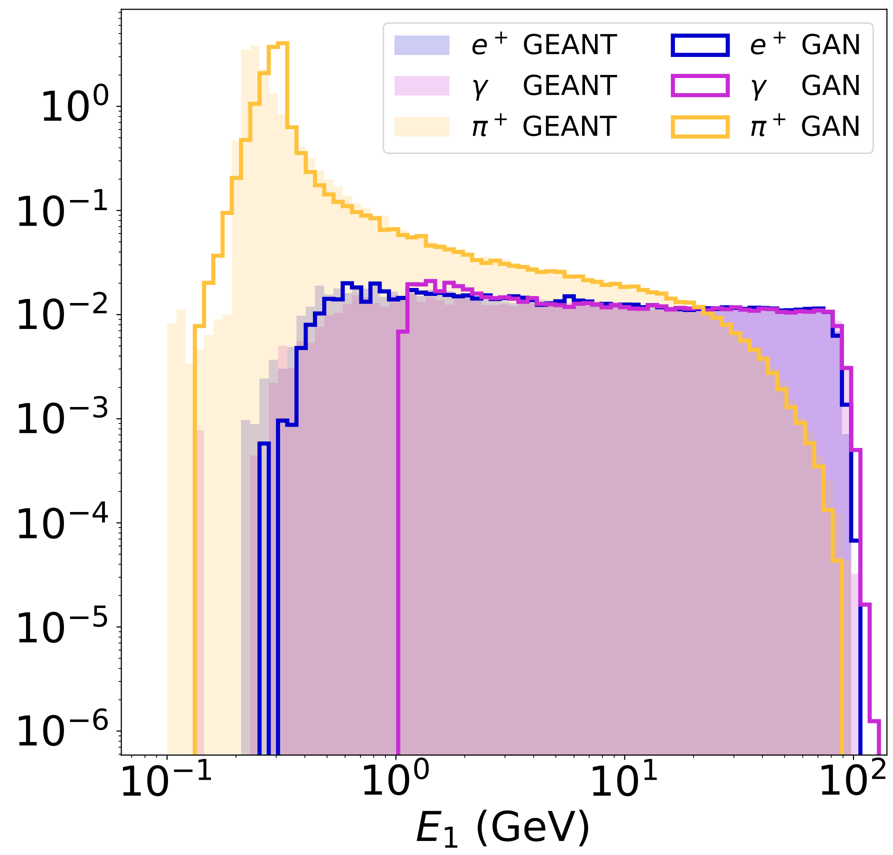


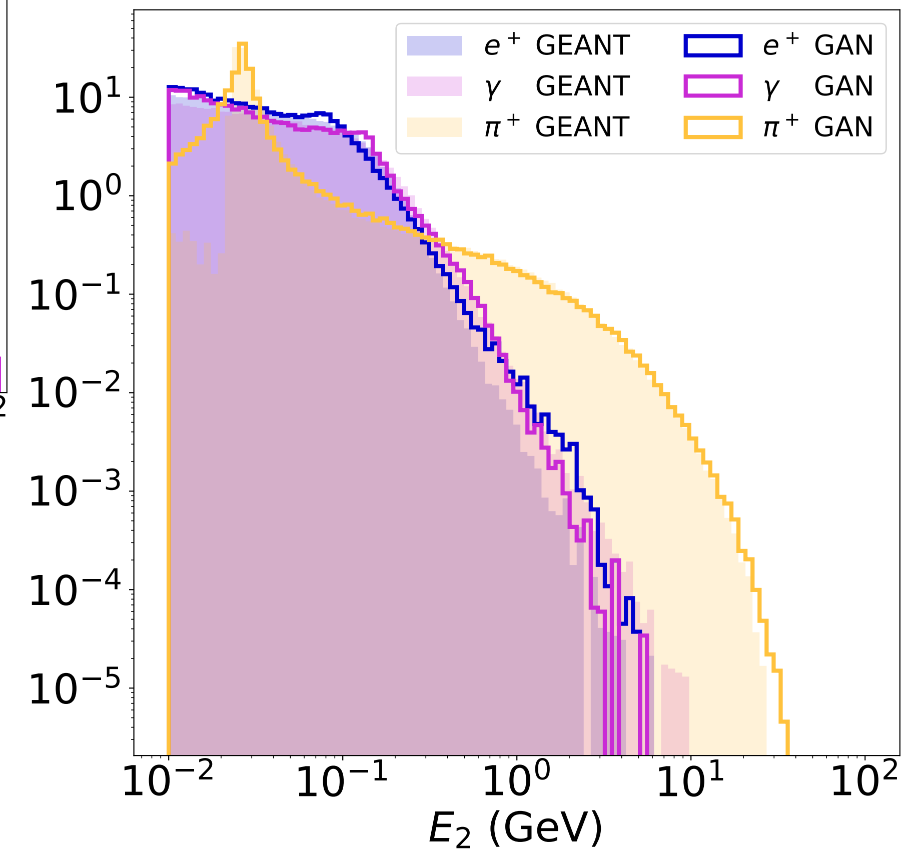
CaloGAN

Energy per layer



Pions deposit much less energy in the first layers; leave the calorimeter with significant energy



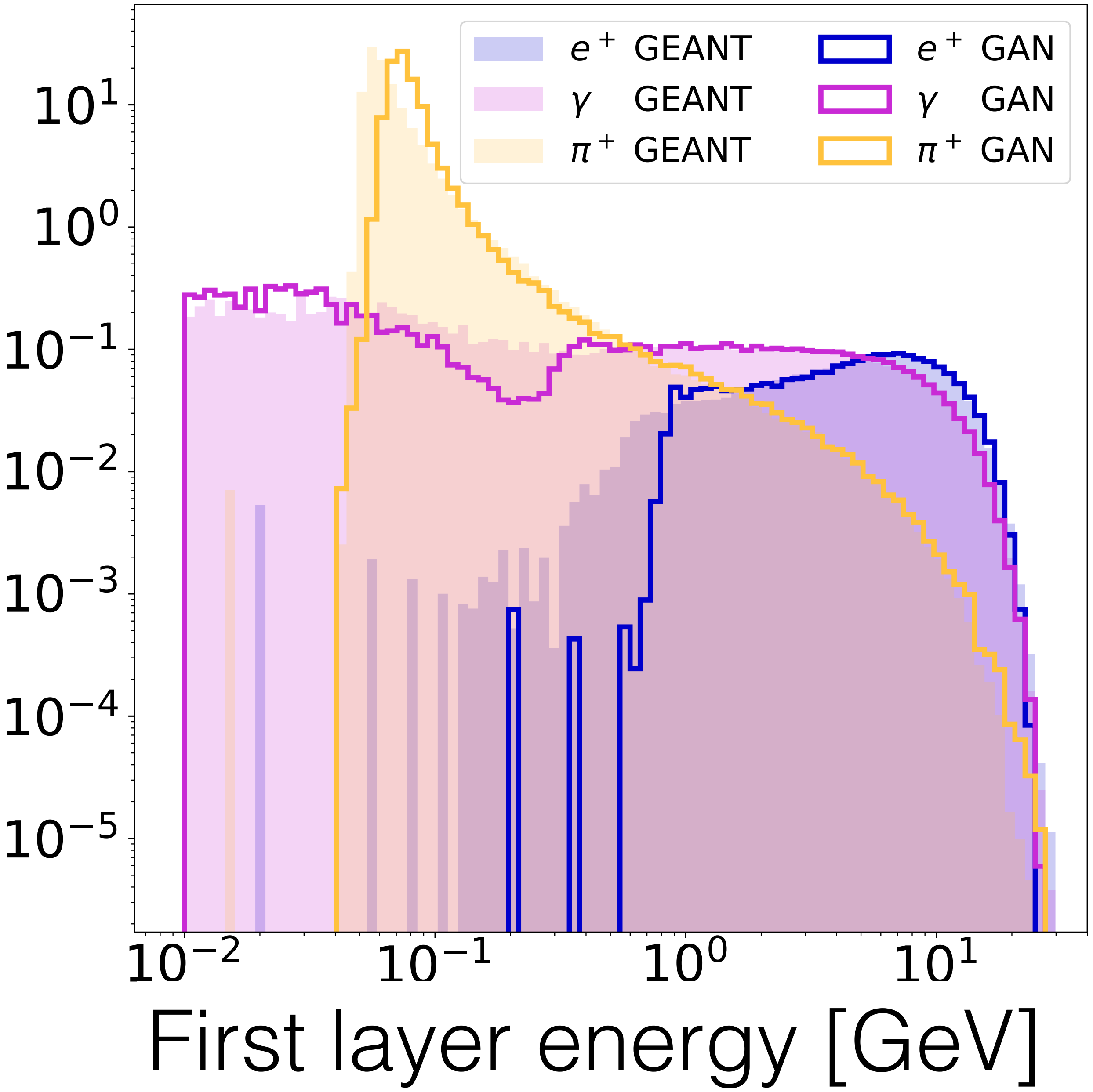


N.B. can always add these (and others) explicitly to the training

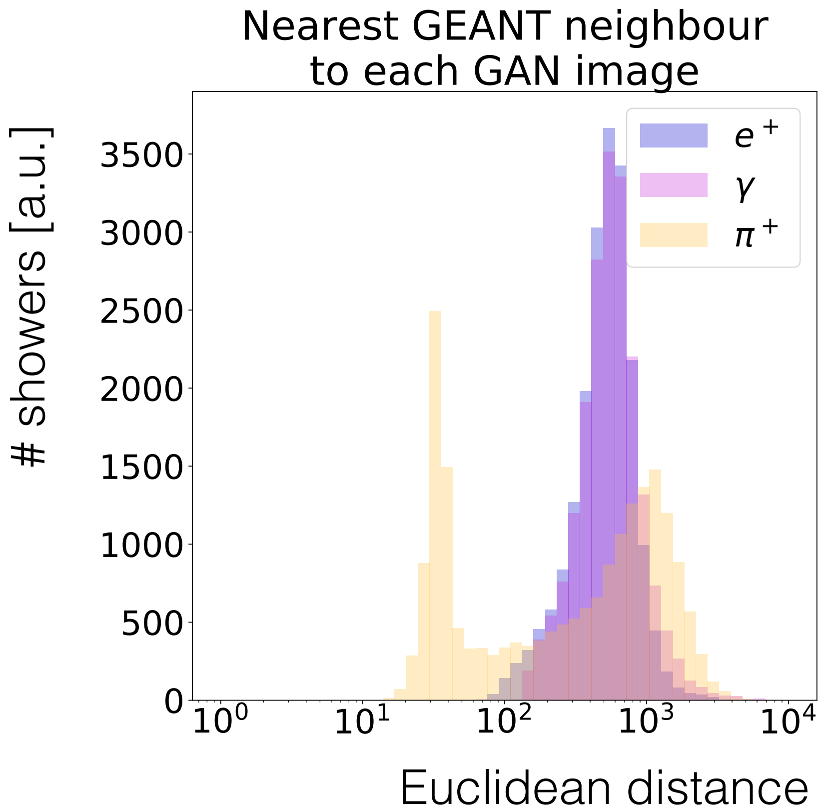
Warning: challenge with GANs

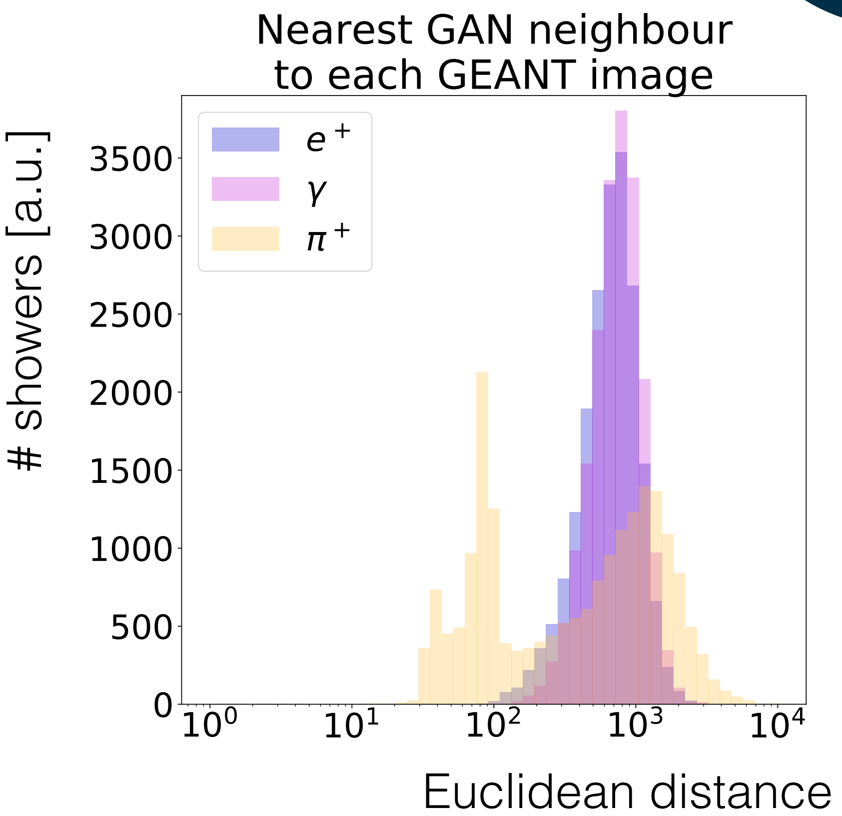
Unlike for classifiers, it is not easy to figure out which GAN is a good GAN - trying to learn a O(1000) generative model and not a single likelihood ratio!

…this is a place where science applications can make a big impact on ML.

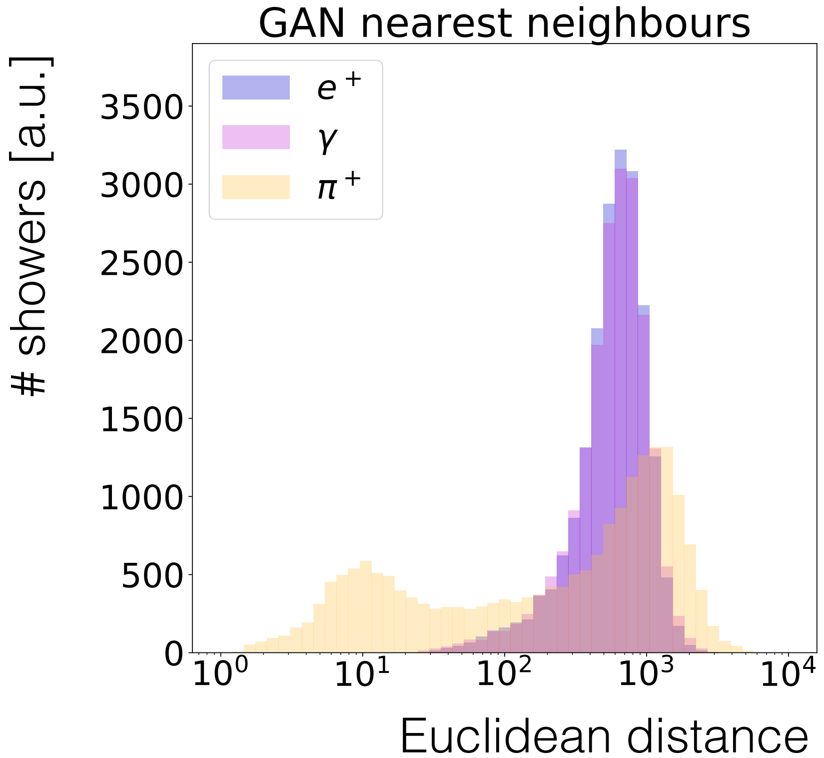


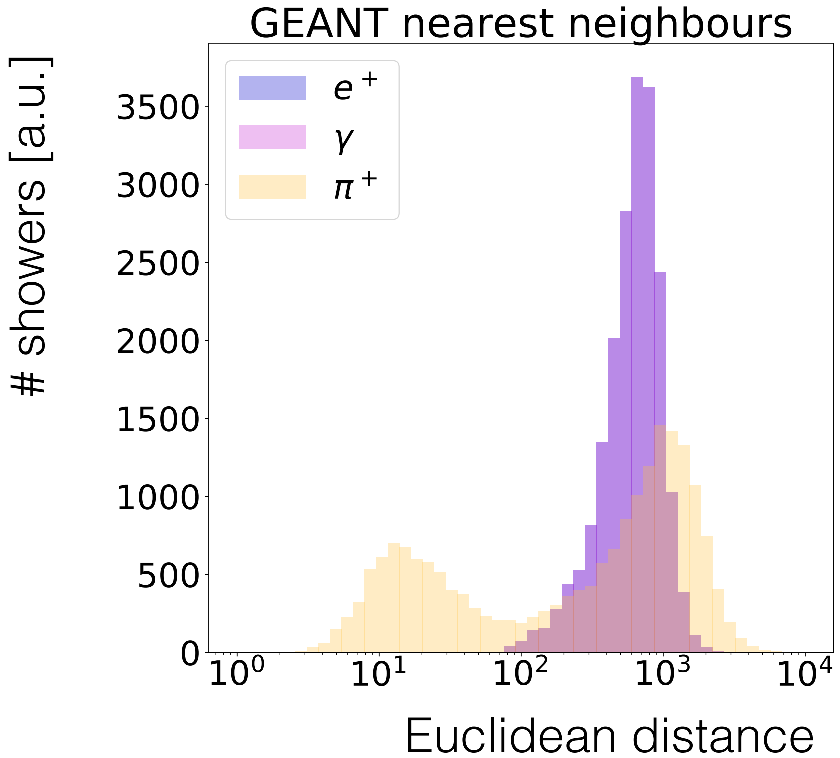
One look at “overtraining”





not memorizing ⟷





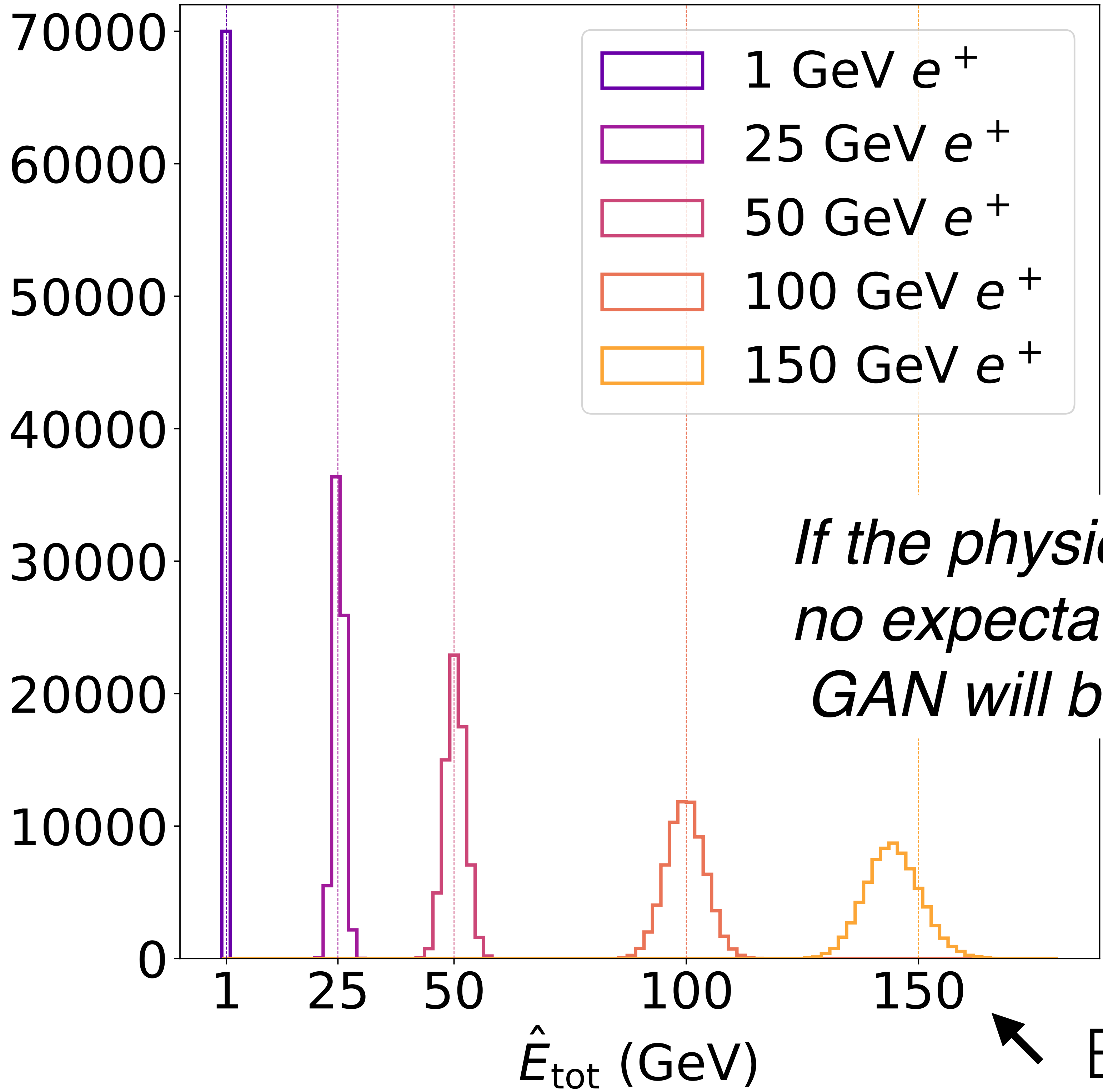
no “mode collapse” ⟷

Timing

Generation Method	Hardware	Batch Size	milliseconds/shower
GEANT4	CPU	N/A	1772 ←
CALOGAN	CPU *Intel Xeon E5-2670*	1	13.1
		10	5.11
		128	2.19
		1024	2.03
	GPU *NVIDIA K80*	1	14.5
		4	3.68
		128	0.021
		512	0.014
		1024	0.012 ←

(clearly these numbers will change as both technologies improve - this is simply meant to be qualitative and motivating!)

Extrapolating



If the physics changes, no expectation that the GAN will be accurate.

Beyond our training sample!

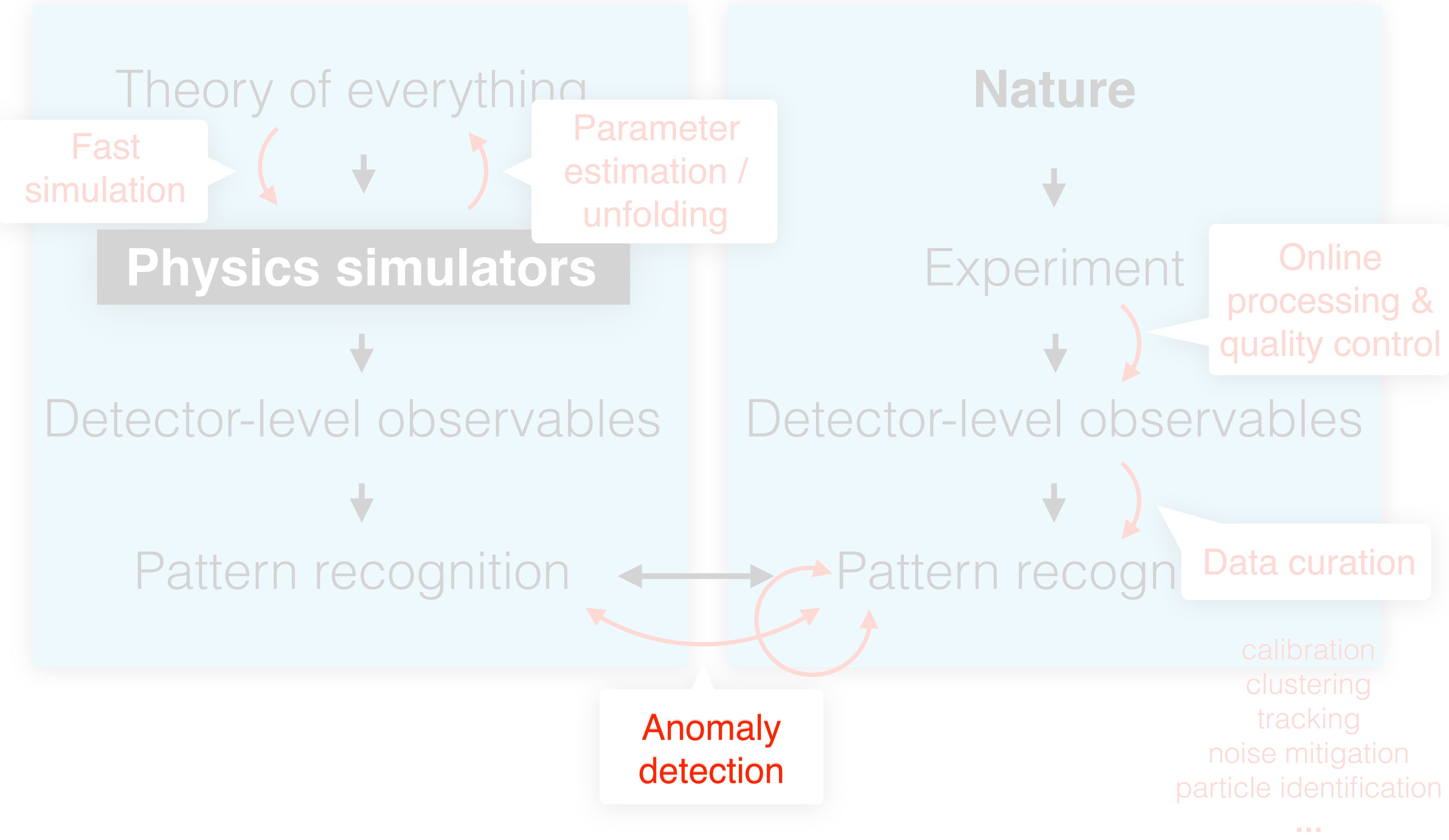
Partial Conclusions for Generation

Generative models are becoming more powerful & popular
(not just GANs, but other models like Variational Autoencoders and Normalizing Flows)

Our applications are often more challenging than industry because our data are less “structured” than natural images and we also have a strong requirement of quantitive and not just qualitative quality (e.g. jets versus celebrities)

…you will hear more about GANs in HEP tomorrow!

Data analysis in HEP + Machine Learning



"But what are the uncertainties on the NN"?

- question asked by every reviewer

Uncertainties & simulation-based inference

"But what are the uncertainties on the NN"?

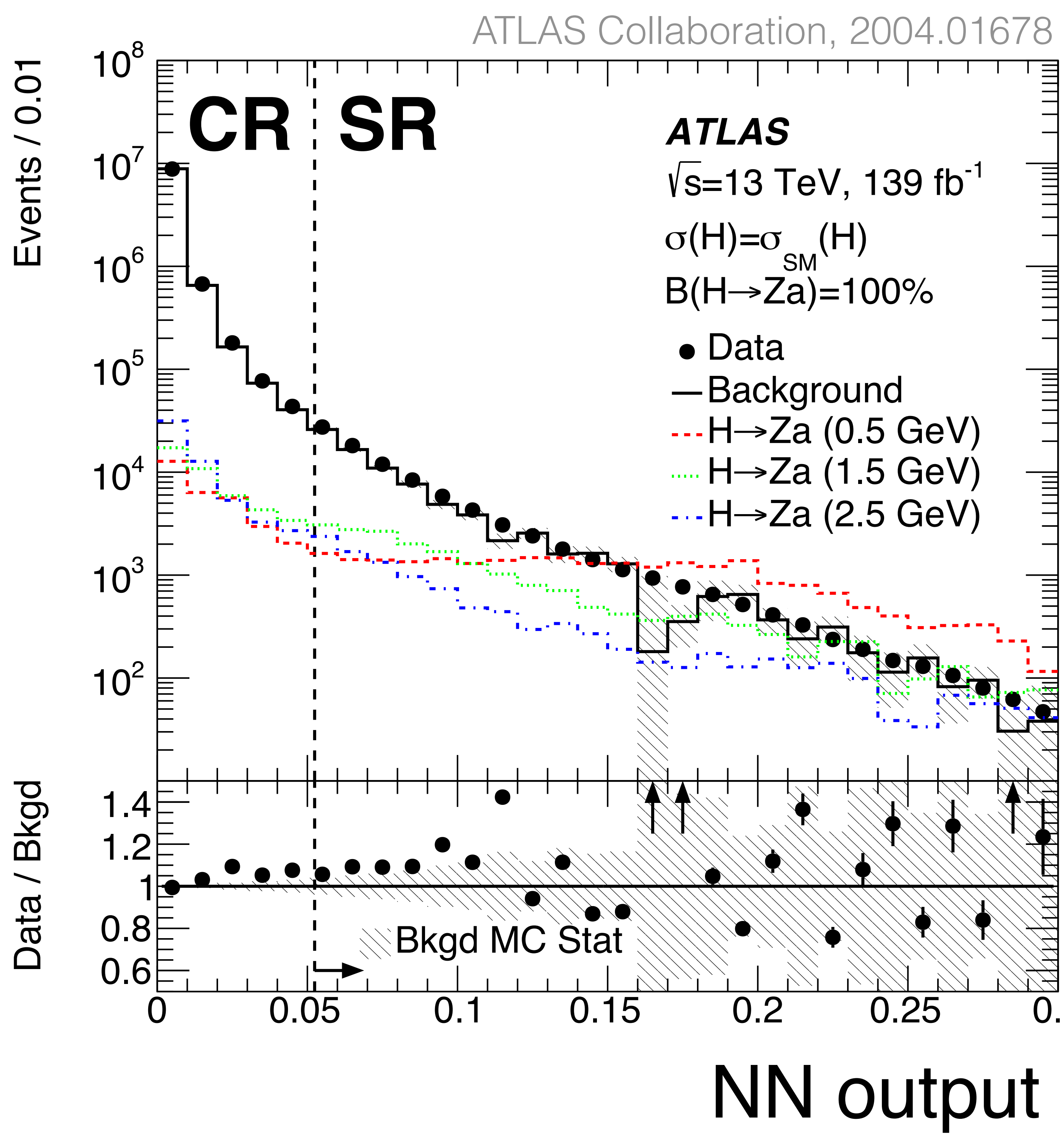
- question asked by every reviewer

Let's consider this question in the context of a search for new particles in collision events.

this is representative for many analyses at the LHC, for example

Setup

1. Train a classifier (in sim.) for signal vs. background.
2. Define a control region (**CR**) and a signal region (**SR**) using (1).
3. Check / modify simulation in CR.
4. Compare data and simulation in SR.



`Significantly different? go to Stockholm : publish limits.`

Uncertainties for a NN-based analysis

Precision / Optimality

*Bad use of our data, time, money, etc. but **not wrong**.*

Accuracy / Bias

Uncertainties for a NN-based analysis

Precision / Optimality: $\mathrm{NN(x)} \neq \frac{p_{\mathrm{true}}(x|\mathrm{S+B})}{p_{\mathrm{true}}(x|\mathrm{B})}$

↑ Optimal by Neyman-Pearson

Accuracy / Bias

Note that this is not p(x|S) / p(x|B), however the two are monotonically related to each other.

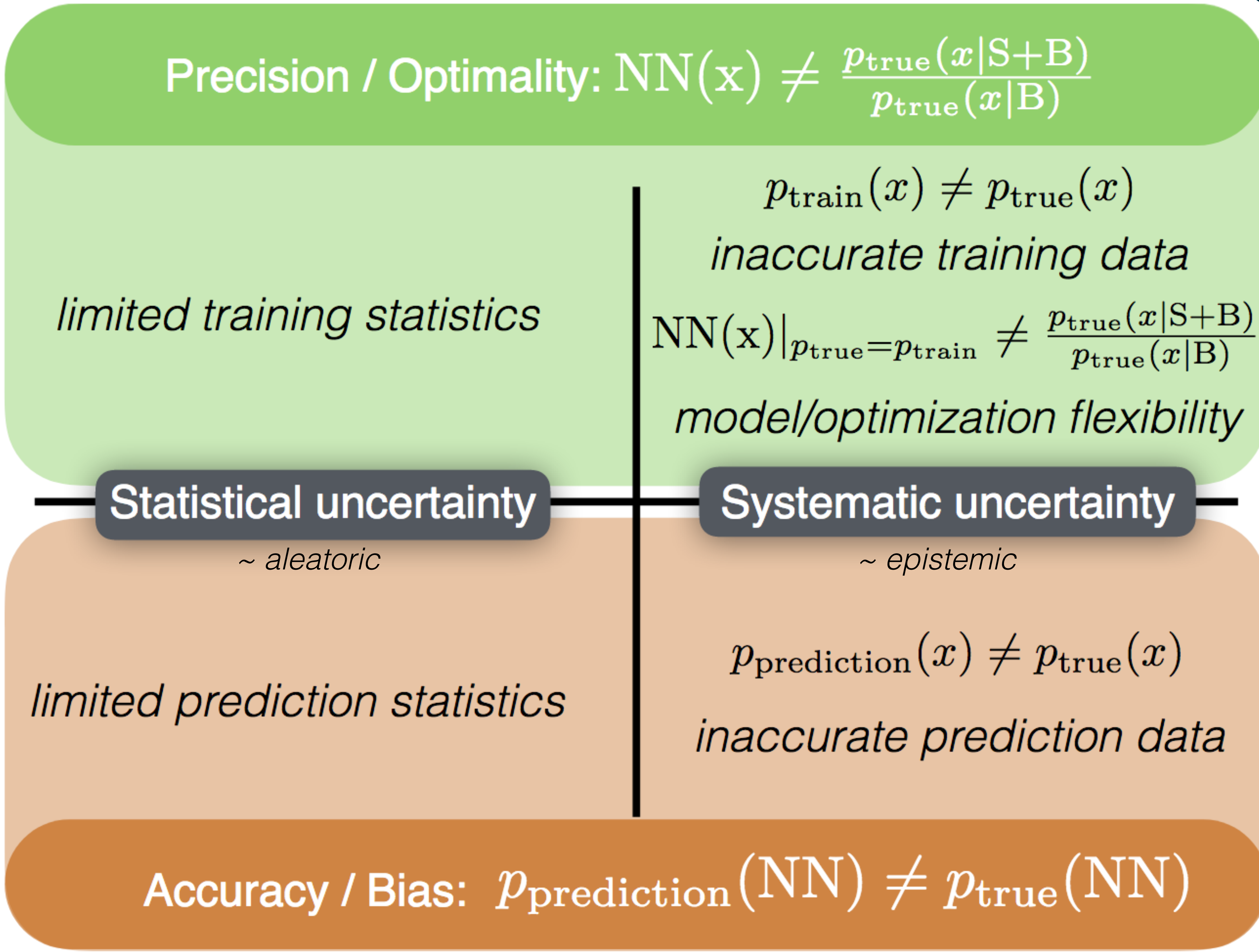
Uncertainties for a NN-based analysis

Precision / Optimality: $\mathrm{NN(x)} \neq \frac{p_{\mathrm{true}}(x|\mathrm{S+B})}{p_{\mathrm{true}}(x|\mathrm{B})}$

Accuracy / Bias: $p_{\mathrm{prediction}}(\mathrm{NN}) \neq p_{\mathrm{true}}(\mathrm{NN})$

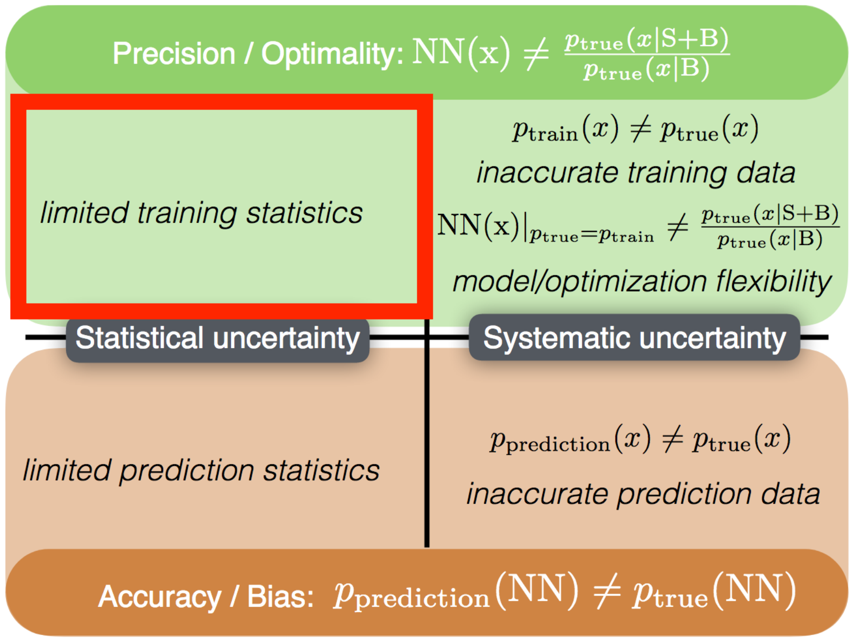
The distribution of the (corrected) sim. is not correct.

Uncertainties for a NN-based analysis



BPN, 1909.03081

How to estimate precision stat. uncerts.



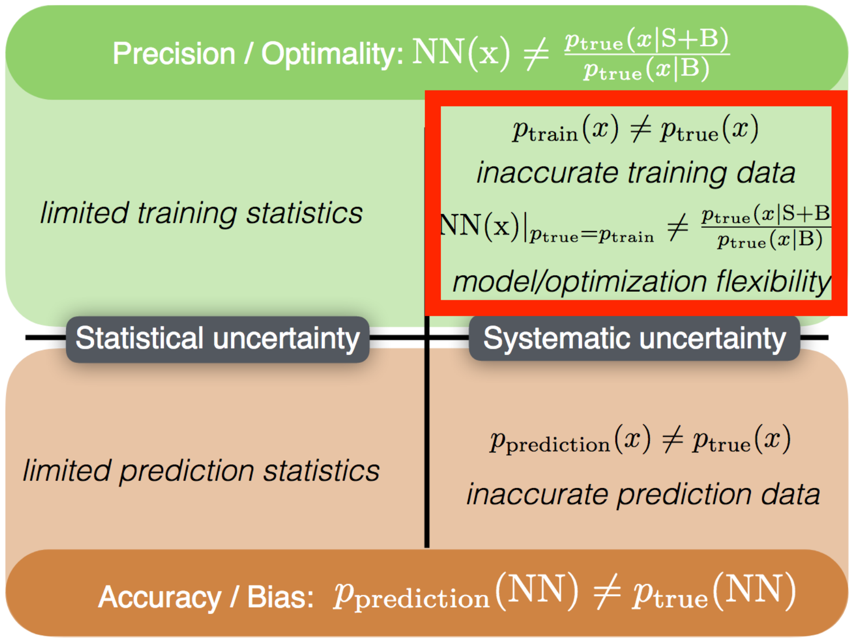
You can always accomplish this by bootstrapping: making pseudo-datasets from resampling and then retraining.

It is important to fix the NN initialization so that you are not also testing your sensitivity to that.

This can be painful because it requires retraining many NNs.

Maybe can accomplish with one Bayesian NN? See e.g. S. Bollweg, et al., SciPost Phys. 8, 006 (2020), 1904.10004 for a particle physics example.

How to estimate precision syst. uncerts.

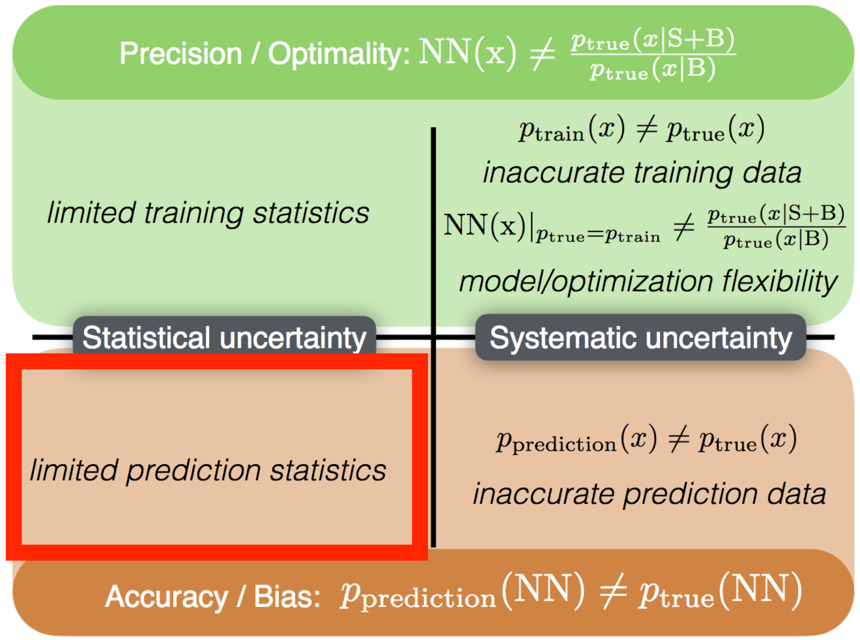


As with all systematic uncertainties, this is hard to quantify.

One component is due to the modeling of p(x) - more on this later.

Testing the flexibility of the network requires checking the sensitivity to the architecture (#layers, nodes/layer, etc.), the initialization, the training procedure (#epochs, learning rate, etc.)

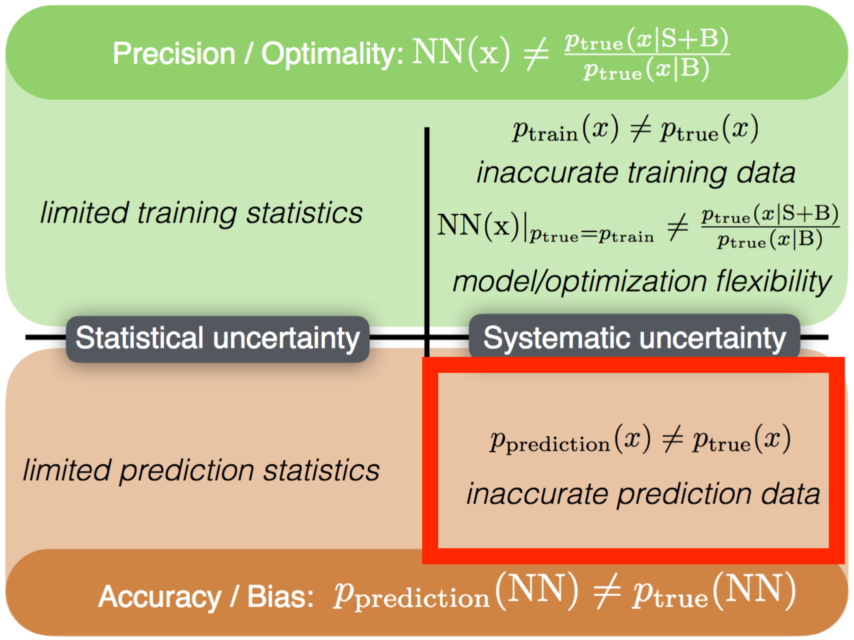
How to estimate bias stat. uncerts.



Can be estimated via bootstrapping. Less painful here because the NN's are fixed.

N.B. it may be possible to design a network that is designed to minimize uncertainty at inference. This does not work in all cases, but early studies in particle physics seem promising: S. Wunsch et al., 2003.07186, P. da Castro et al., CPC 244 (2019) 170, 1806.04743

How to estimate bias syst. uncerts.



This is the trickiest one…

…because we need the uncertainty on the modeling of x and x can be high-dimensional!

In many cases, the uncertainties factorize, e.g. the uncertainty on two photon energies can be decomposed into the uncertainty on each photon.

However, in many cases, we simply do not know the full uncertainty model (= nuisance parameters and their distribution)

High-dimensional Bias Uncertainties

One word of caution: current paradigm for uncertainties may be too naive for high-dimensional analysis!

(truly end-to-end)

e.g. for some uncertainties, we often compare two different models - one nuisance parameter.

How can we even see how sensitive we are to high-dimensional effects?

High-dimensional Bias Uncertainties

One word of caution: current paradigm for uncertainties may be too naive for high-dimensional analysis!

(truly end-to-end)

e.g. for some uncertainties, we often compare two different models - one nuisance parameter.

How can we even see how sensitive we are to high-dimensional effects?

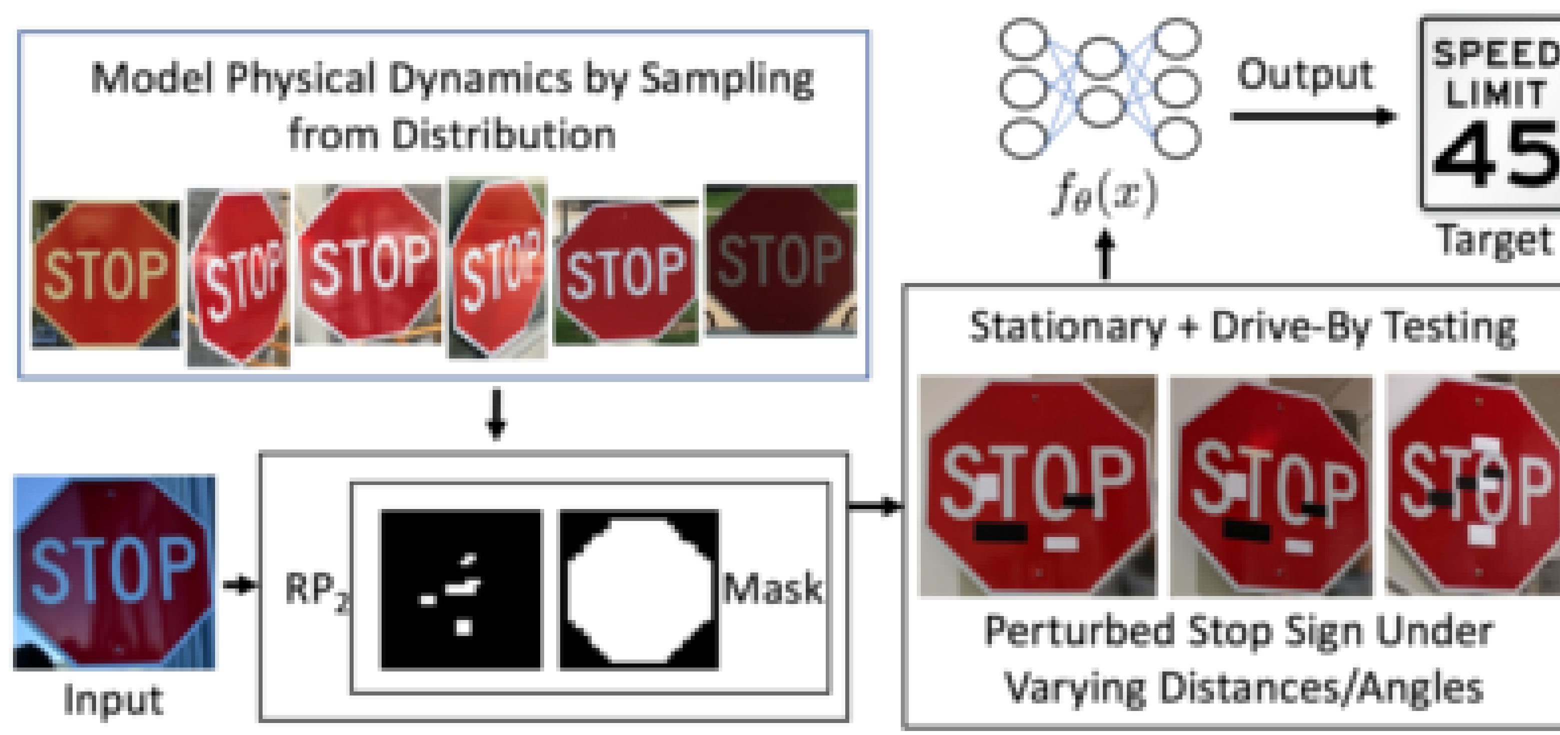
Answer: borrow tools from AI Safety

AI Safety

There is a vast literature on how easy it is to “attack” a NN.

They want to know: how subtle can an attack be and still significantly impact the output.

We know (hope?!) that nature is not evil, but these tools can help us probe the high-dimensional sensitivity of our NNs.



K. Eykholt et. al, 1707.08945

Bounding high-dim. uncerts: strategy

J = collision event (in all of its high-dimensional glory)

f = fixed classifier for signal vs. background

Loss

$$\mathcal{L}_{\text{sig}} = \log(1 - f(g(J))),$$

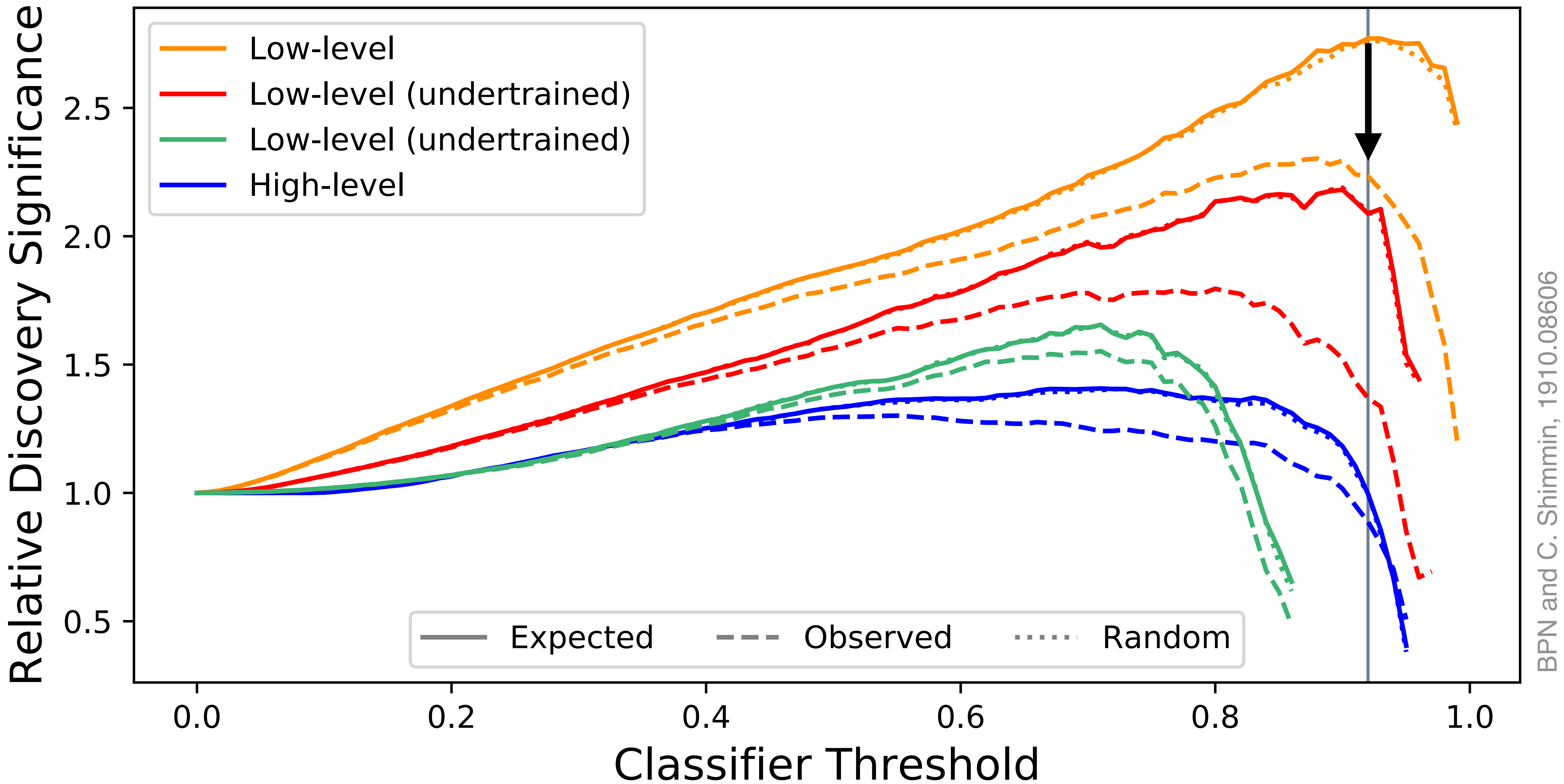
$$\mathcal{L}_{\text{bg}} = \lambda_{\text{cls}}(f(J) - f(g(J)))^2 + \sum_i \lambda_{\text{obs}}^{(i)}(\mathcal{O}^{(i)}(J) - \mathcal{O}^{(i)}(g(J))^2$$

BPN and C. Shimmin, 1910.08606

g is a learned NN that maps J to $J + \delta J$.

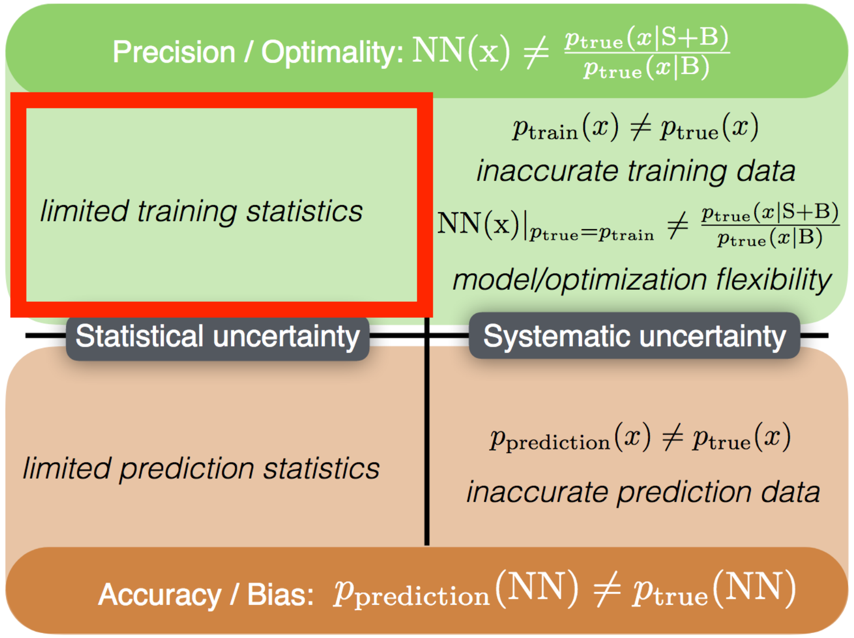
O(J) are observables that will be validated in the CR.

High-dimensional Uncertainty



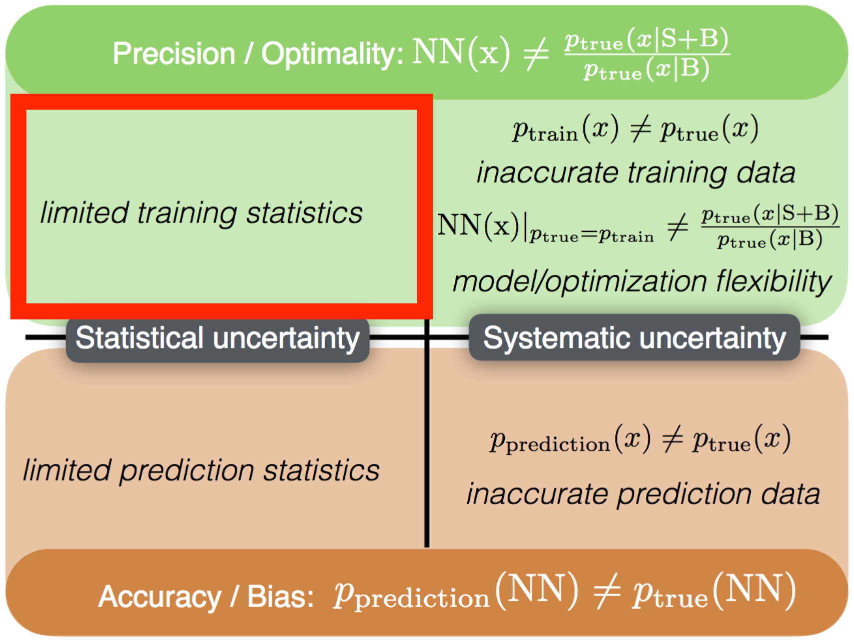
“worst-case uncertainty”

How to reduce precision stat. uncerts.



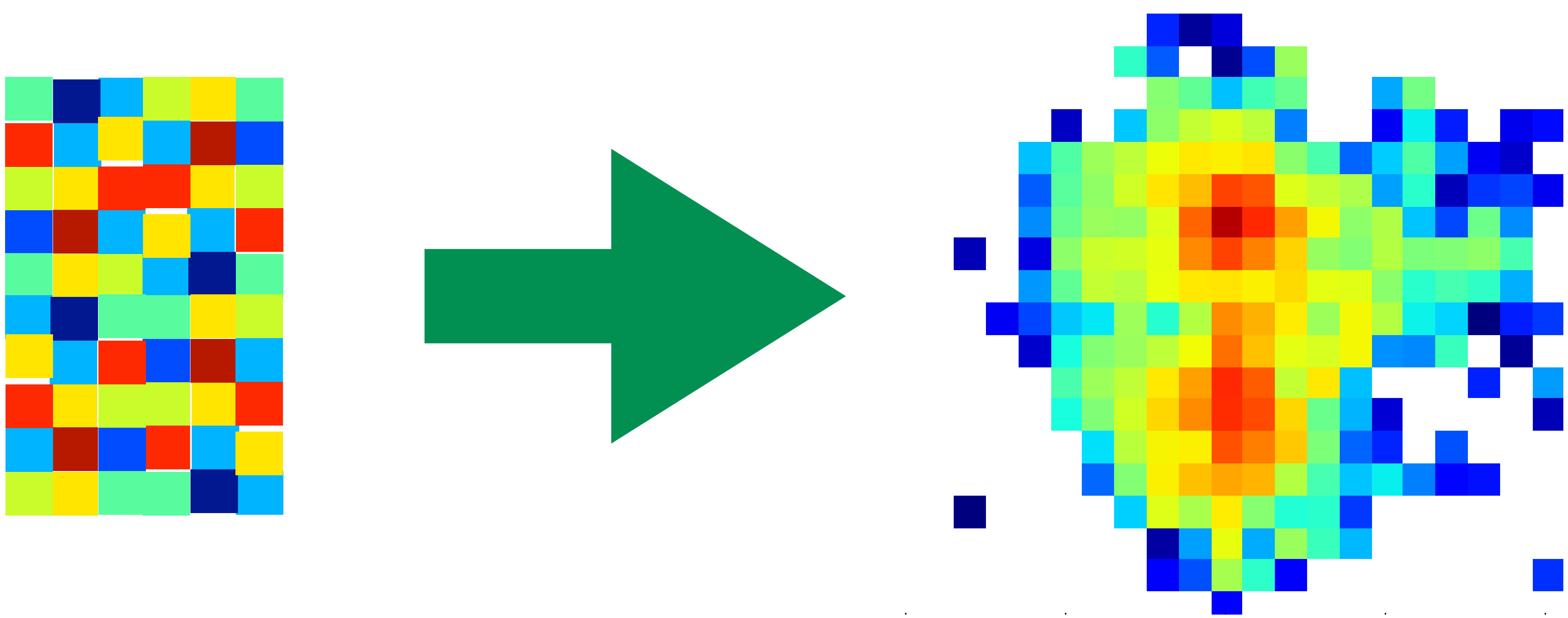
Train with more events!

How to reduce precision stat. uncerts.

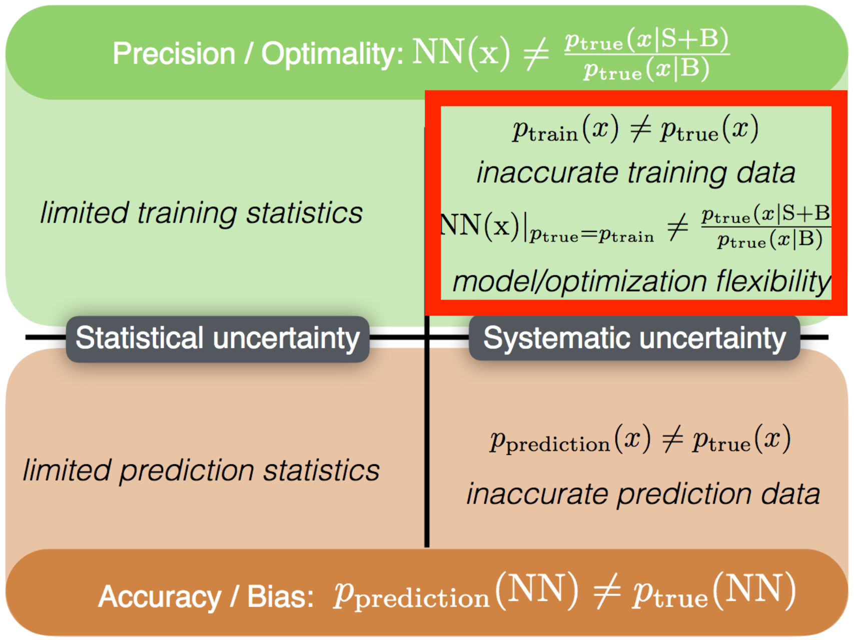


Train with more events!

…maybe use NN's to help with that



How to reduce precision syst. uncerts.



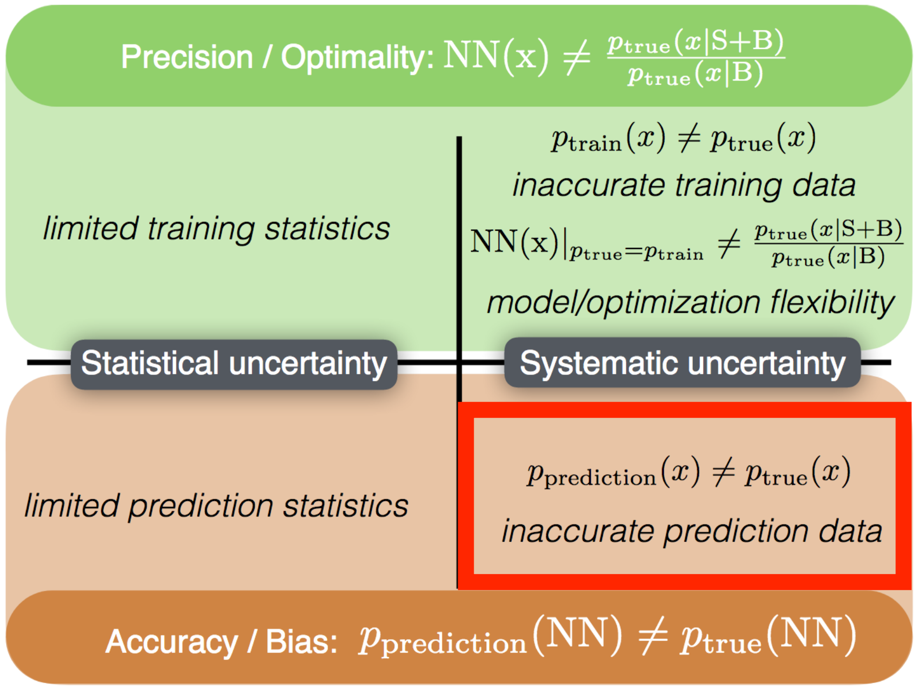
Might be possible to reduce uncertainties or at least alleviate analysis complexity by making your NN independent of known nuisance parameters*.

…might also be better to explicitly depend on the nuisance parameters and profile them in data.

*see G. Louppe, et al., NIPS 2017, 1611.01046 for particle physics and many papers since.

How to get around high-D bias uncerts?

Work hard to understand the true nuisance parameters in the hypervariate parameter space.

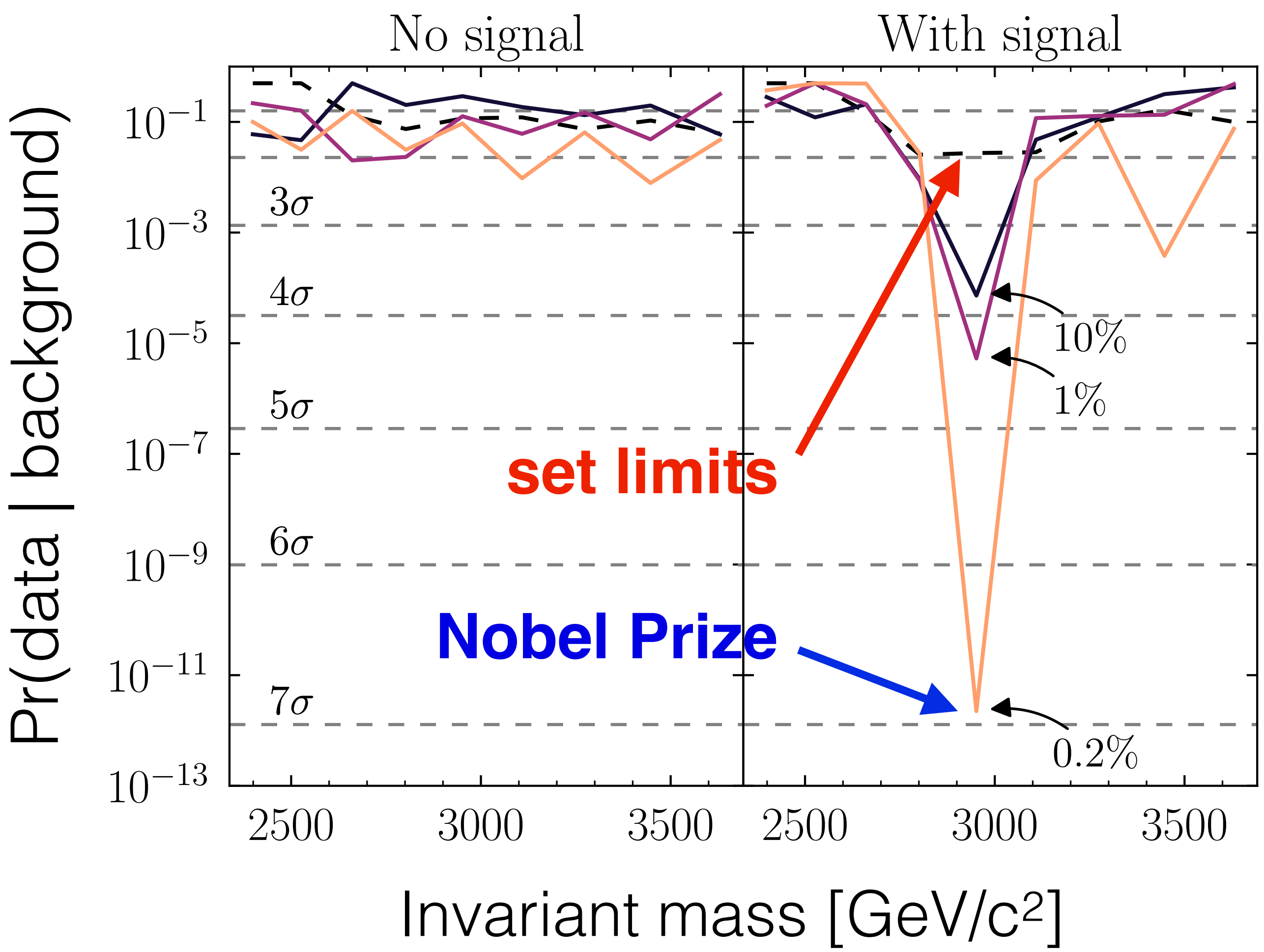


In my opinion, this is **THE** biggest challenge with deploying NN-based analyses … solving it will require hard physics work.

How to get around high-D bias uncerts?

Work hard to understand the true nuisance parameters in the hypervariate parameter space.

Don't use simulation!
(not always possible and of course, still has assumptions…)



J. Collins, K. Howe, BPN,
Phys. Rev. Lett. 121 (2018) 241803, 1805.02664

What is the problem?

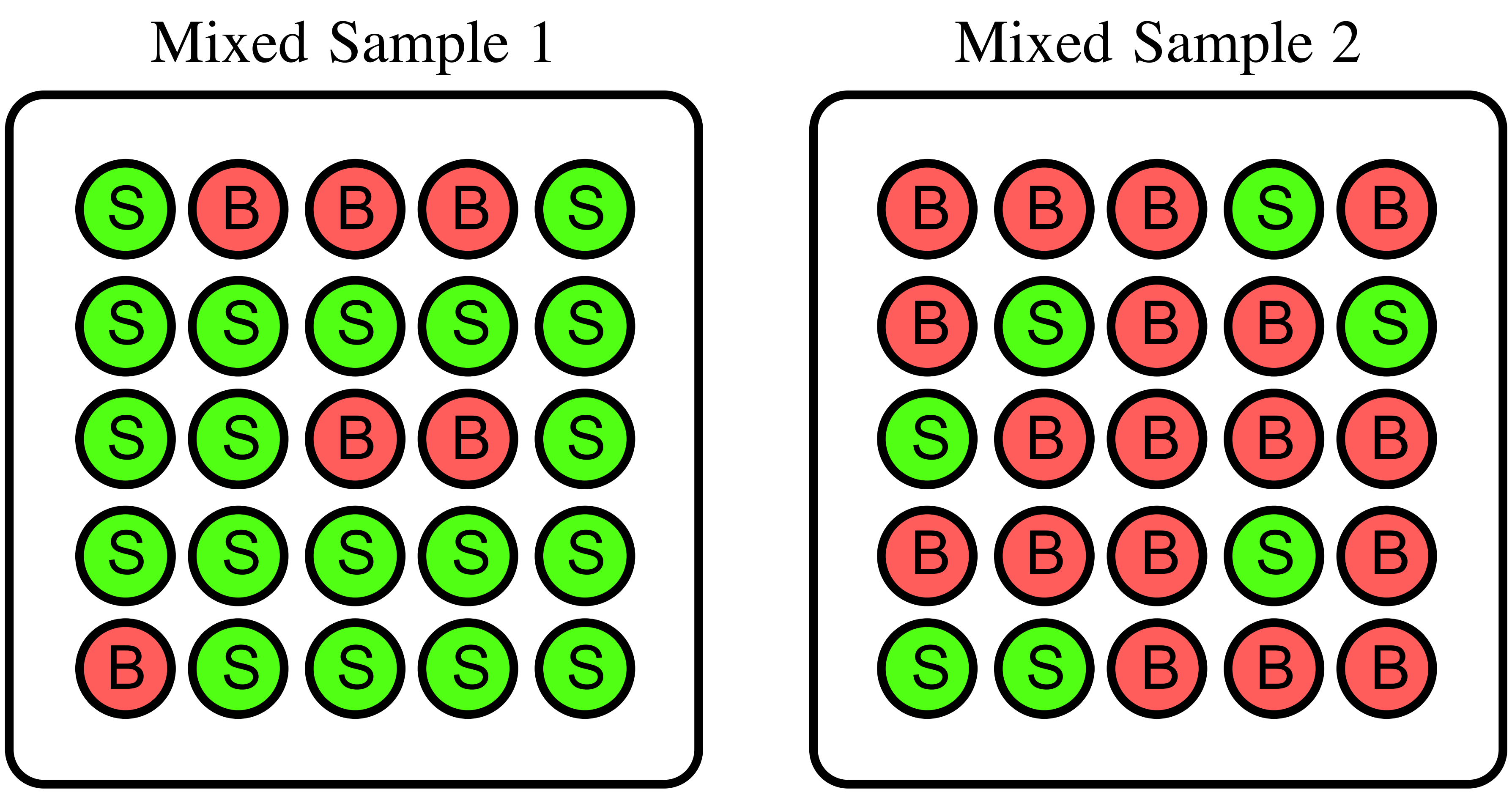
Why can't I just pay some physicists to label events and then train a neural network using those labels?

Image credit: pixabay.com

Answer: this is not cats-versus-dogs … thanks to quantum mechanics it is **not possible to know** what happened.

What is the problem?

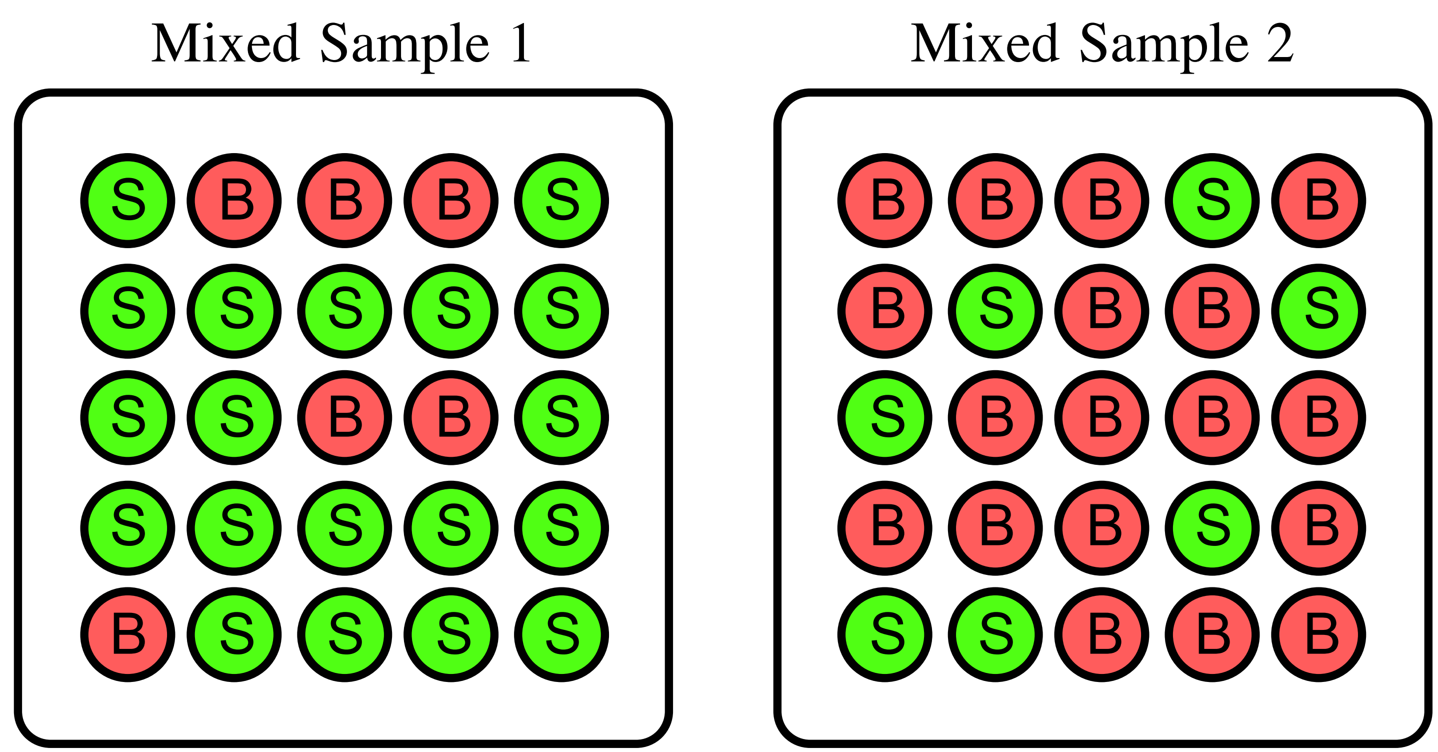
The data are unlabeled and in the best case, come to us as mixtures of two classes (“signal” and “background”).



(we don’t get to observe the color of the circles)

Weak supervision: *Classification Without Labels*

Can we learn without any label information?



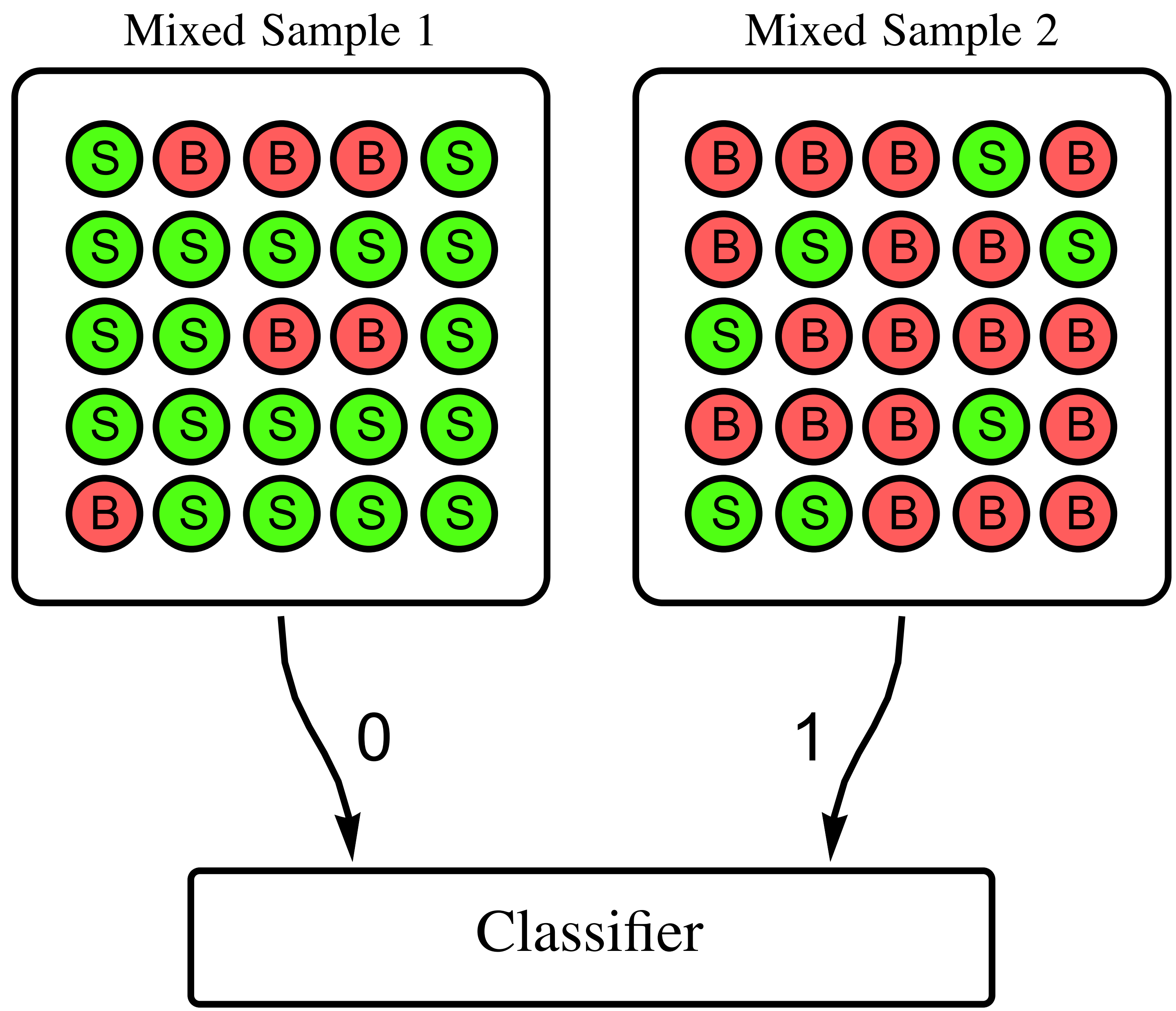
E. Metodiev, BPN, J. Thaler, JHEP 10 (2017) 51

Weak supervision: *Classification Without Labels*

Can we learn without any label information?

Yes !

Training on impure samples is (asymptotically) equivalent to training on pure samples



E. Metodiev, **BPN**, J. Thaler, JHEP 10 (2017) 51

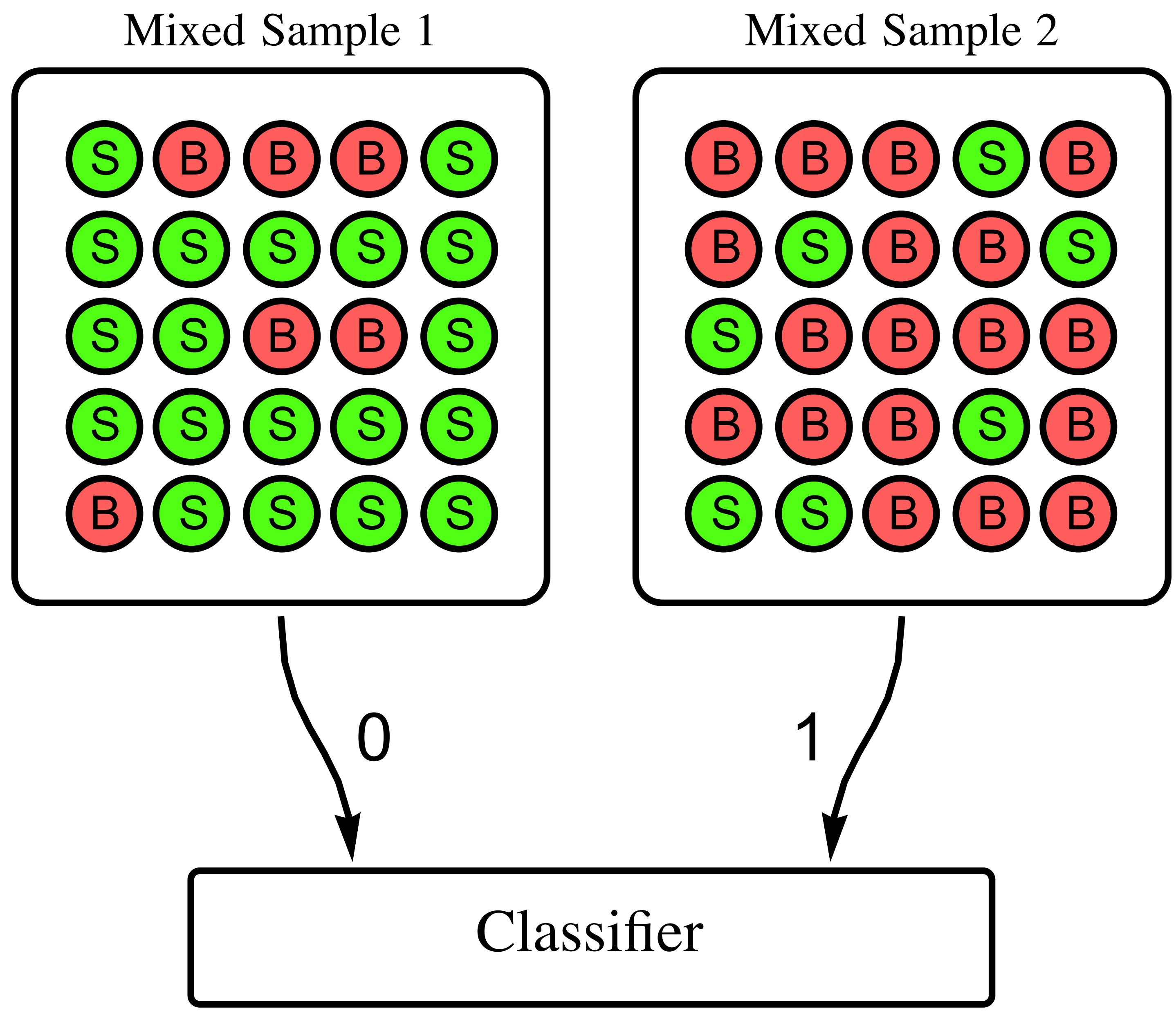
Weak supervision: *Classification Without Labels*

Can we learn without any label information?

Yes !

Training on impure samples is (asymptotically) equivalent to training on pure samples

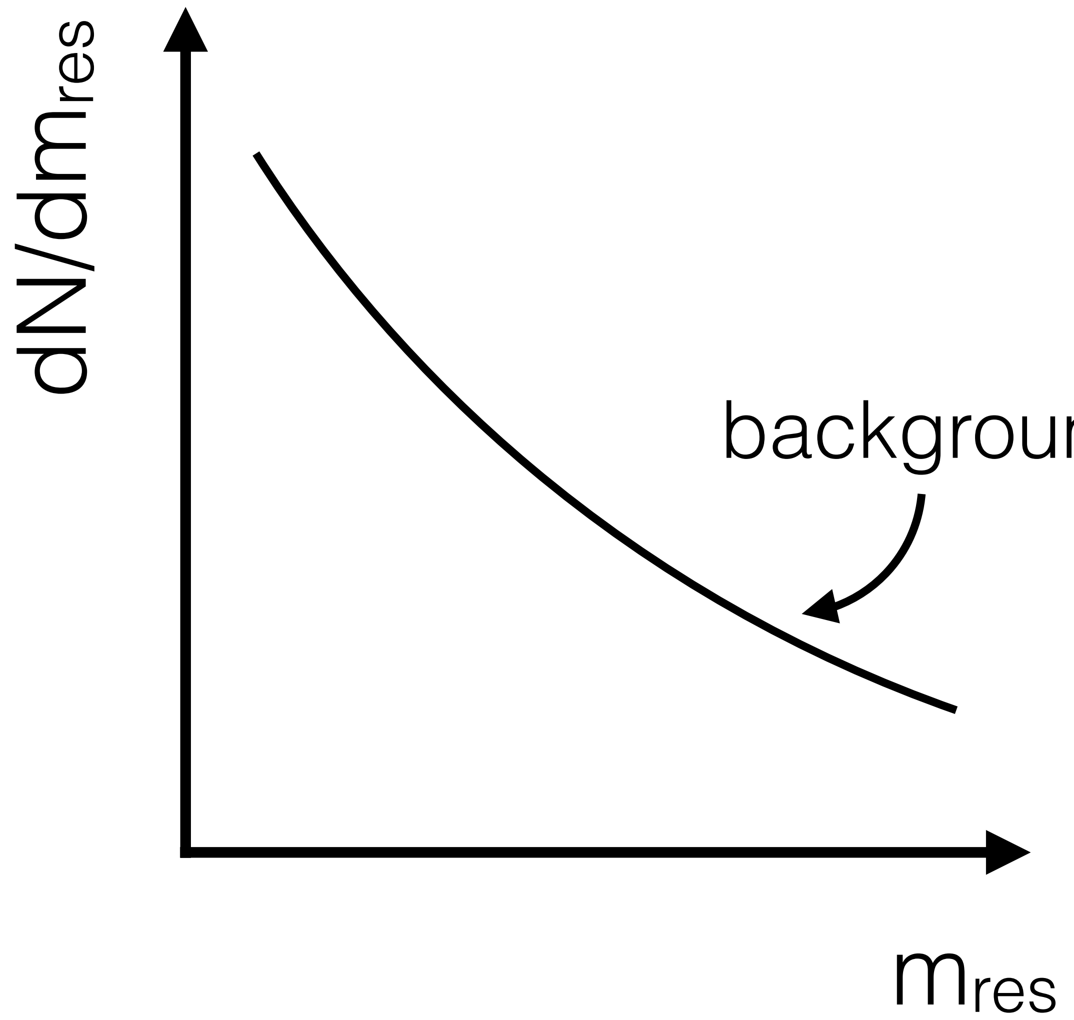
Exercise: What does this mean and can you prove it?



E. Metodiev, **BPN**, J. Thaler, JHEP 10 (2017) 51

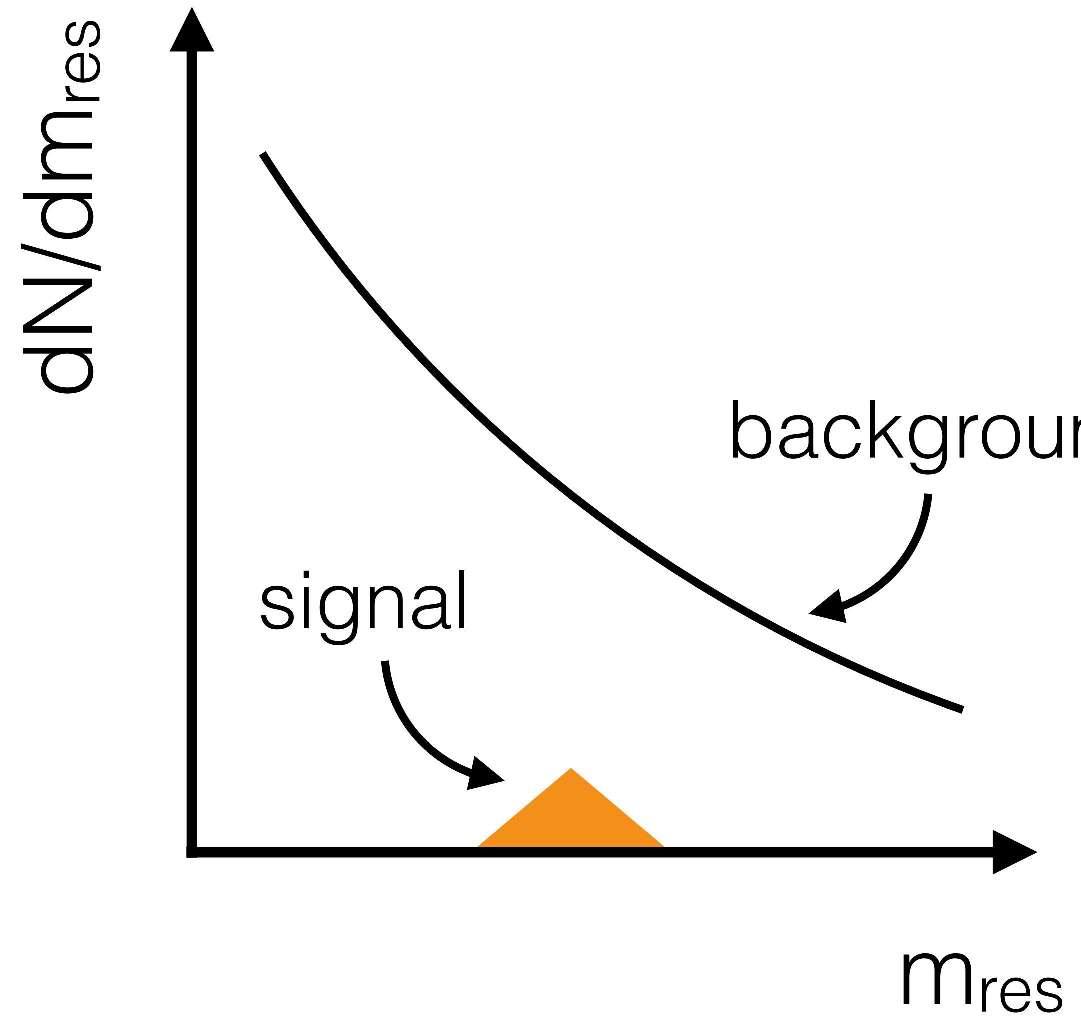
CWoLa for anomaly detection

J. Collins, K. Howe, BPN, Phys. Rev. Lett. 121 (2018) 241803, 1805.02664



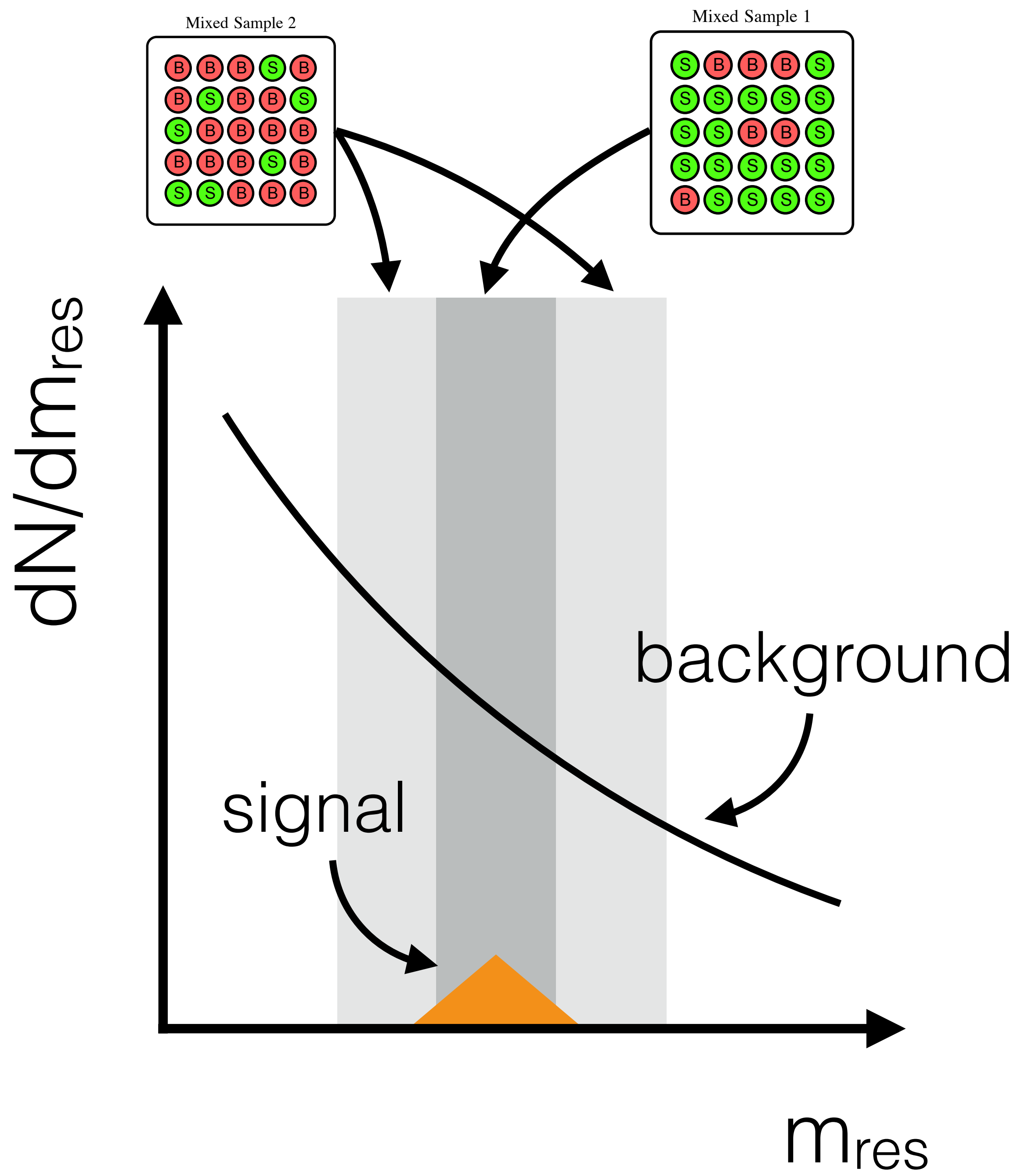
CWoLa for anomaly detection

J. Collins, K. Howe, BPN, Phys. Rev. Lett. 121 (2018) 241803, 1805.02664



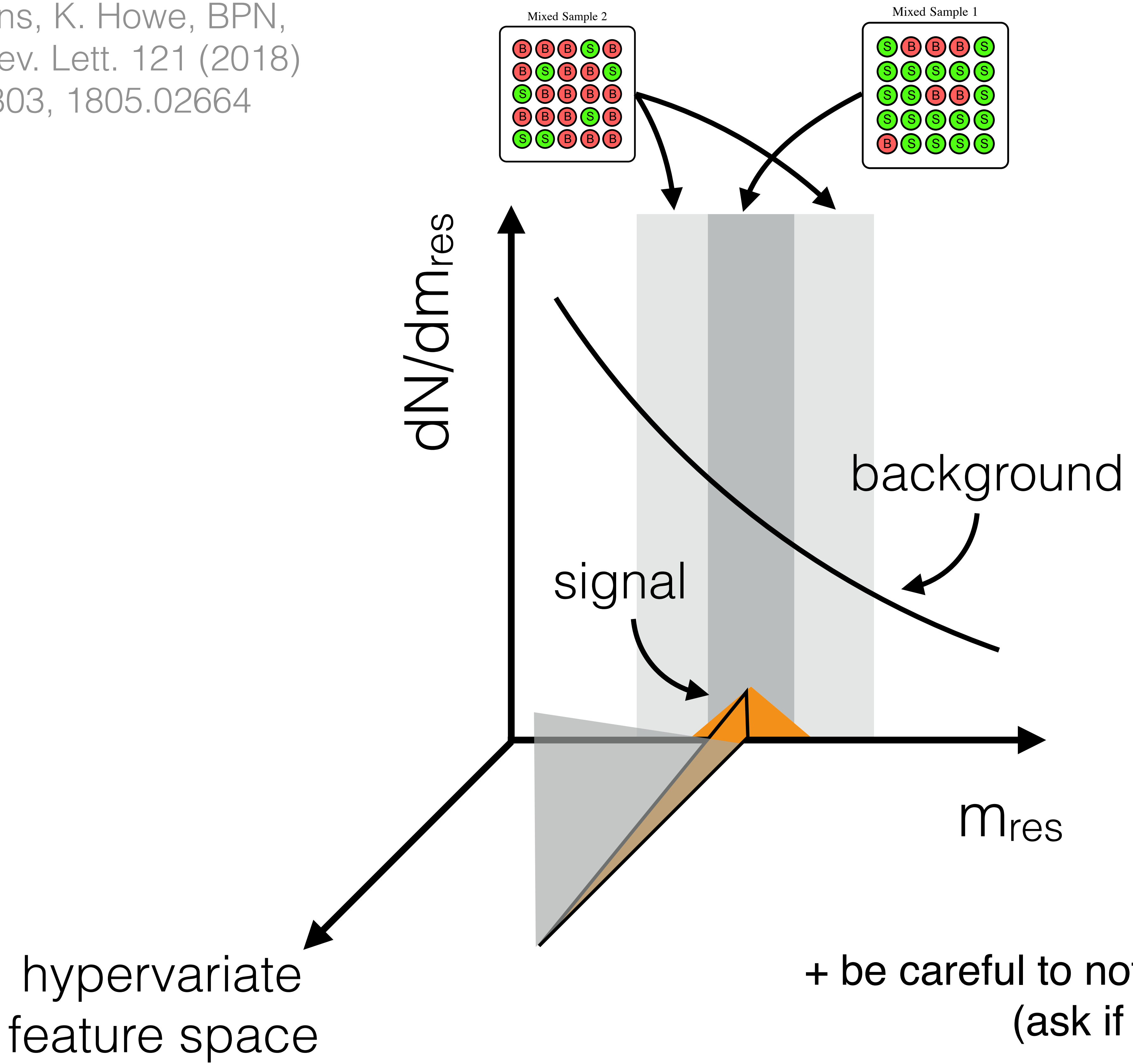
CWoLa for anomaly detection

J. Collins, K. Howe, BPN, Phys. Rev. Lett. 121 (2018) 241803, 1805.02664

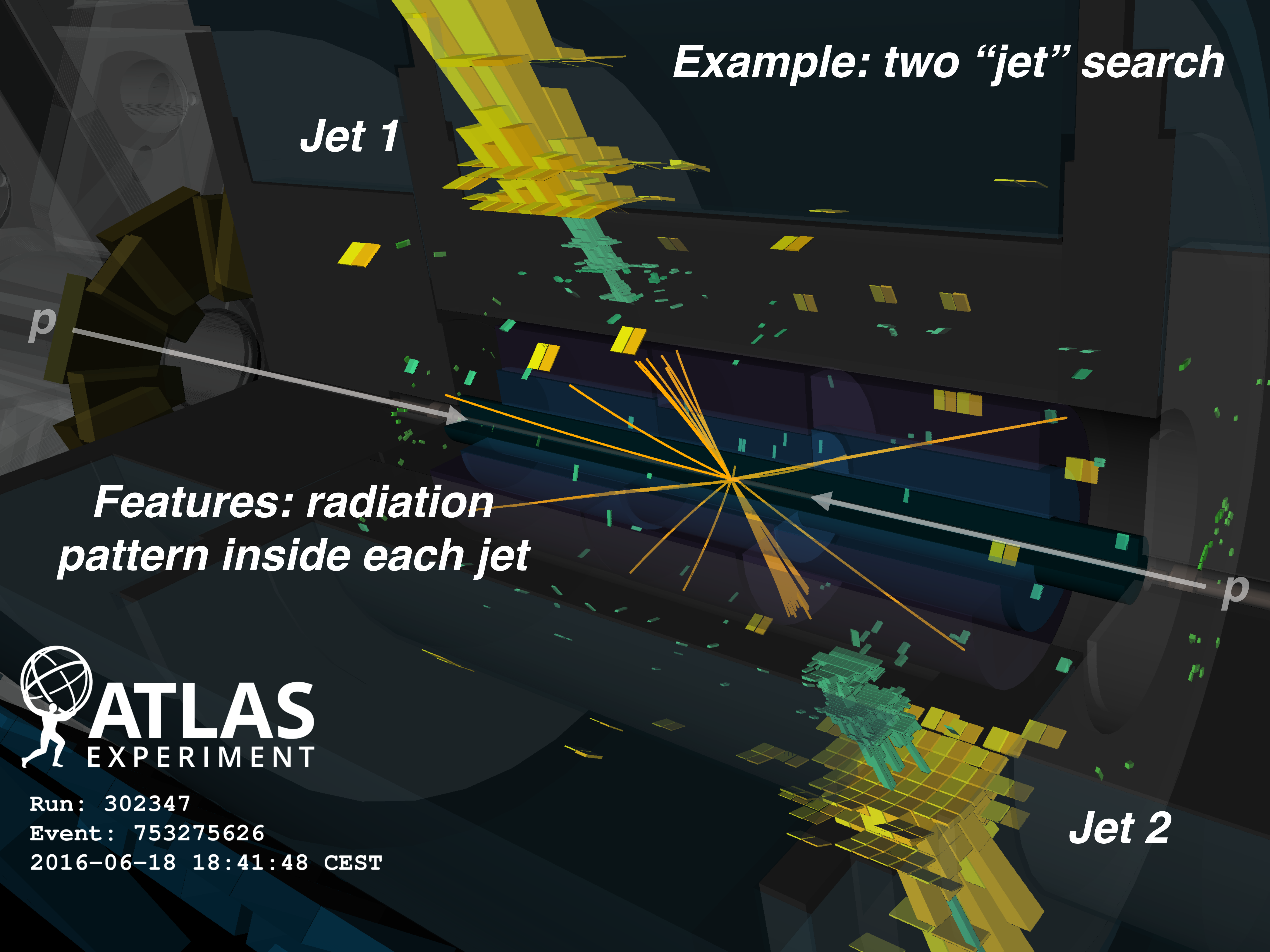


CWoLa for anomaly detection

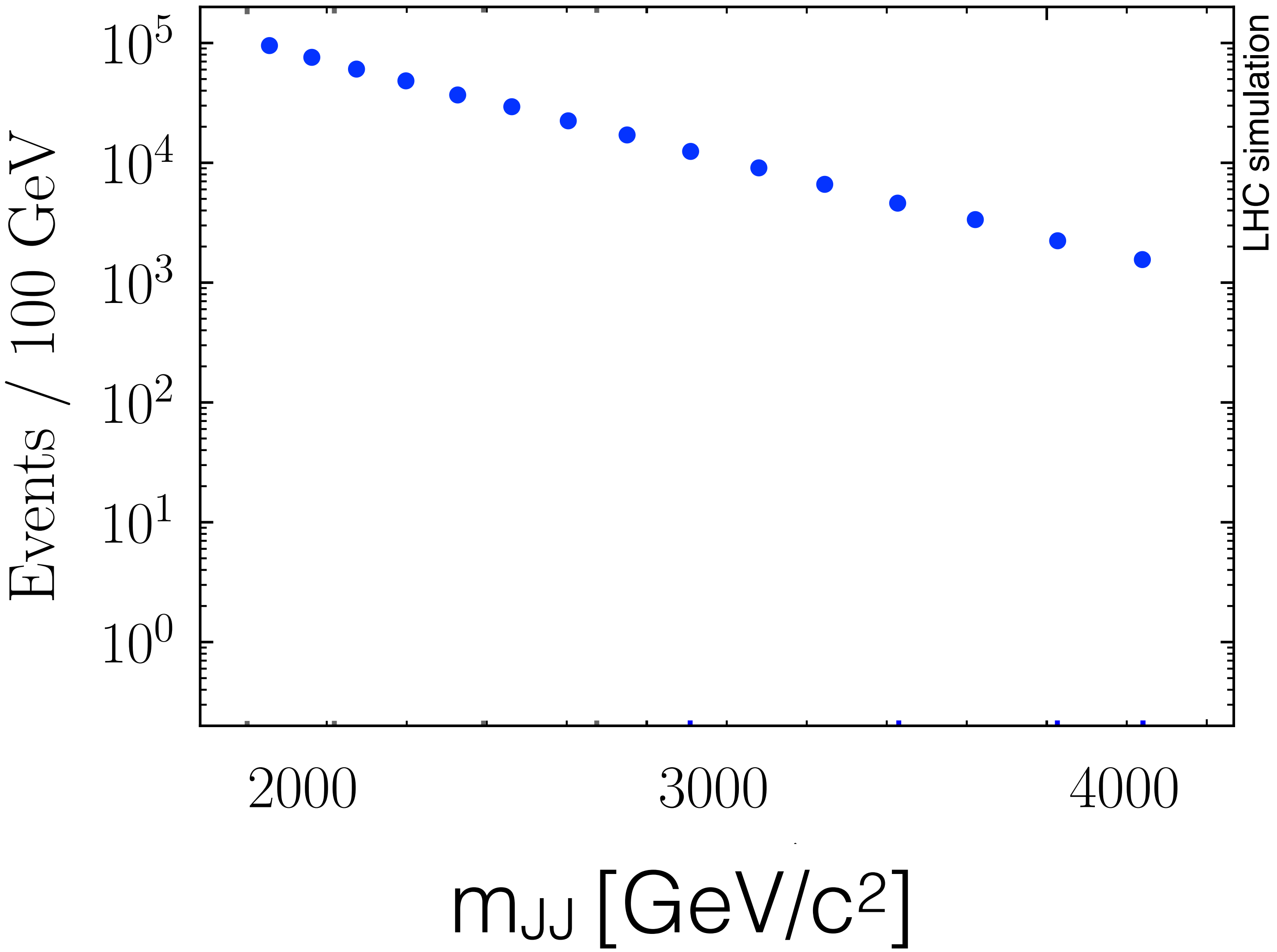
J. Collins, K. Howe, BPN, Phys. Rev. Lett. 121 (2018) 241803, 1805.02664



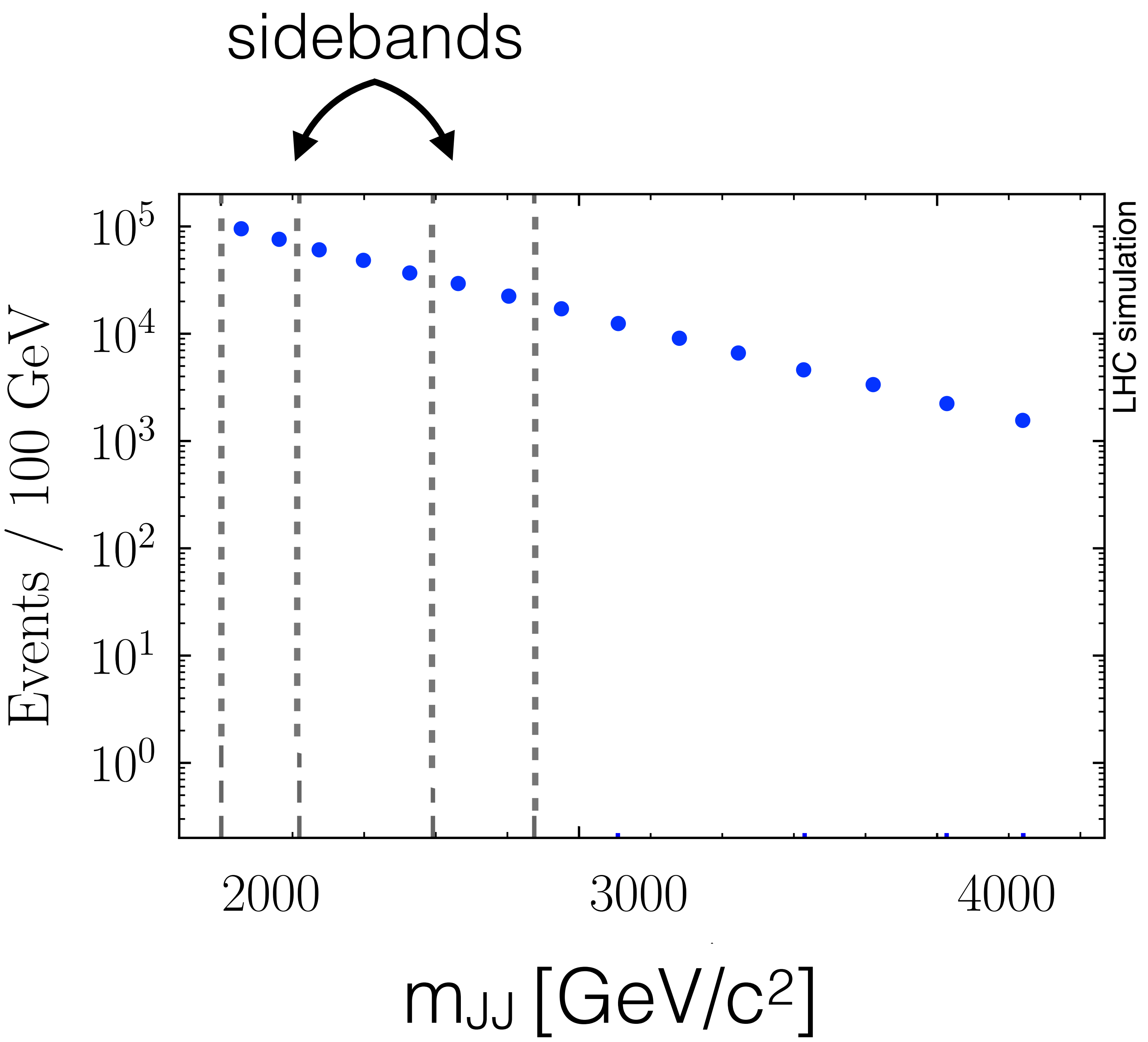
+ be careful to not pay a big trials factor (ask if interested)


Example: two "jet" search
Jet 1
p
Features: radiation
pattern inside each jet
p
ATLAS
EXPERIMENT
Run: 302347
Event: 753275626
2016-06-18 18:41:48 CEST
Jet 2

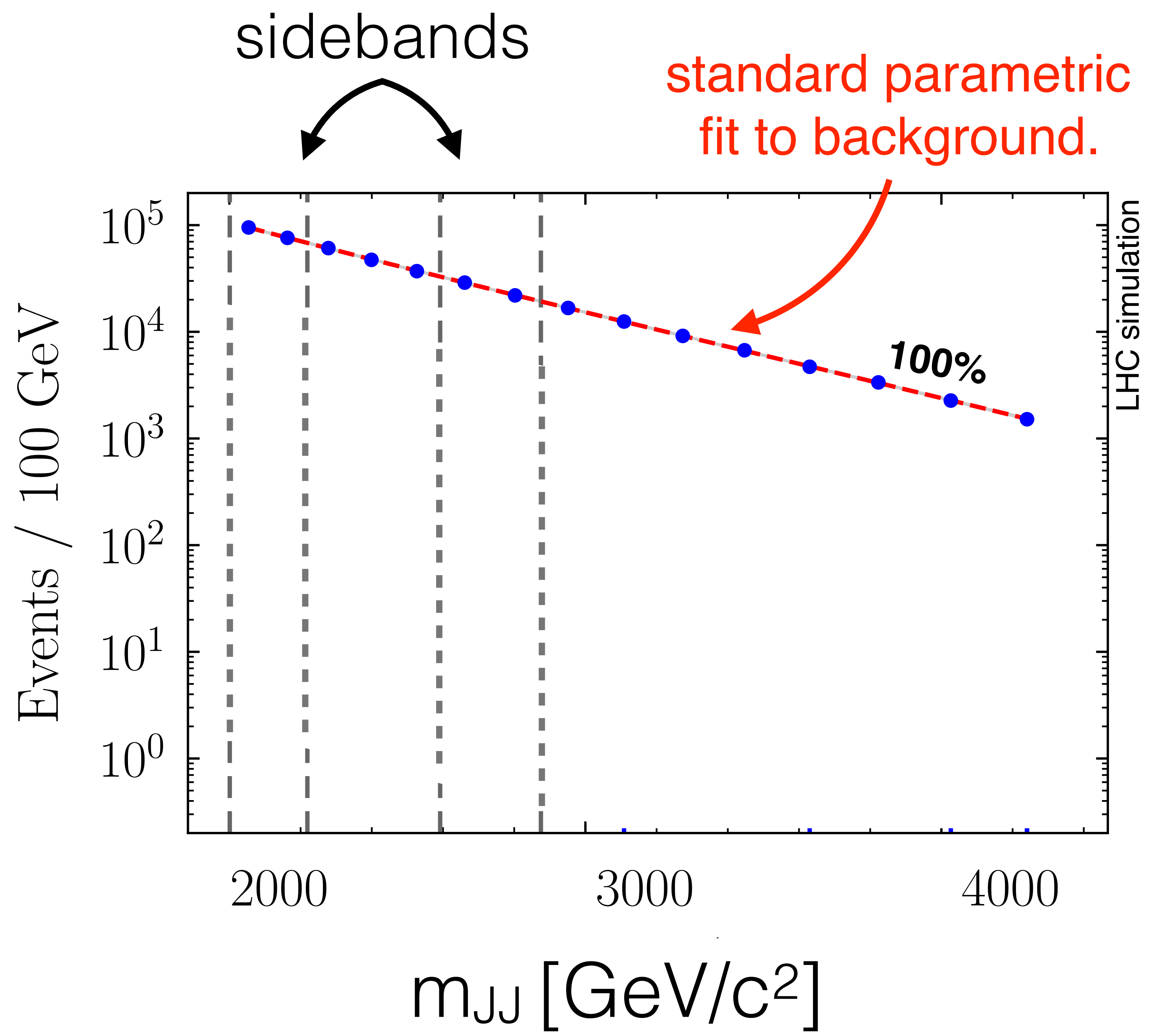
Example: two-“jet” search



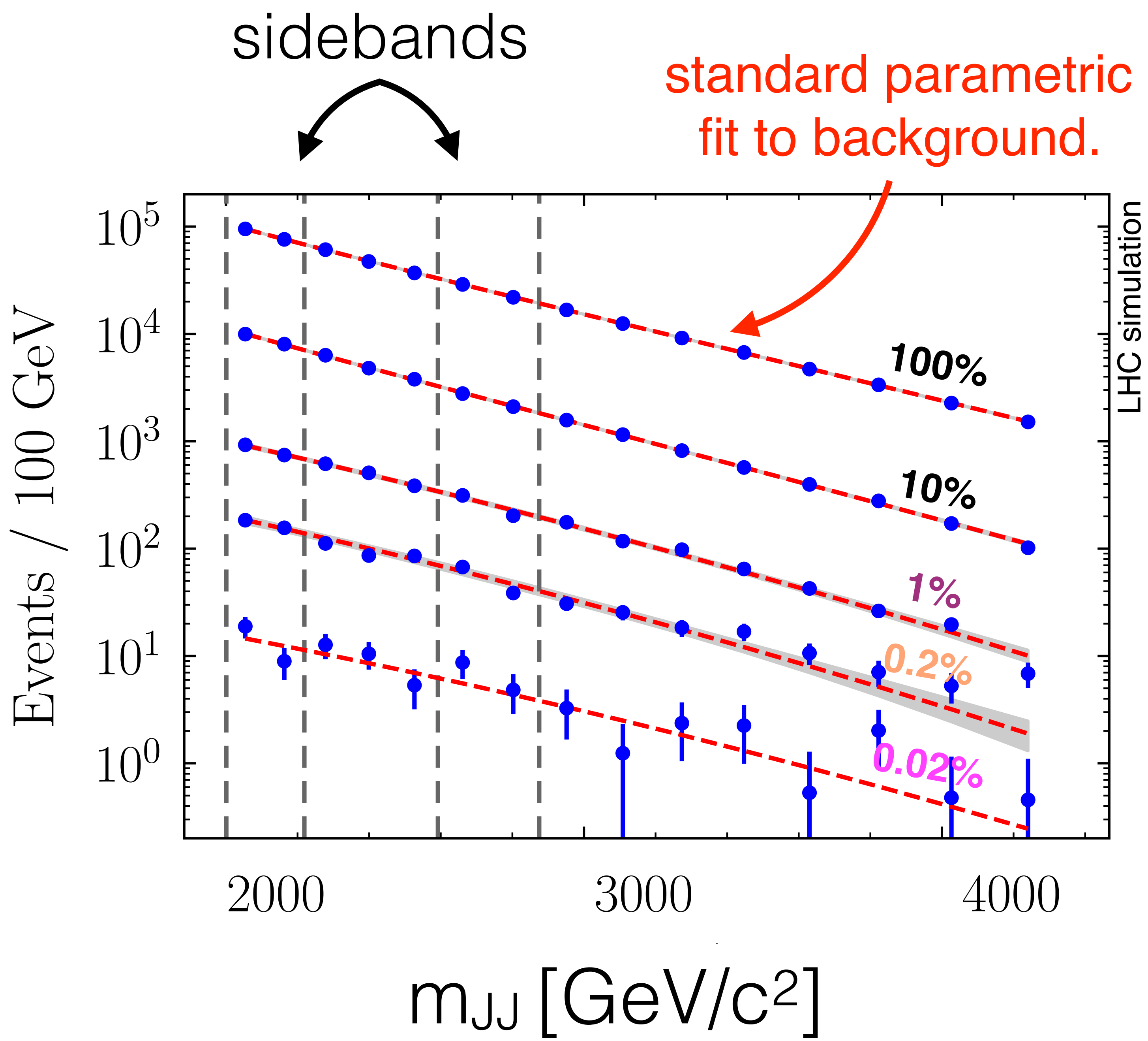
Example: two-"jet" search



Example: two-"jet" search

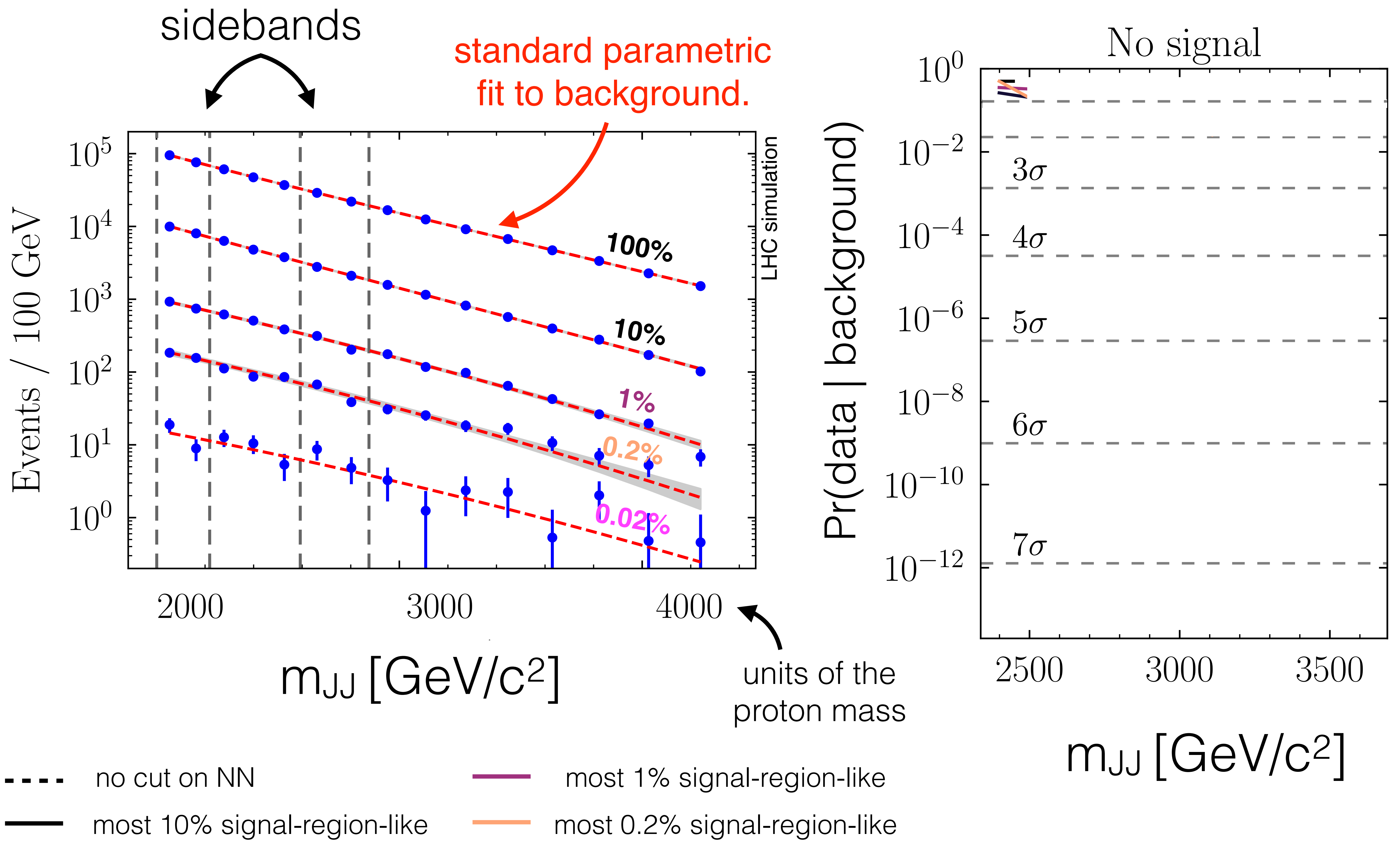


Example: two-“jet” search

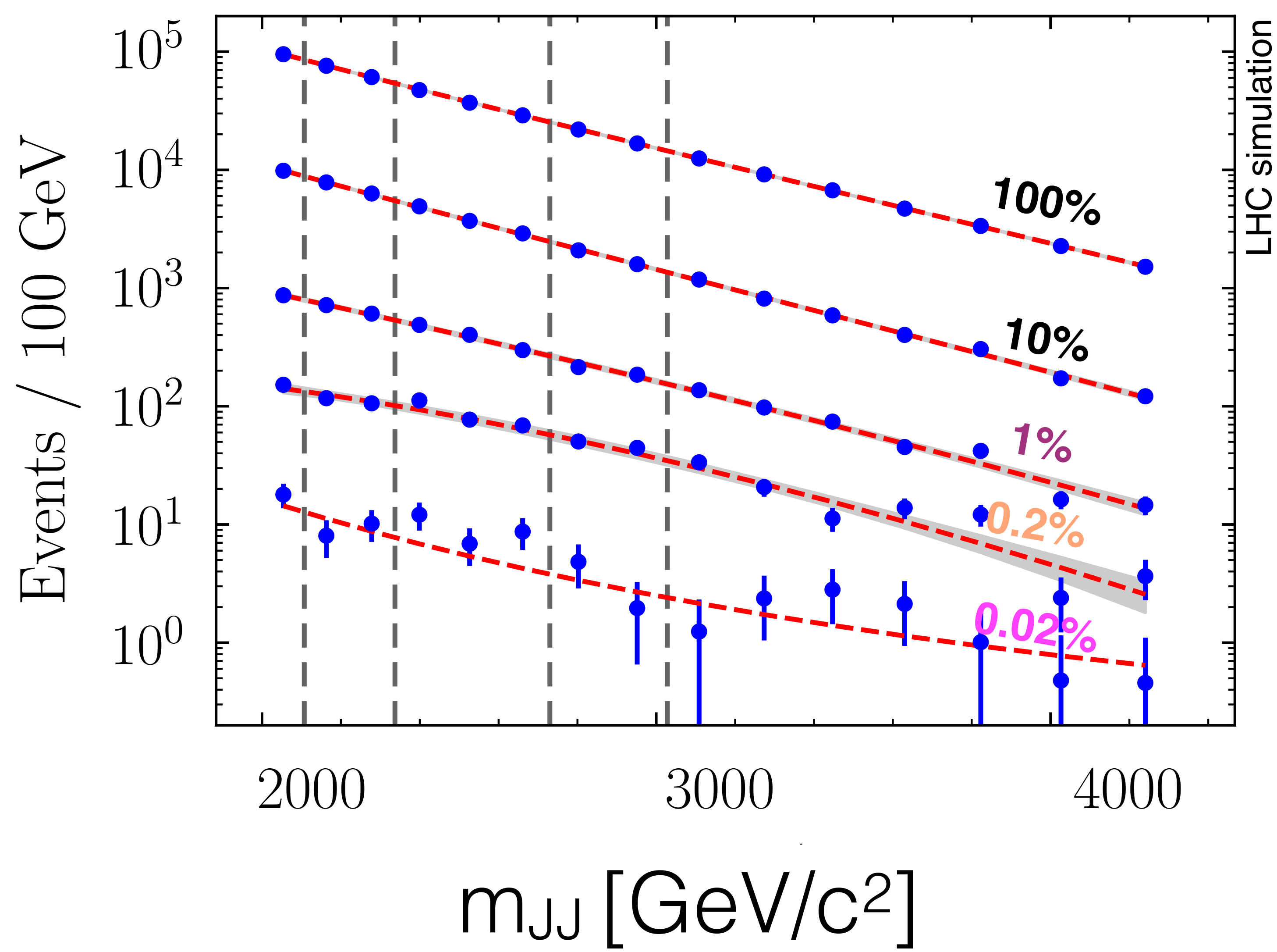


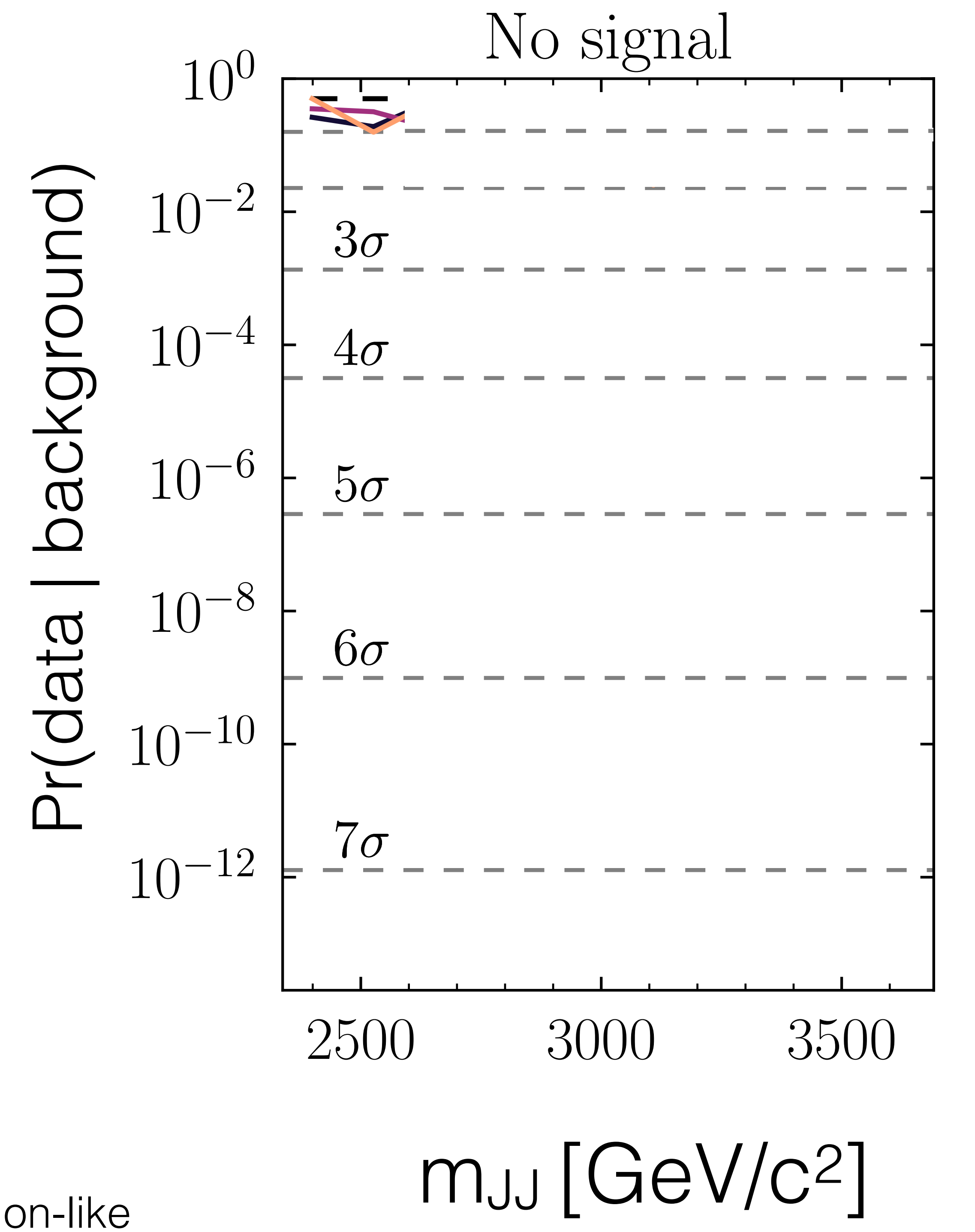
- - - - no cut on NN
—— most 10% signal-region-like
—— most 1% signal-region-like
—— most 0.2% signal-region-like

Example: two-"jet" search



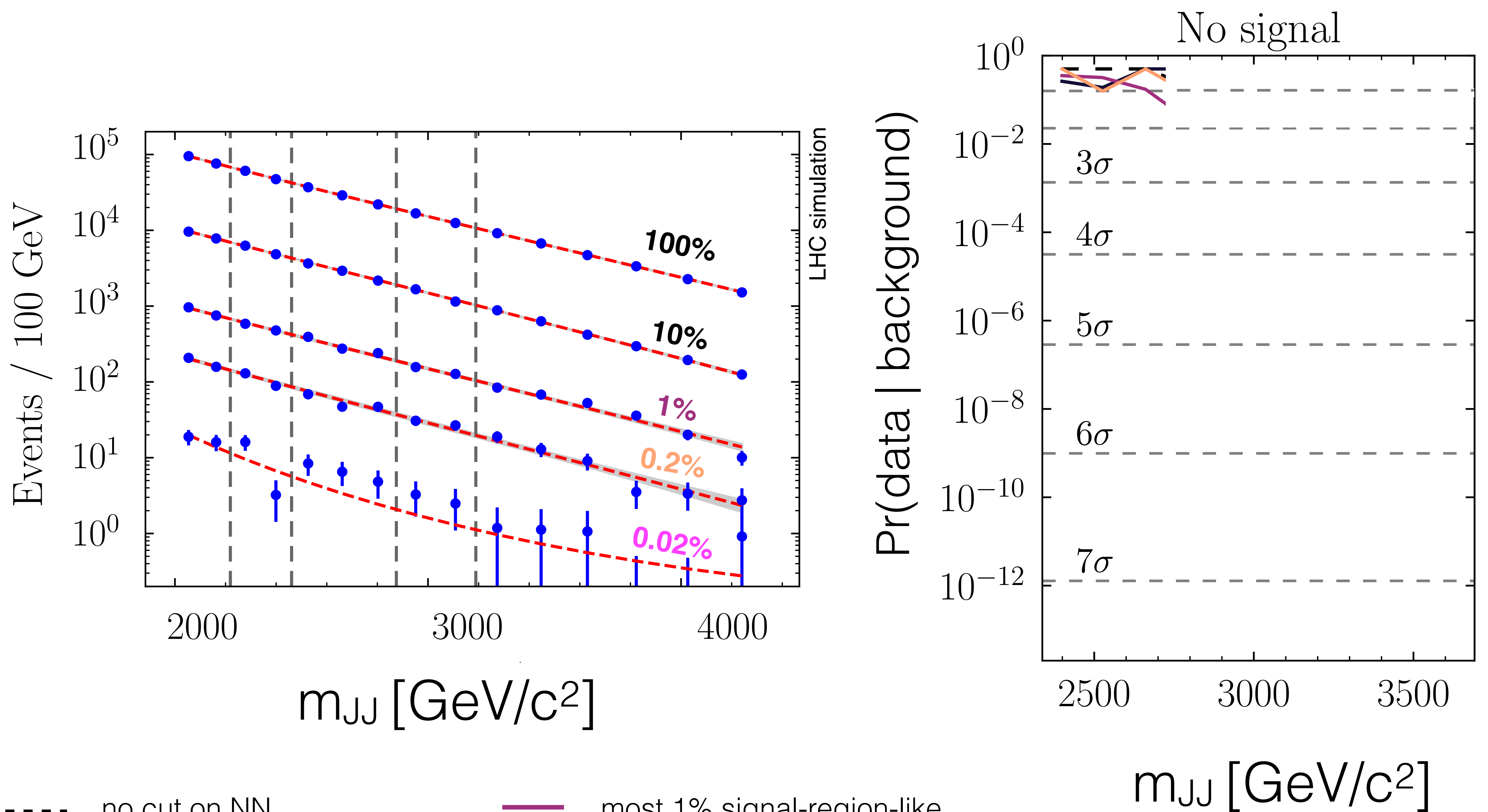
Example: two-“jet” search





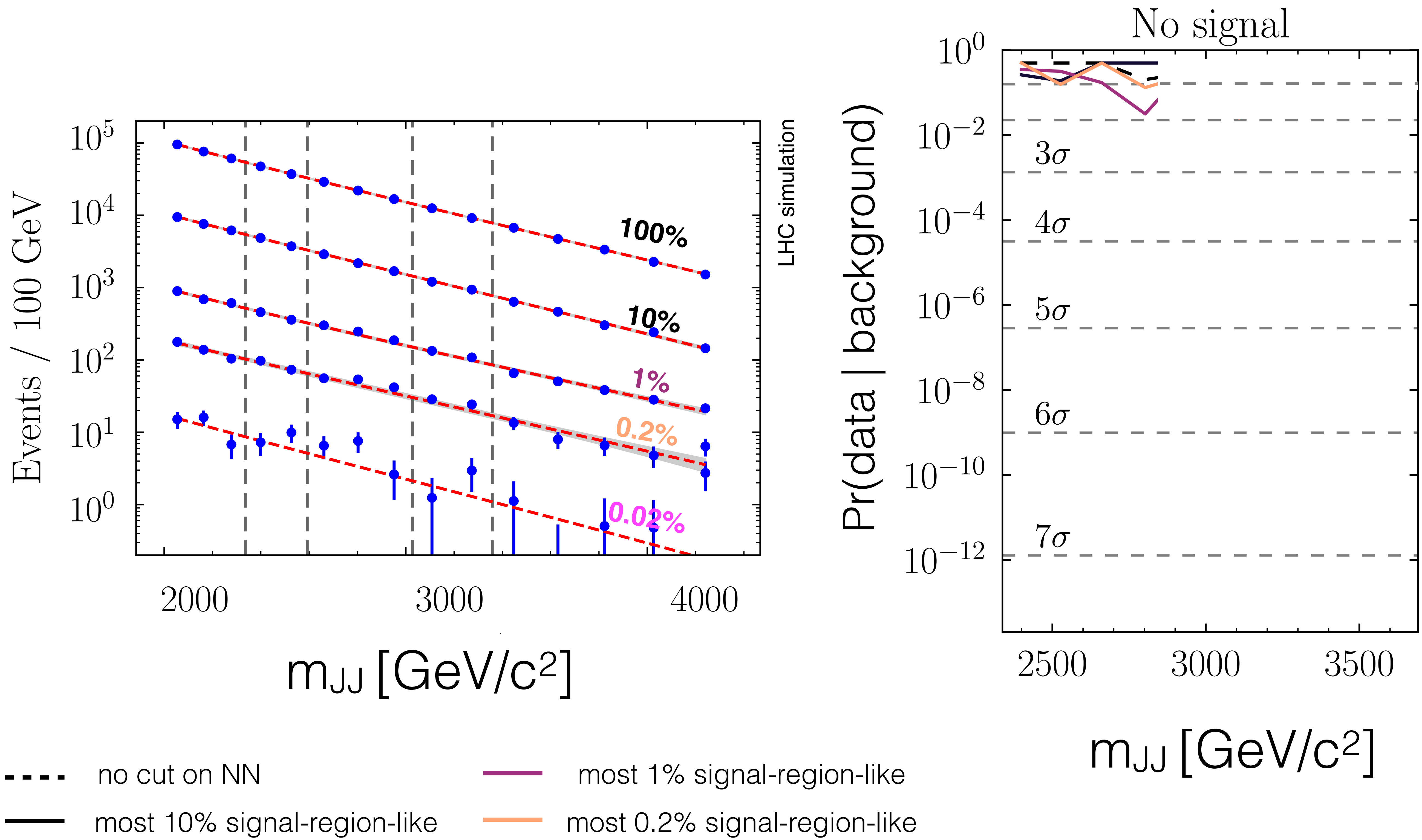
no cut on NN
most 10% signal-region-like
most 1% signal-region-like
most 0.2% signal-region-like

Example: two-“jet” search

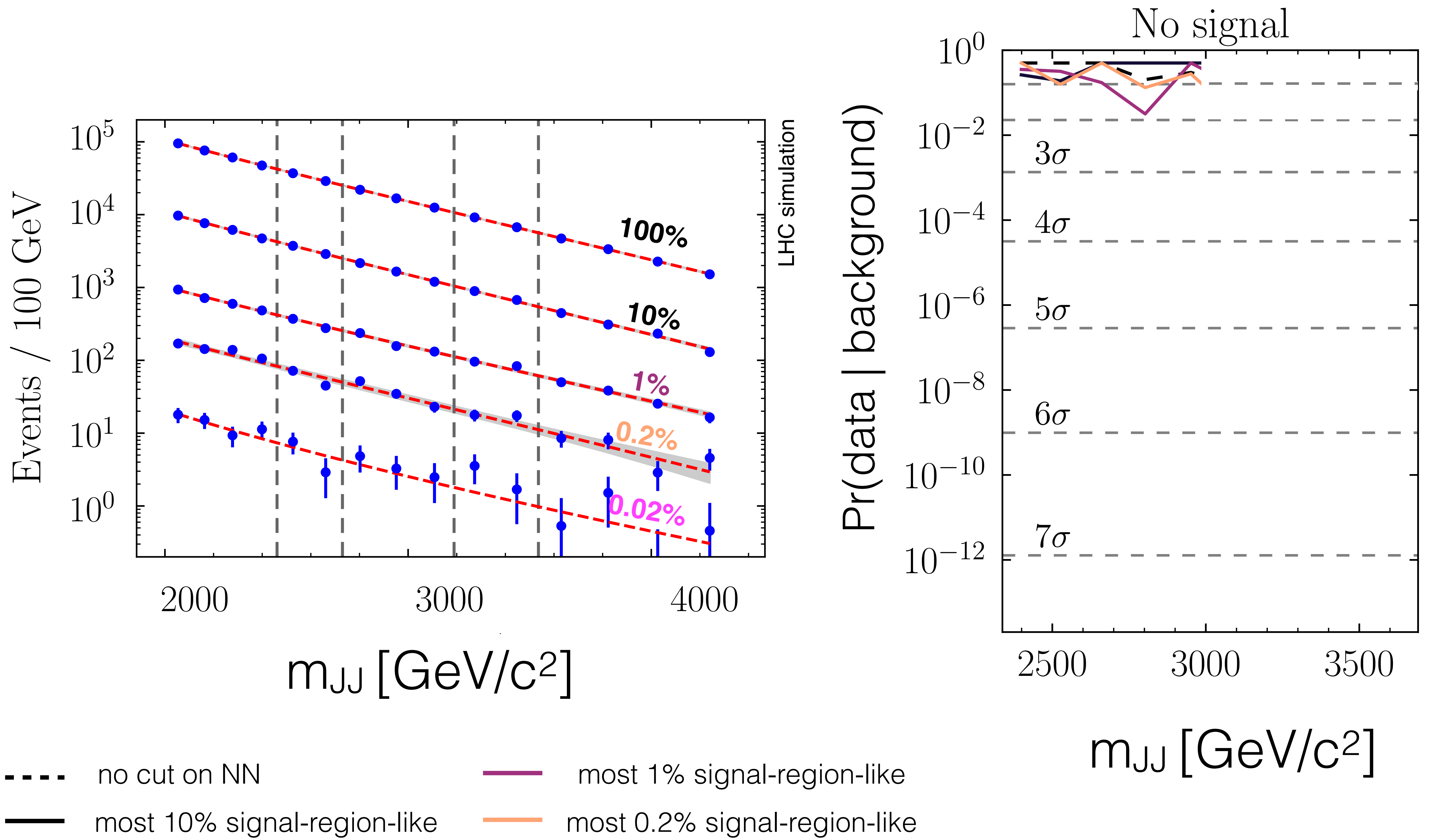


- - - no cut on NN
— most 10% signal-region-like
— most 1% signal-region-like
— most 0.2% signal-region-like

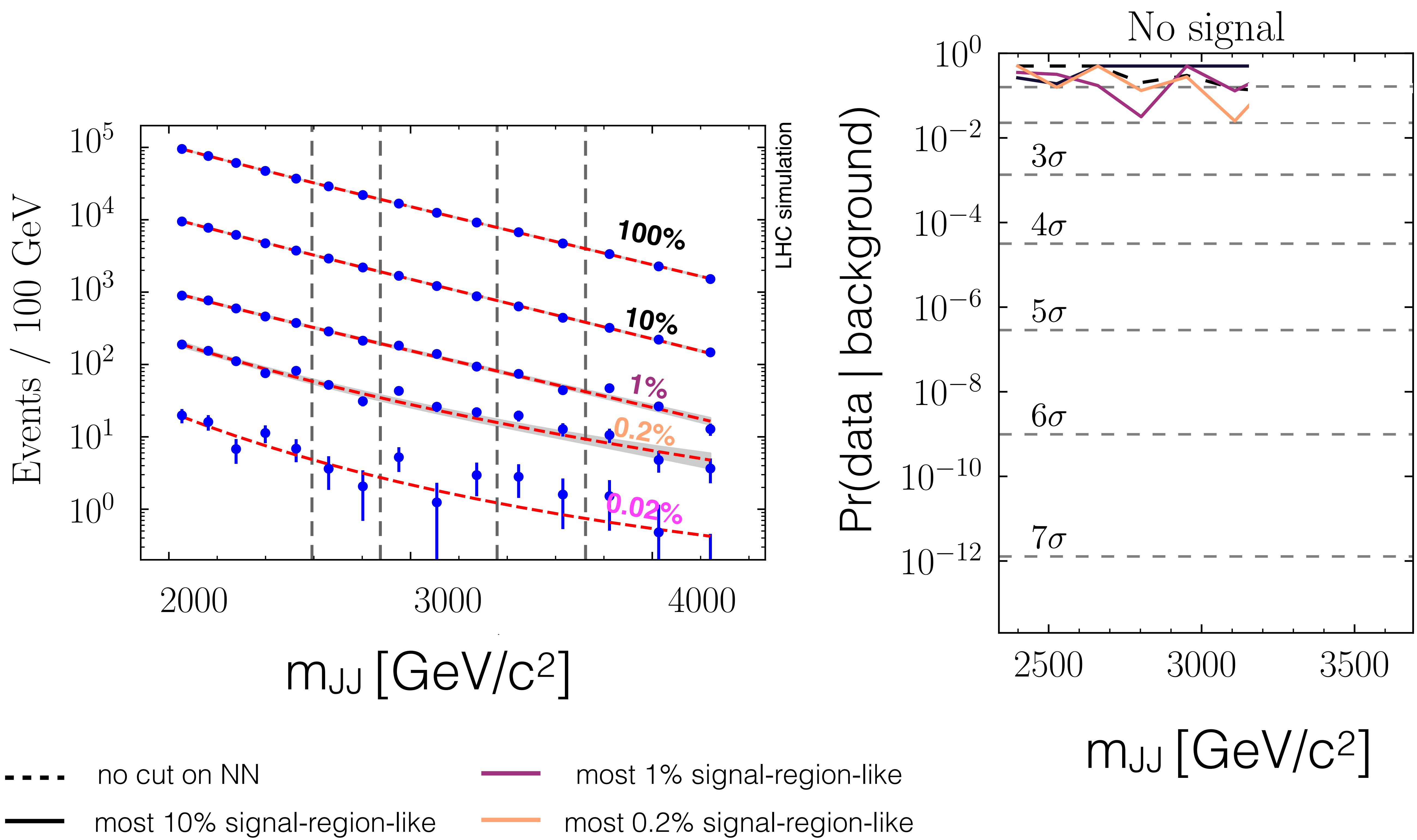
Example: two-“jet” search



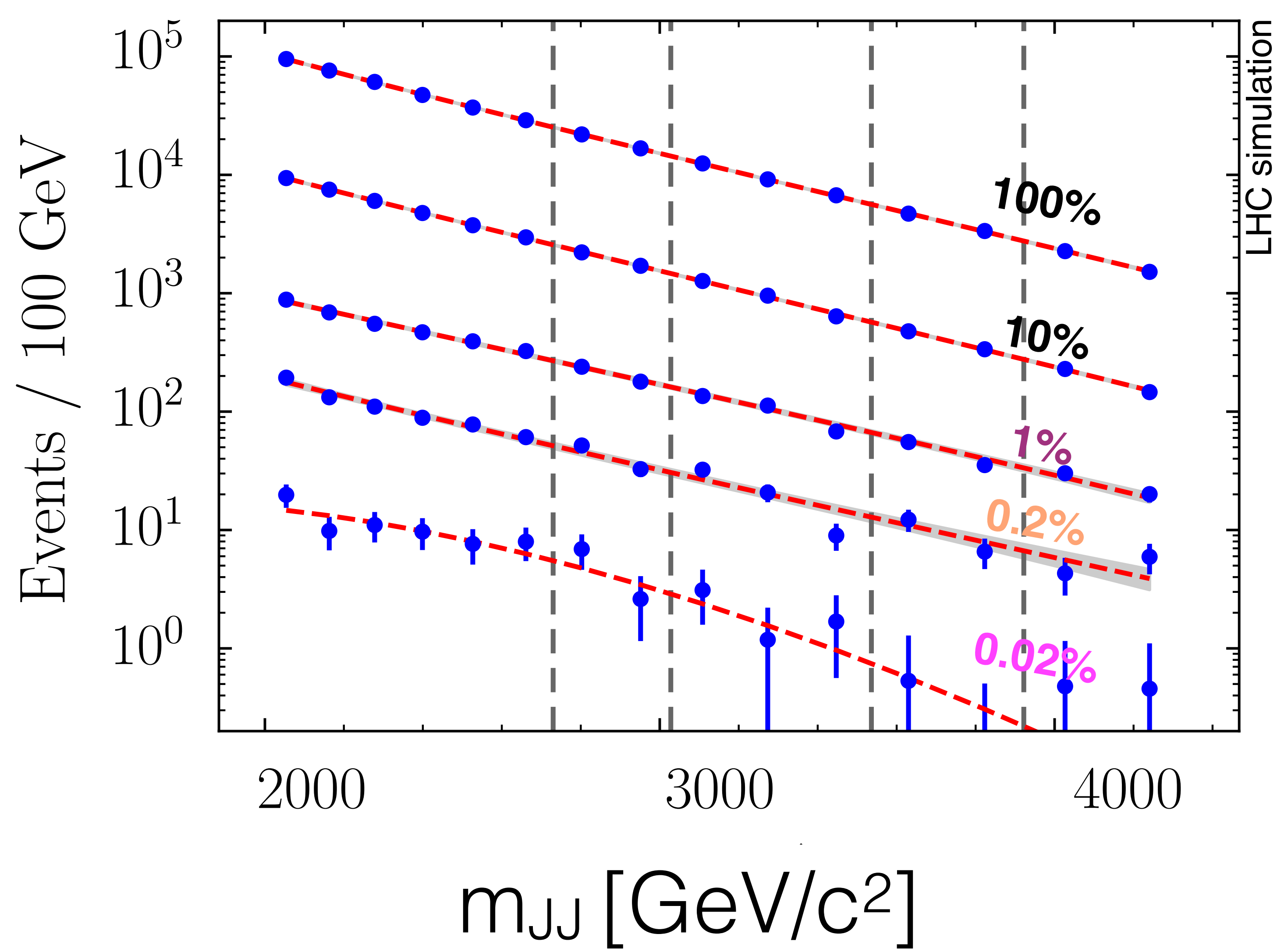
Example: two-“jet” search

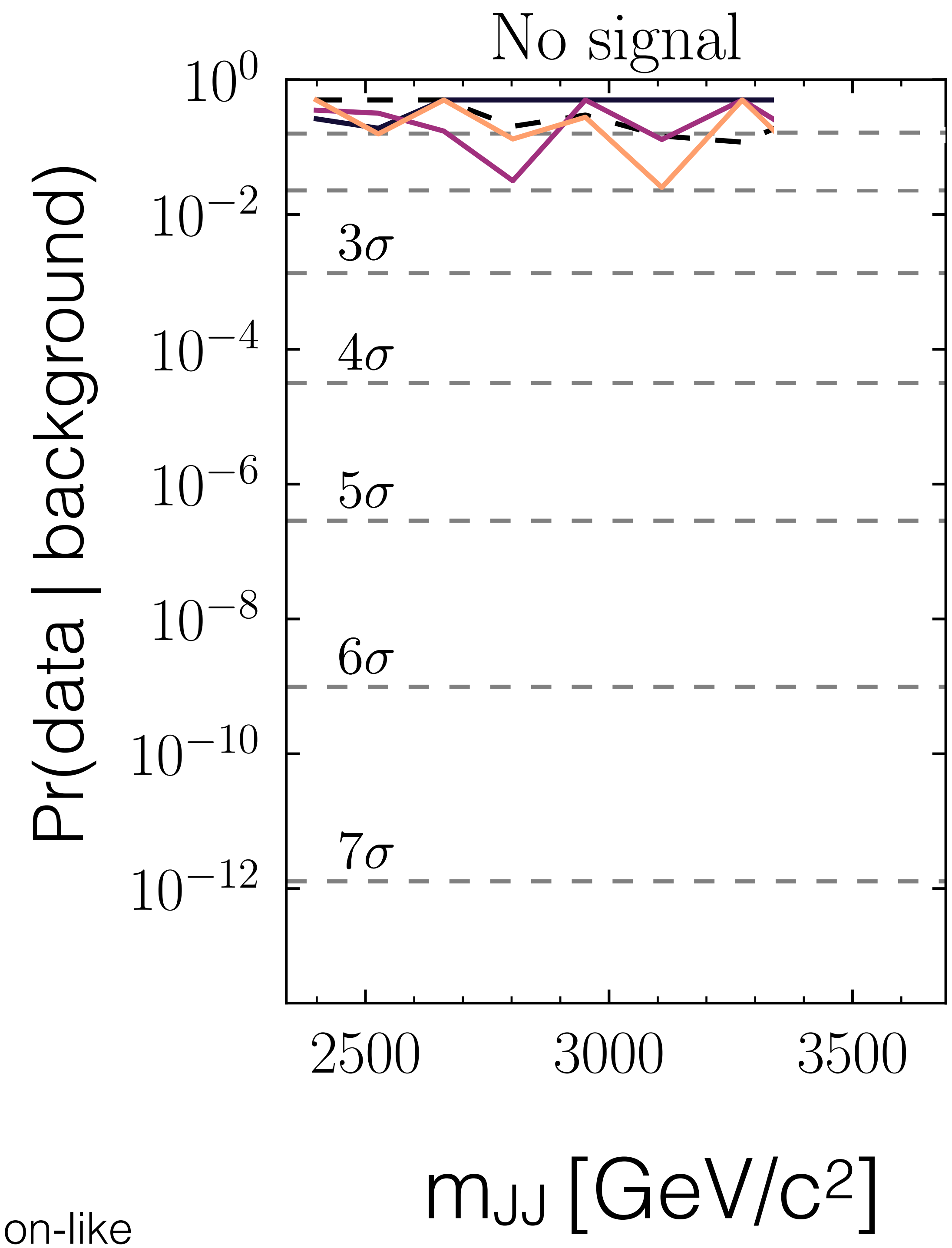


Example: two-“jet” search



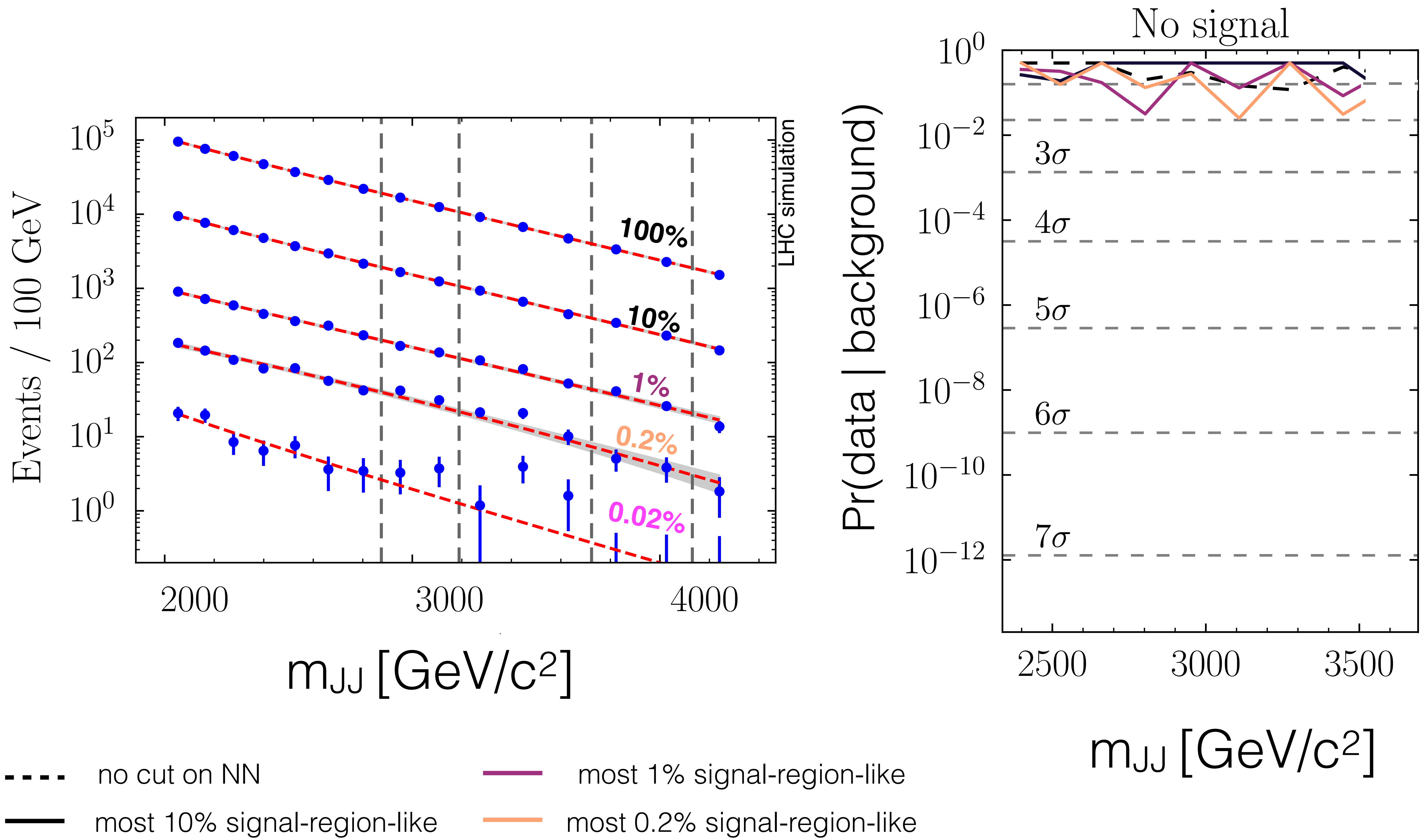
Example: two-“jet” search



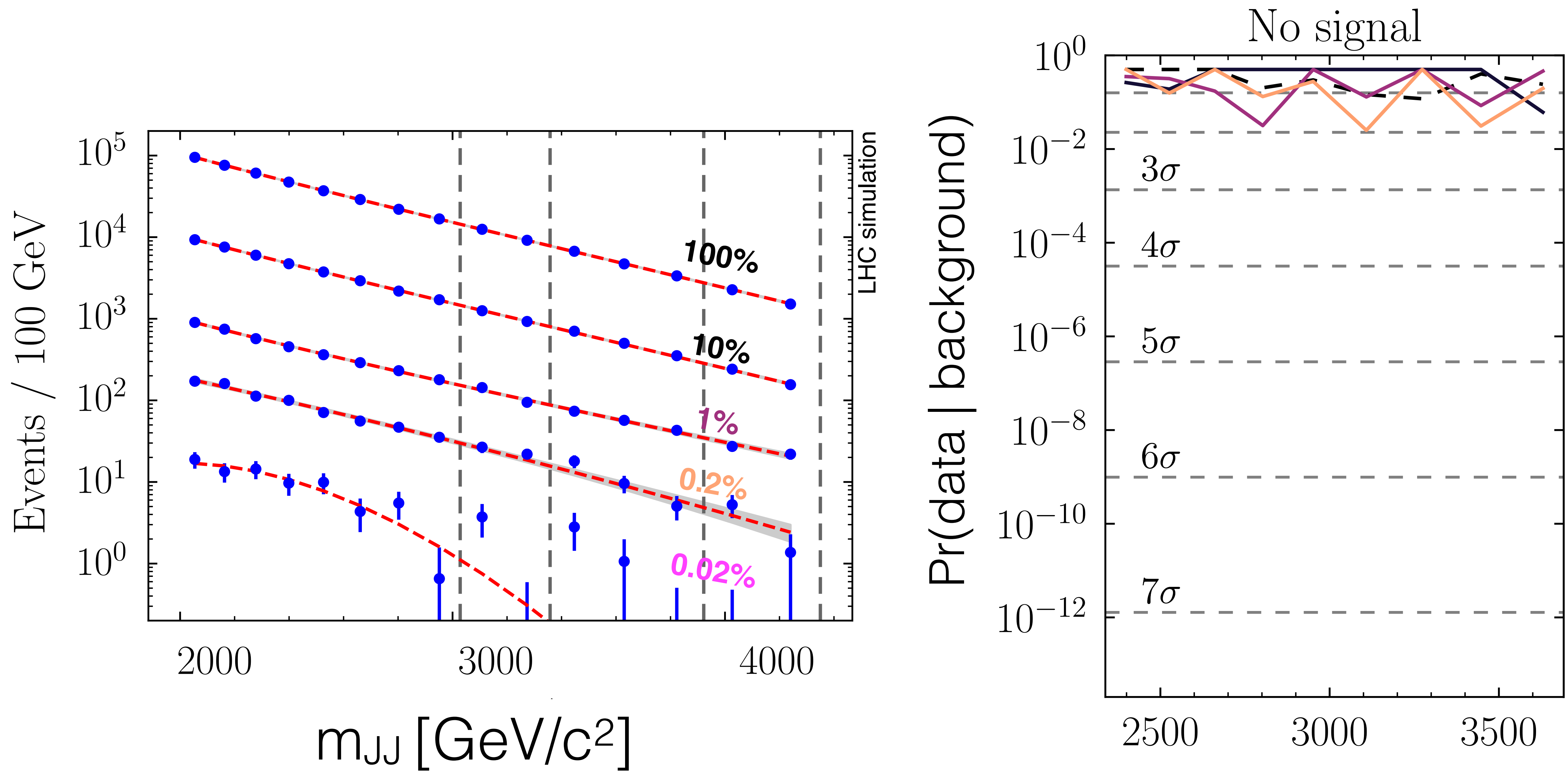


no cut on NN
most 10% signal-region-like
most 1% signal-region-like
most 0.2% signal-region-like

Example: two-“jet” search

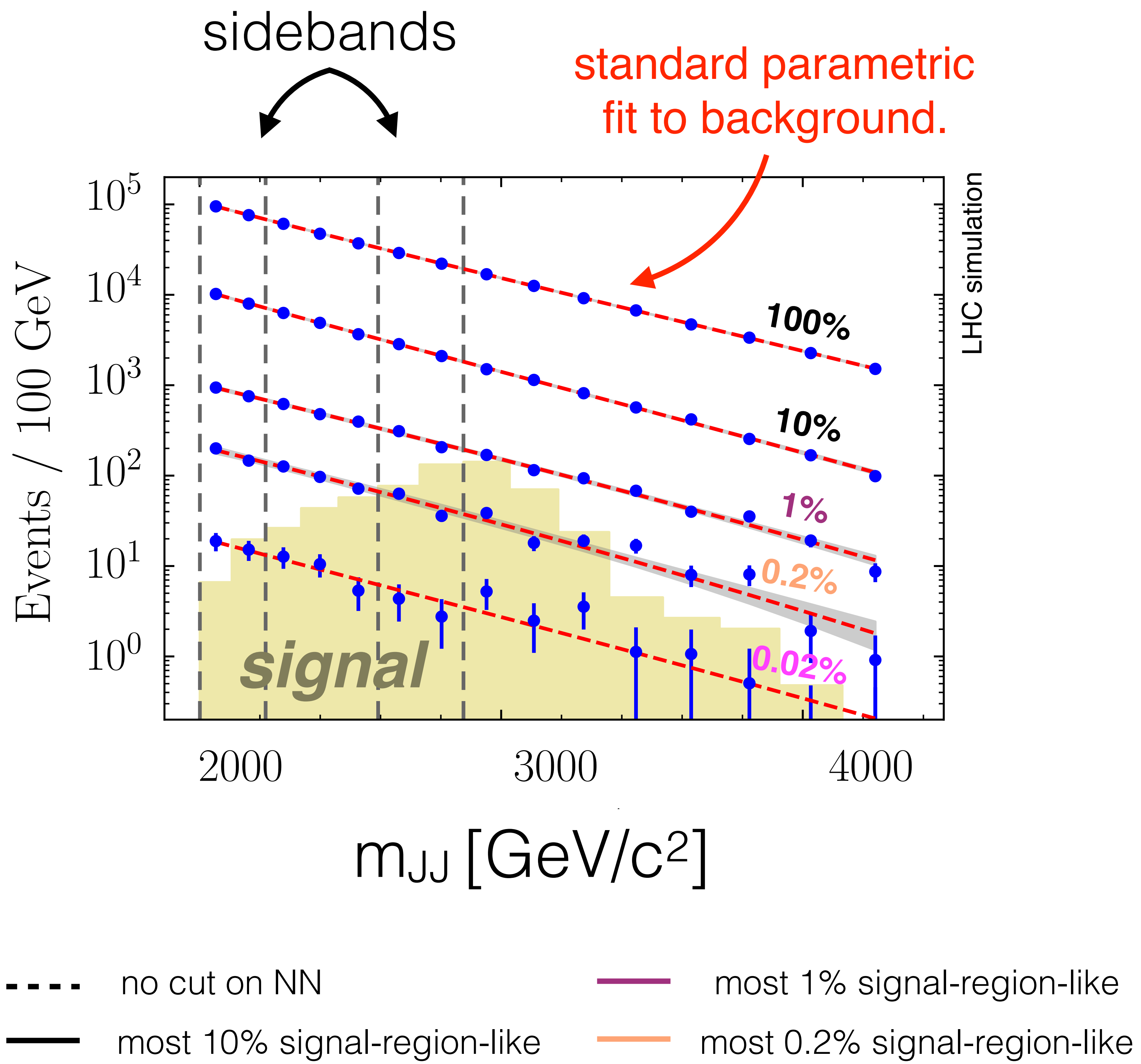


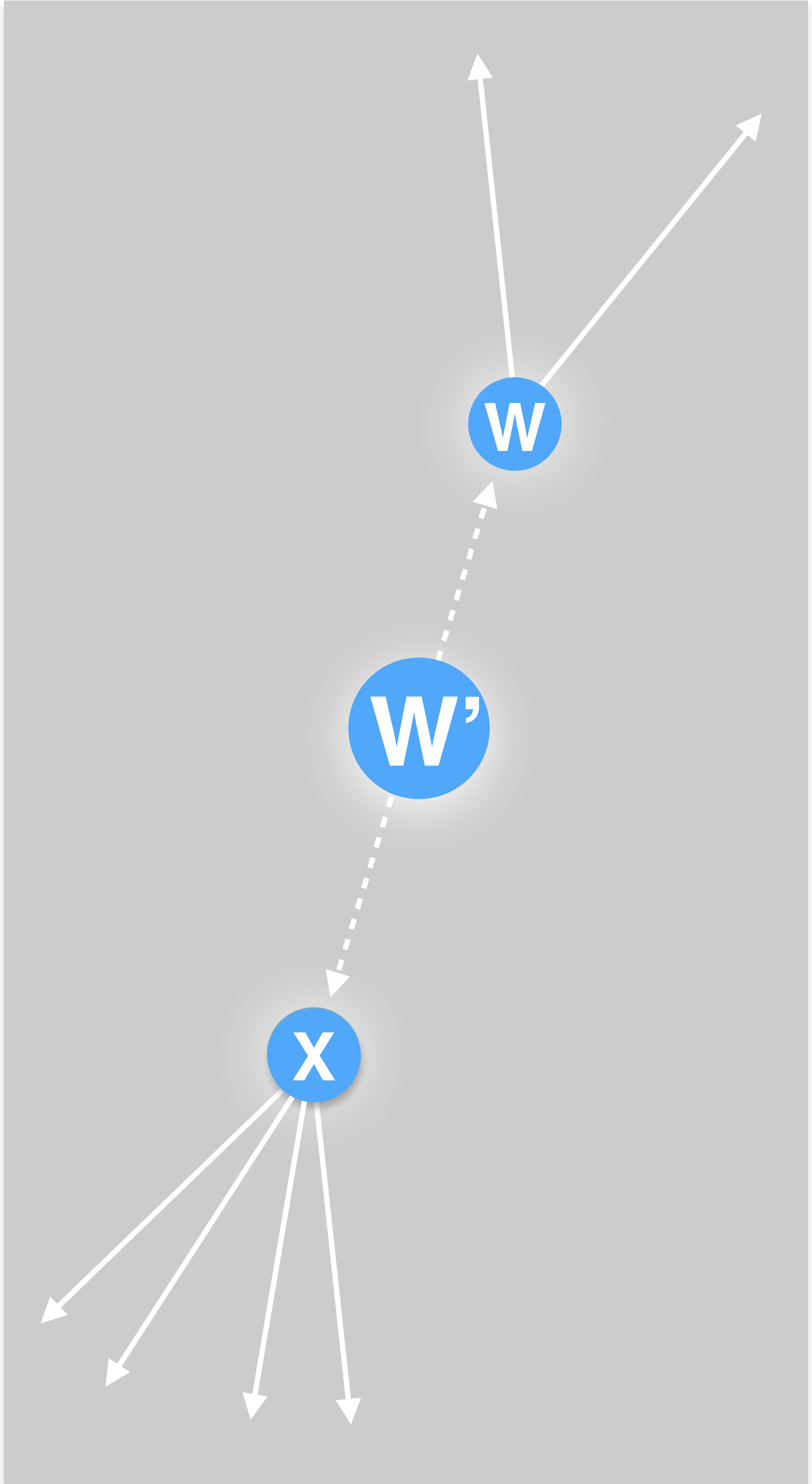
Example: two-“jet” search



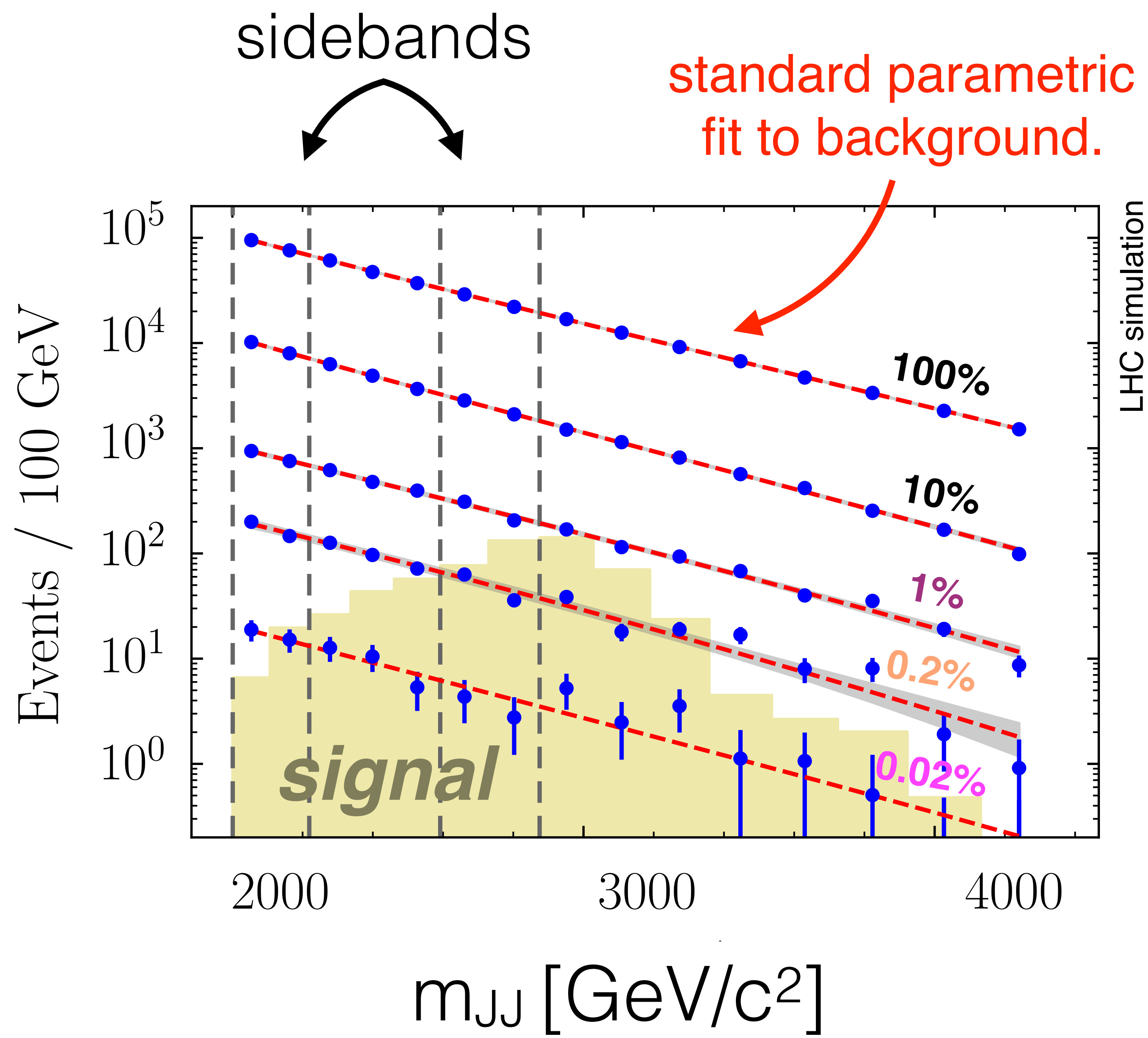
- - - no cut on NN
— most 10% signal-region-like
— most 1% signal-region-like
— most 0.2% signal-region-like

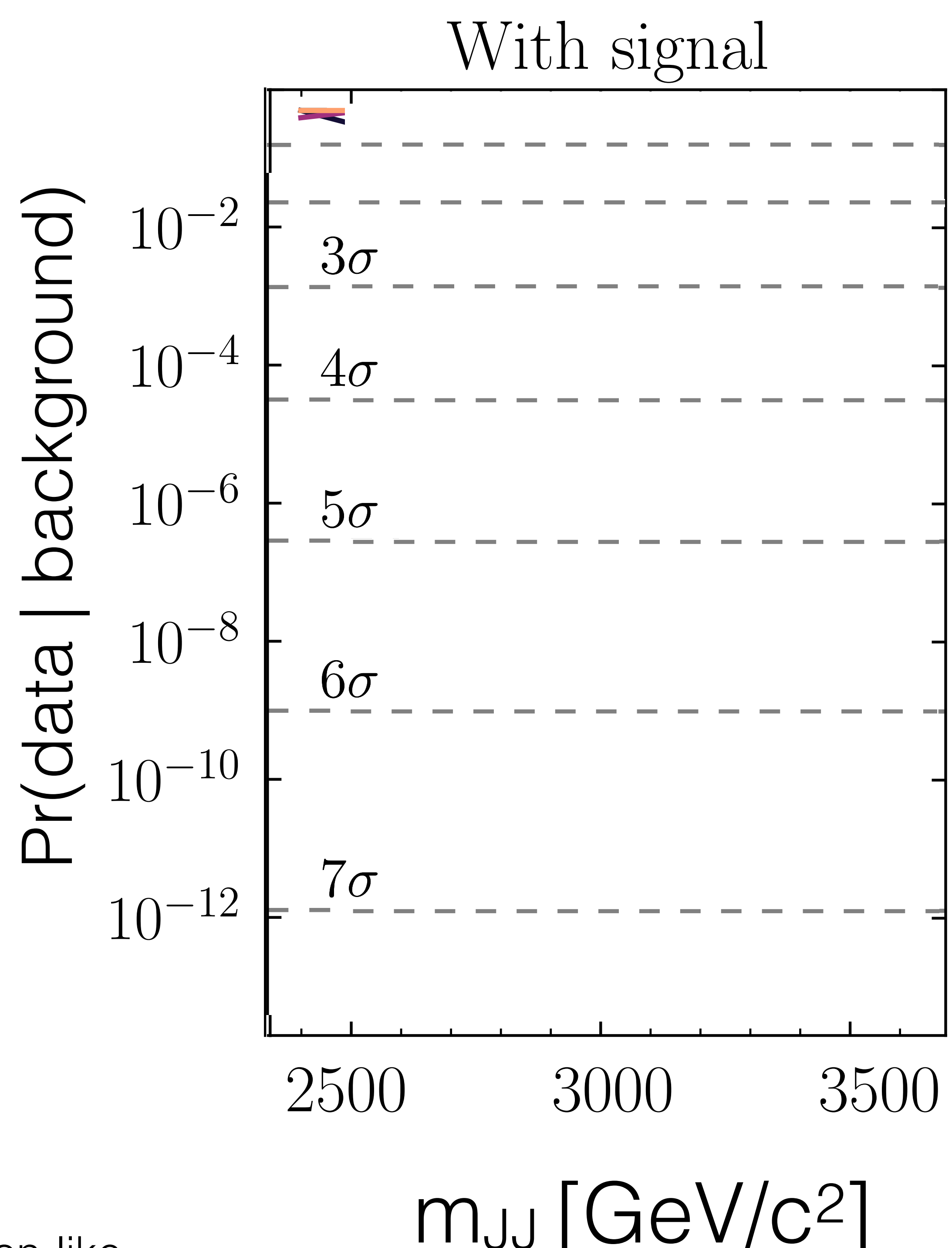
…and when there is a signal?





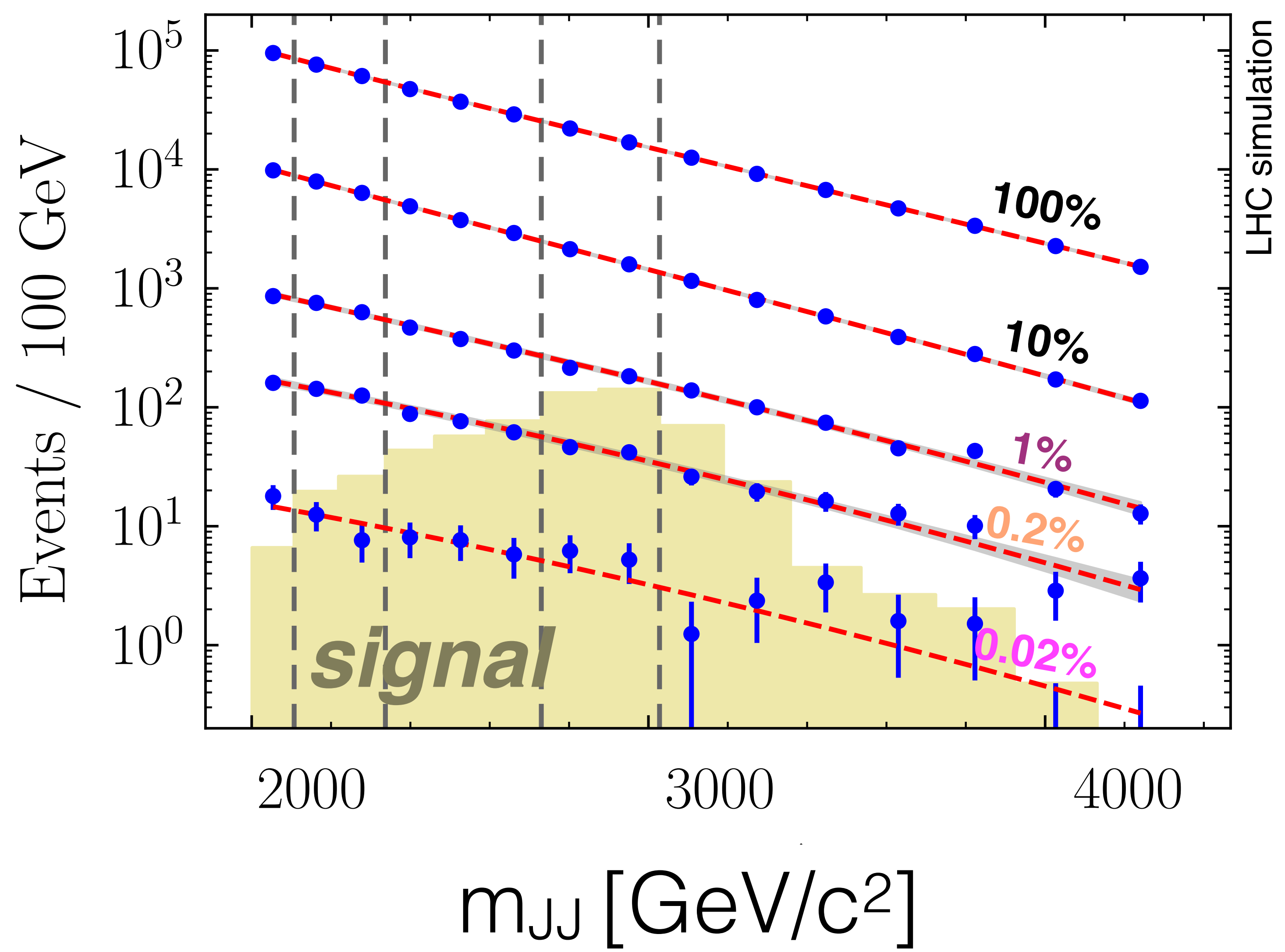
…and when there is a signal?

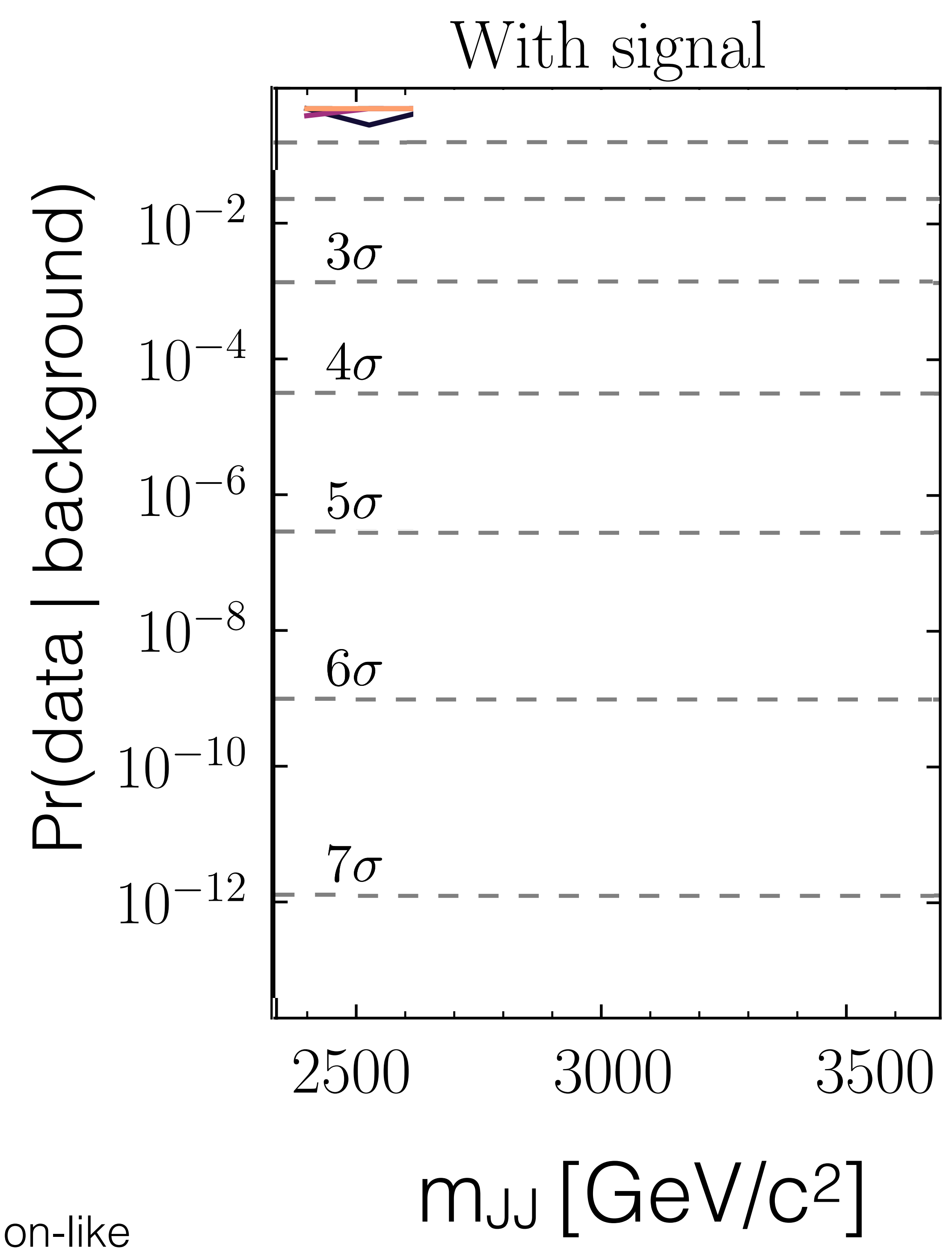




- - - - no cut on NN
— most 10% signal-region-like
— most 1% signal-region-like
— most 0.2% signal-region-like

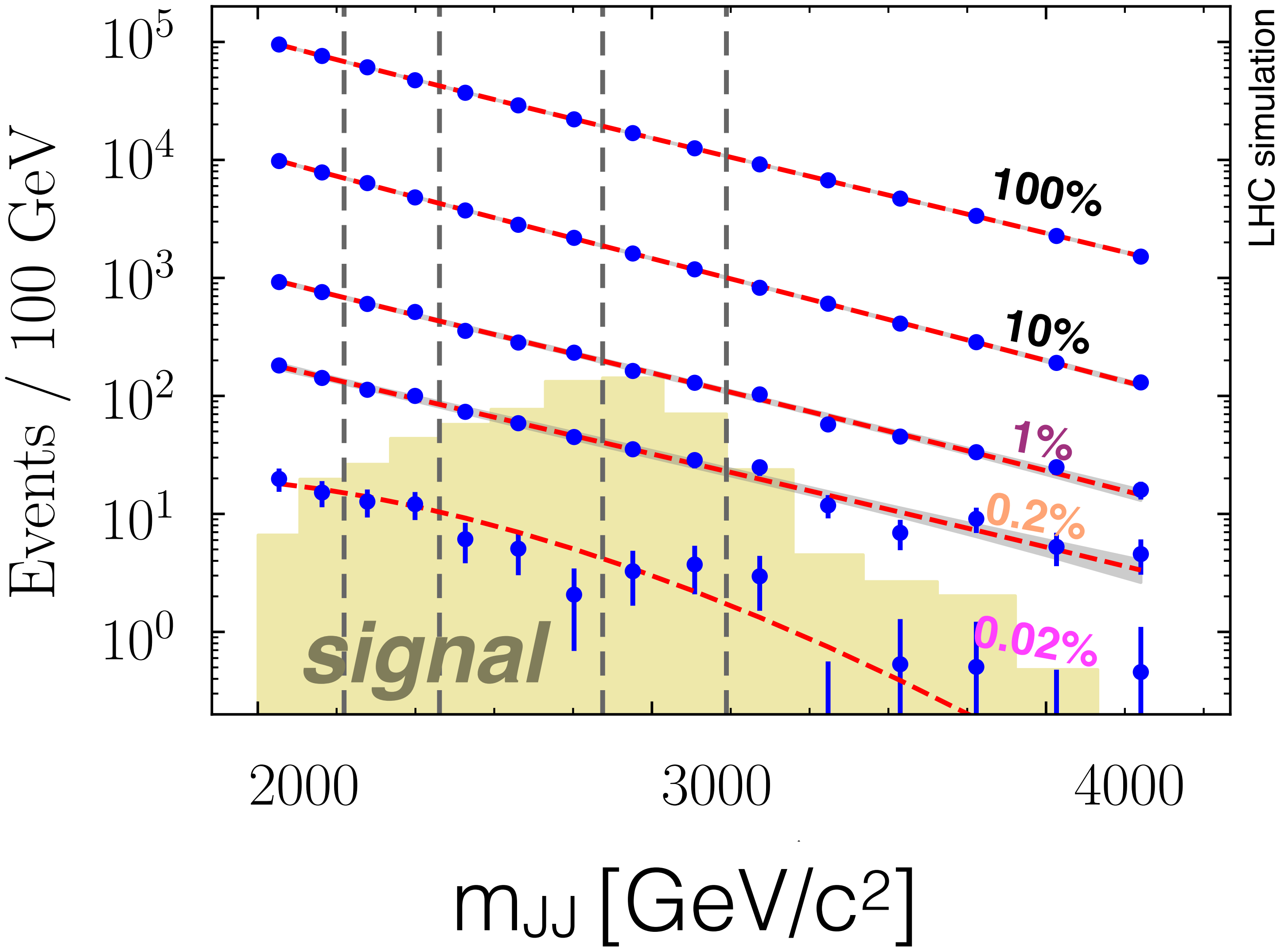
…and when there is a signal?

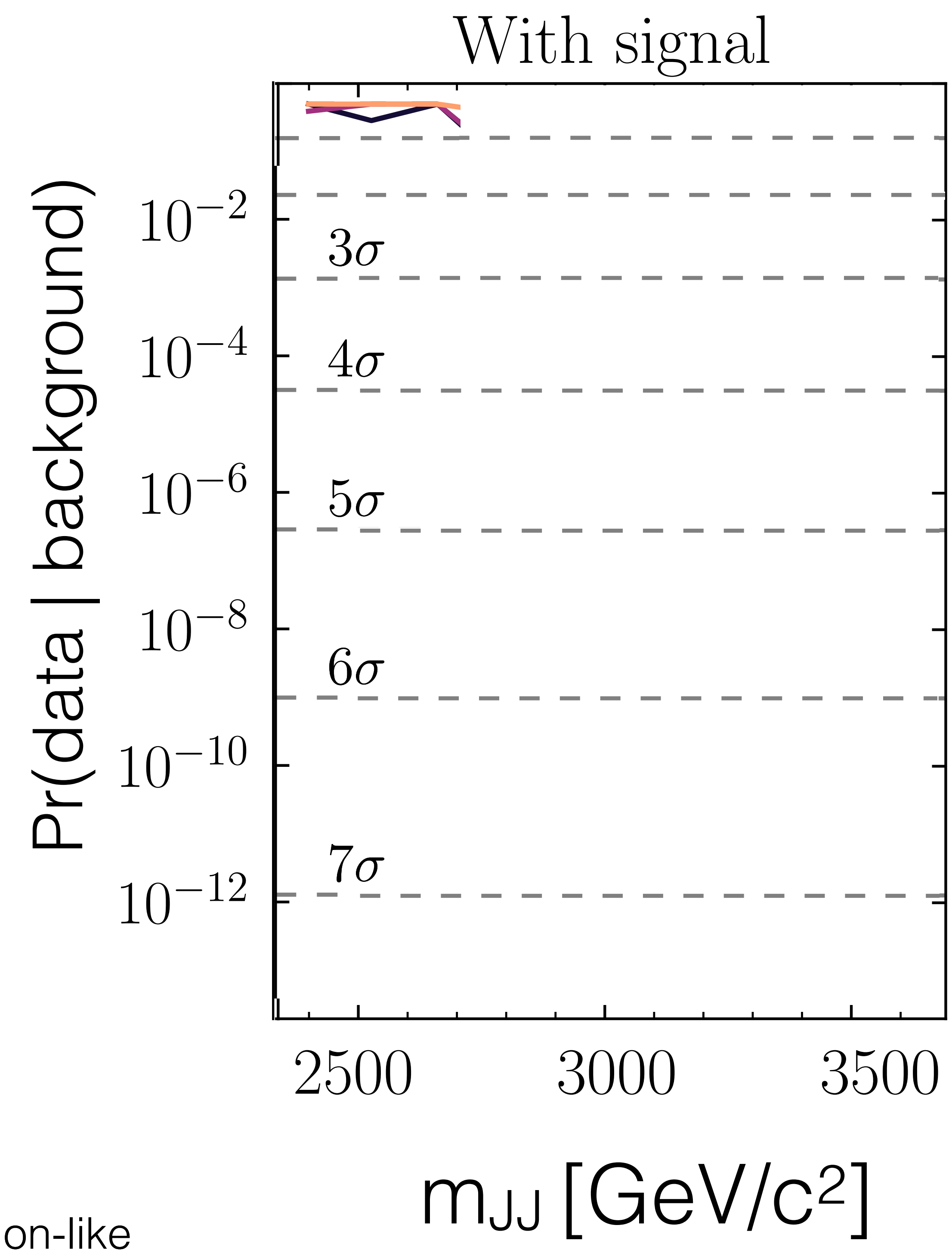




- - - no cut on NN
— most 10% signal-region-like
— most 1% signal-region-like
— most 0.2% signal-region-like

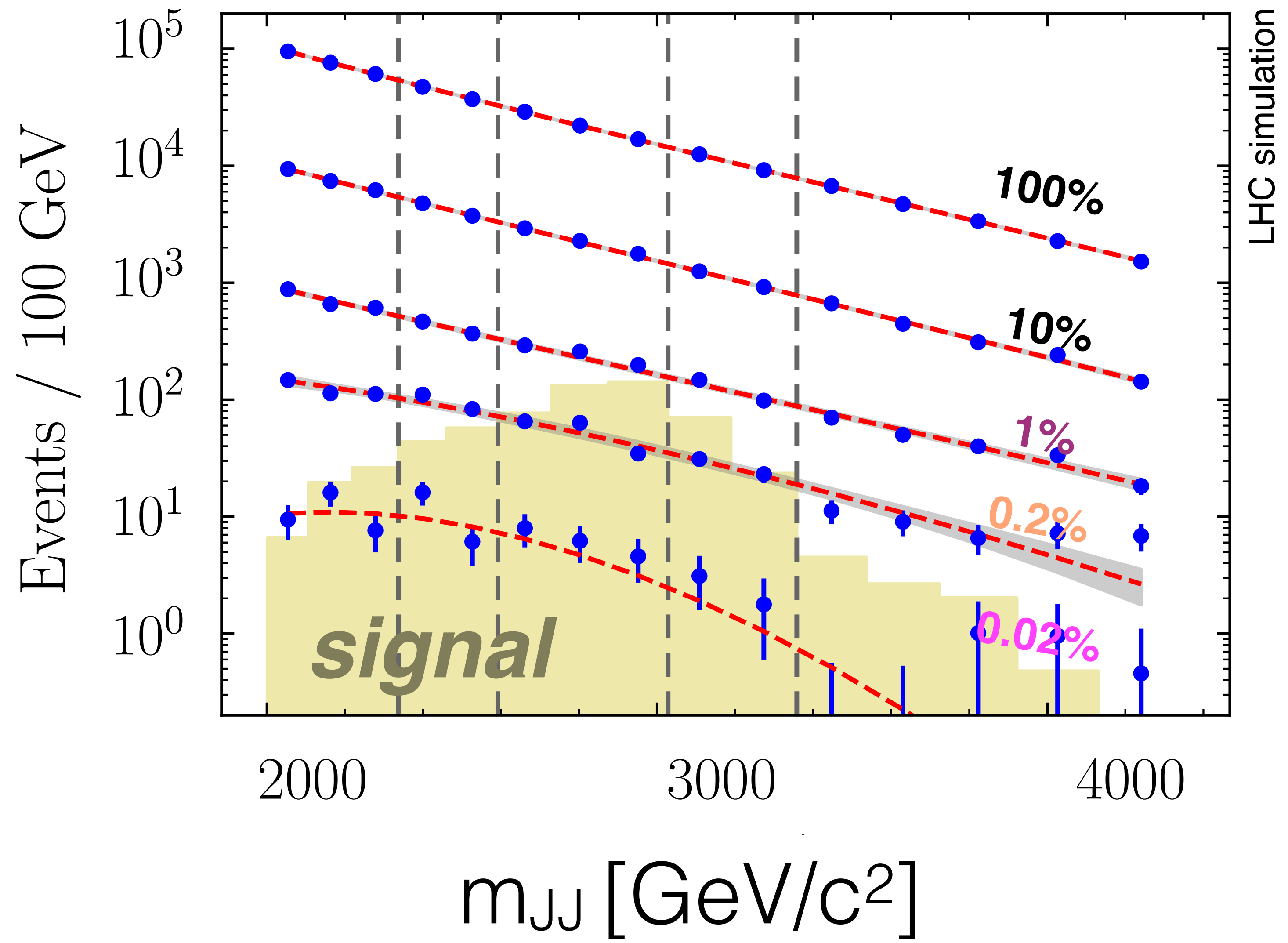
...and when there is a signal?

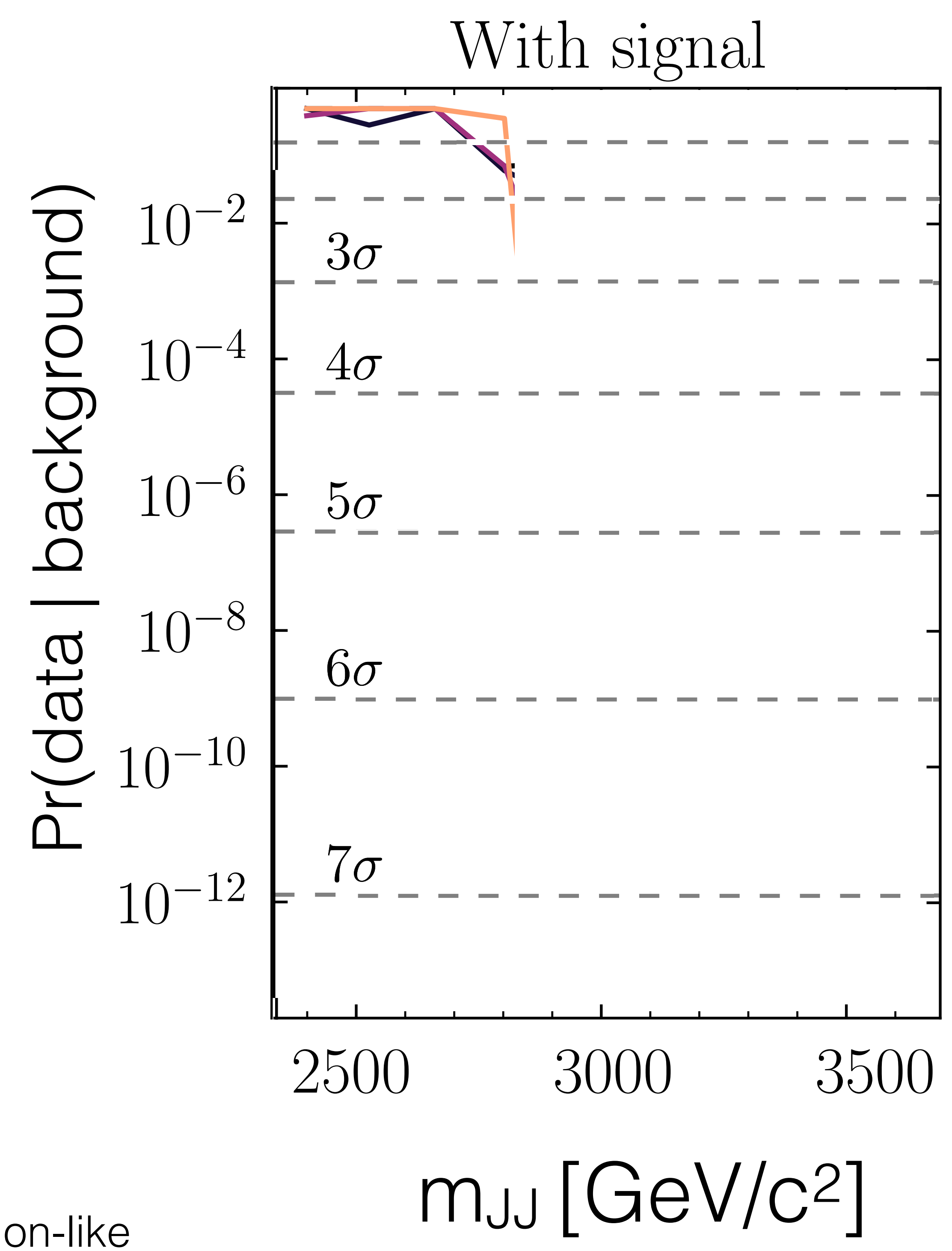




no cut on NN
most 10% signal-region-like
most 1% signal-region-like
most 0.2% signal-region-like

…and when there is a signal?





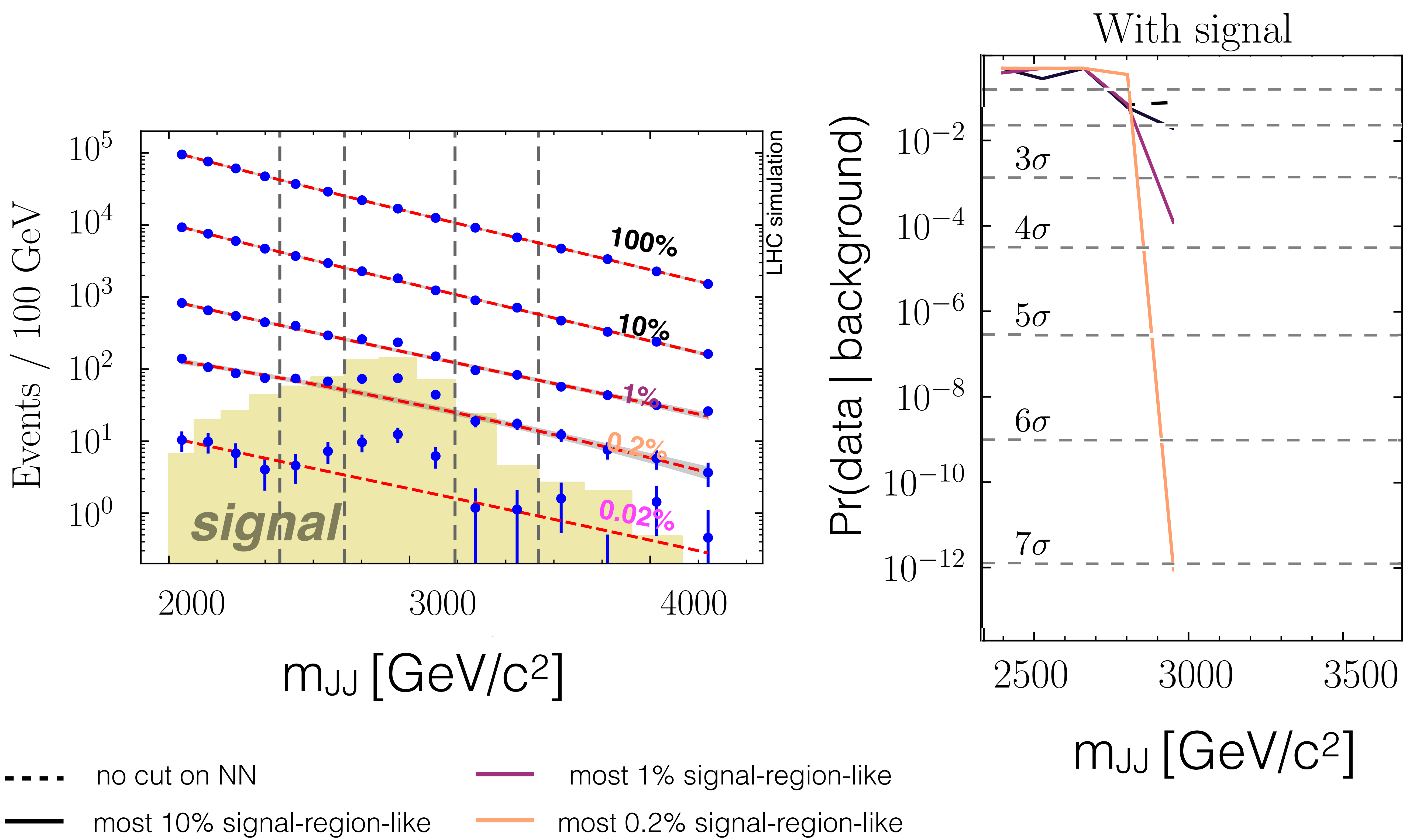
no cut on NN

most 10% signal-region-like

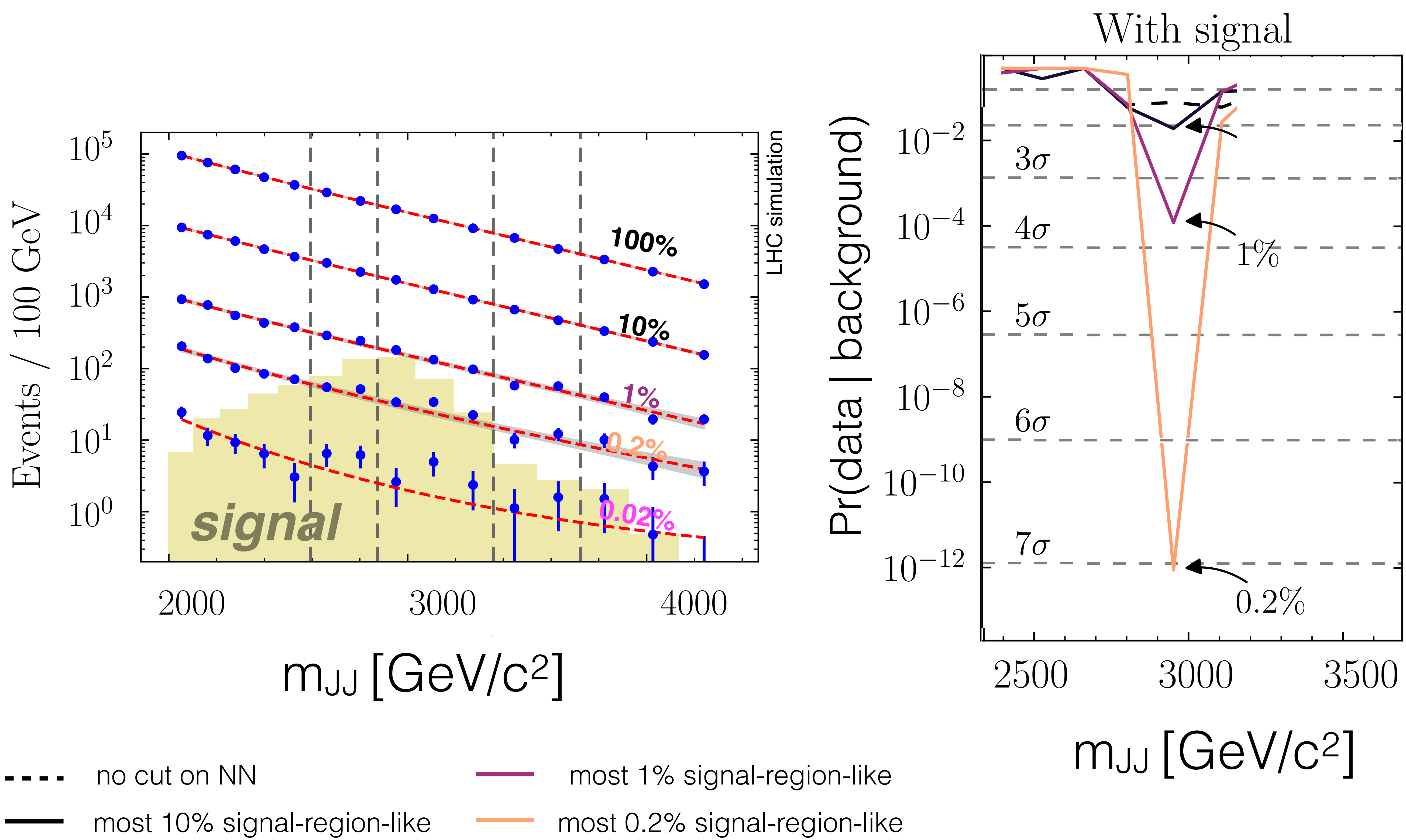
most 1% signal-region-like

most 0.2% signal-region-like

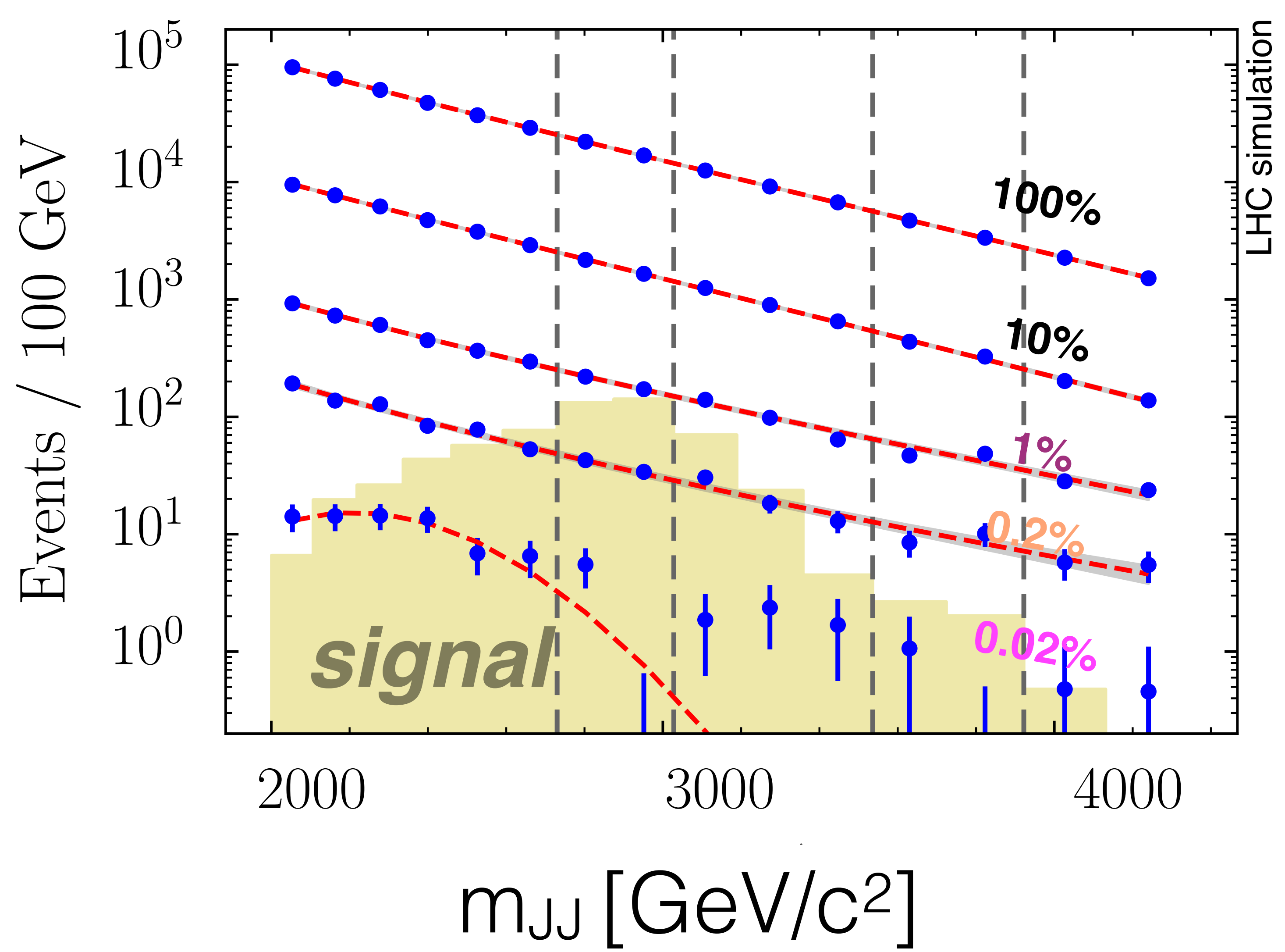
…and when there is a signal?

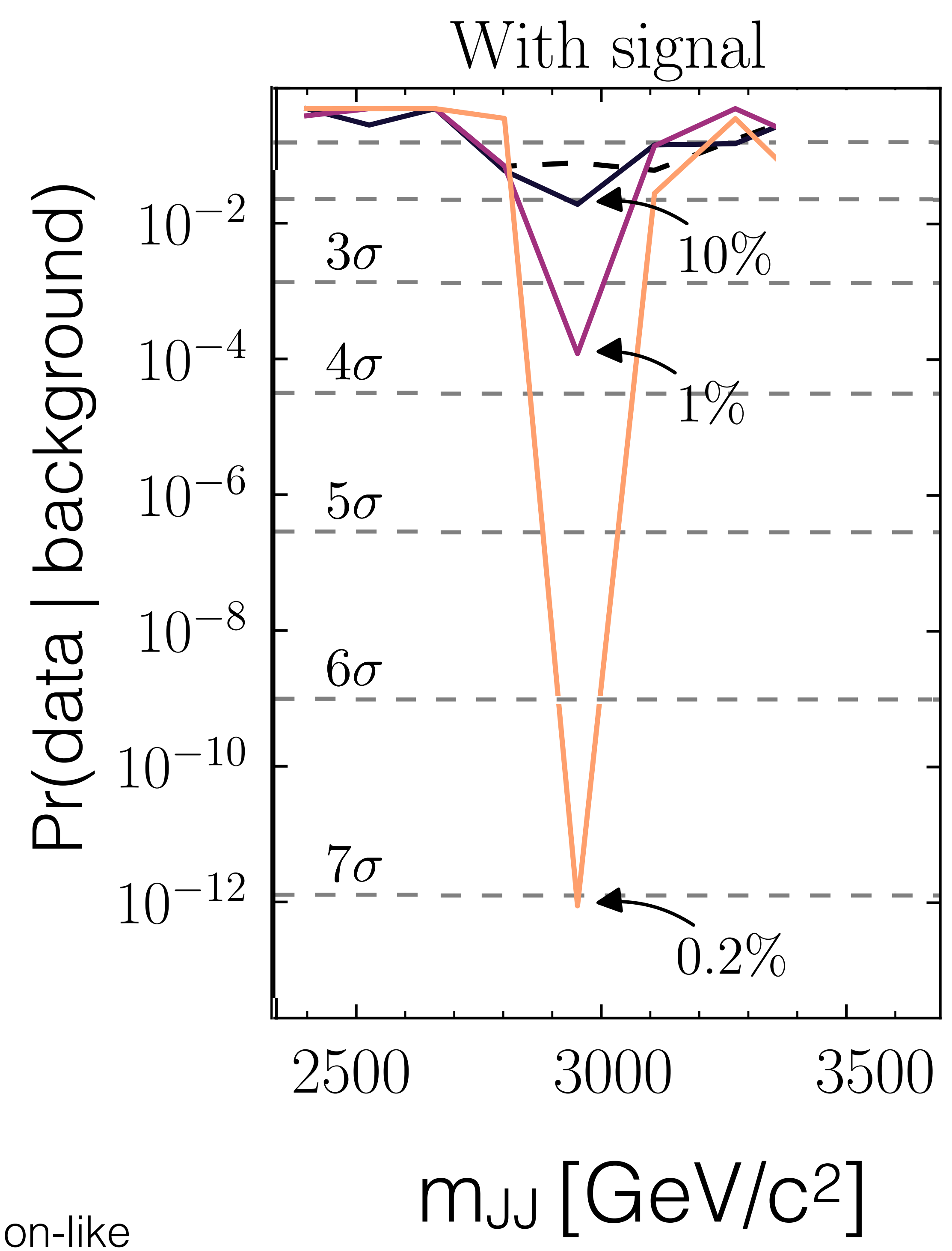


…and when there is a signal?



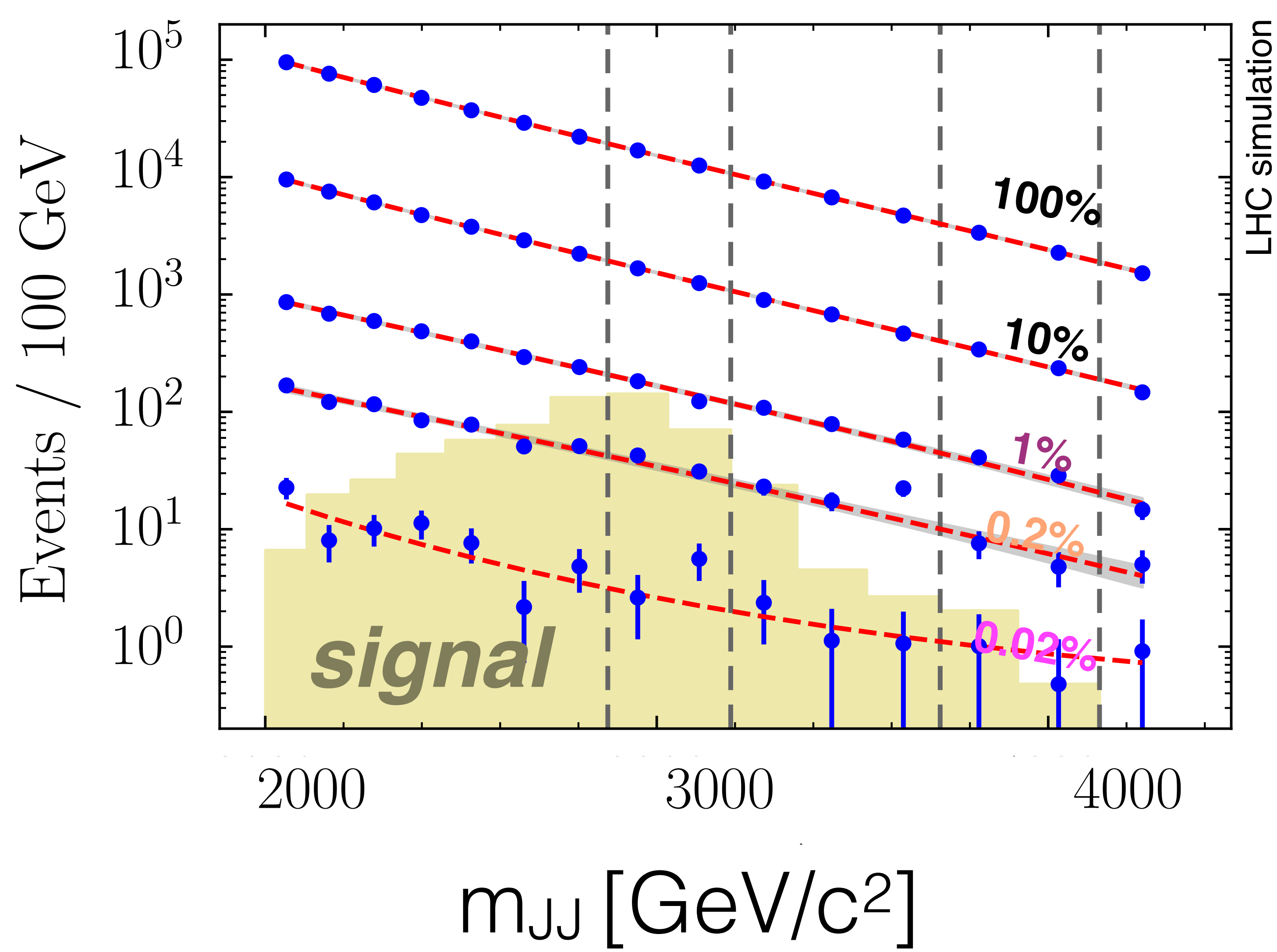
…and when there is a signal?

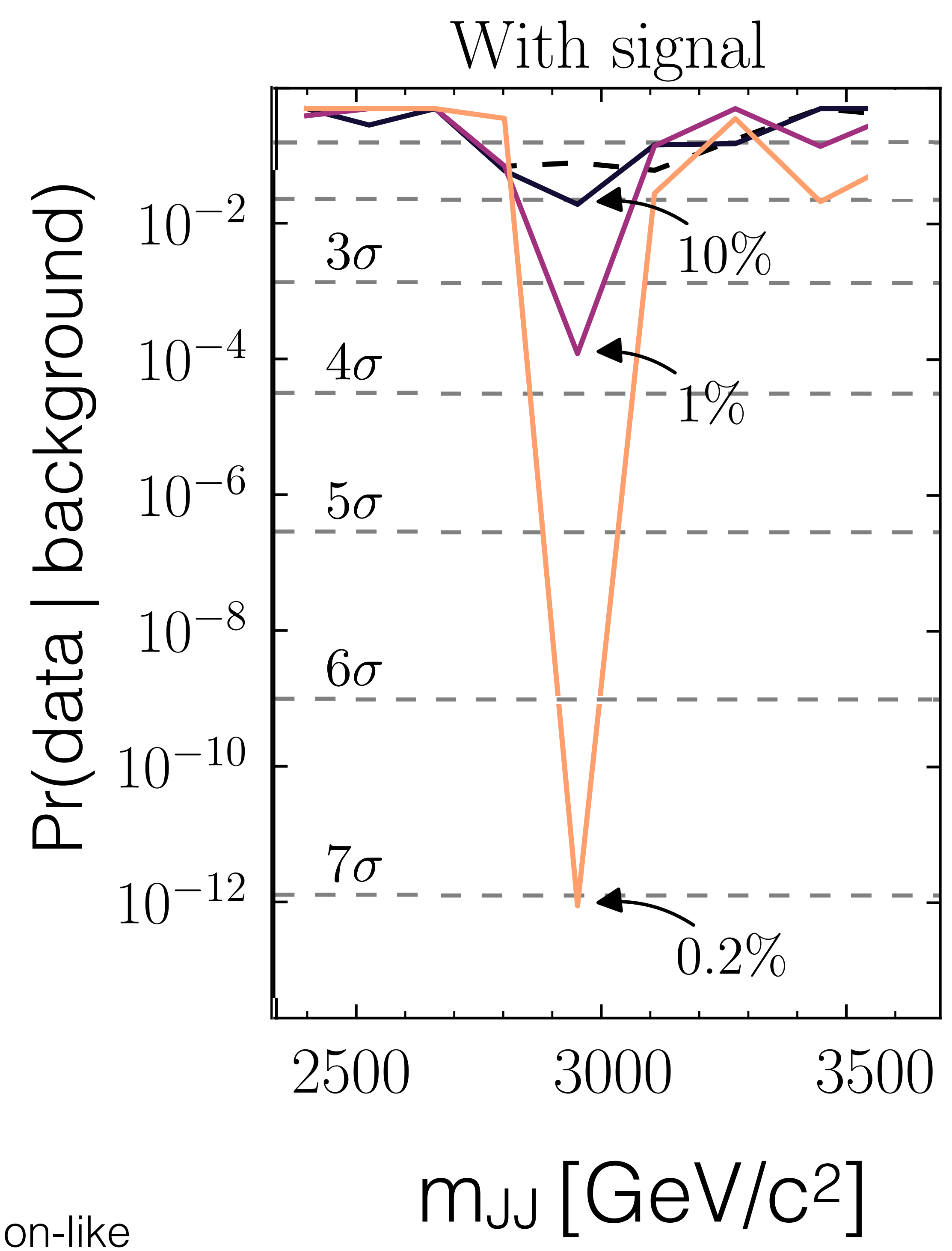




no cut on NN
most 10% signal-region-like
most 1% signal-region-like
most 0.2% signal-region-like

…and when there is a signal?





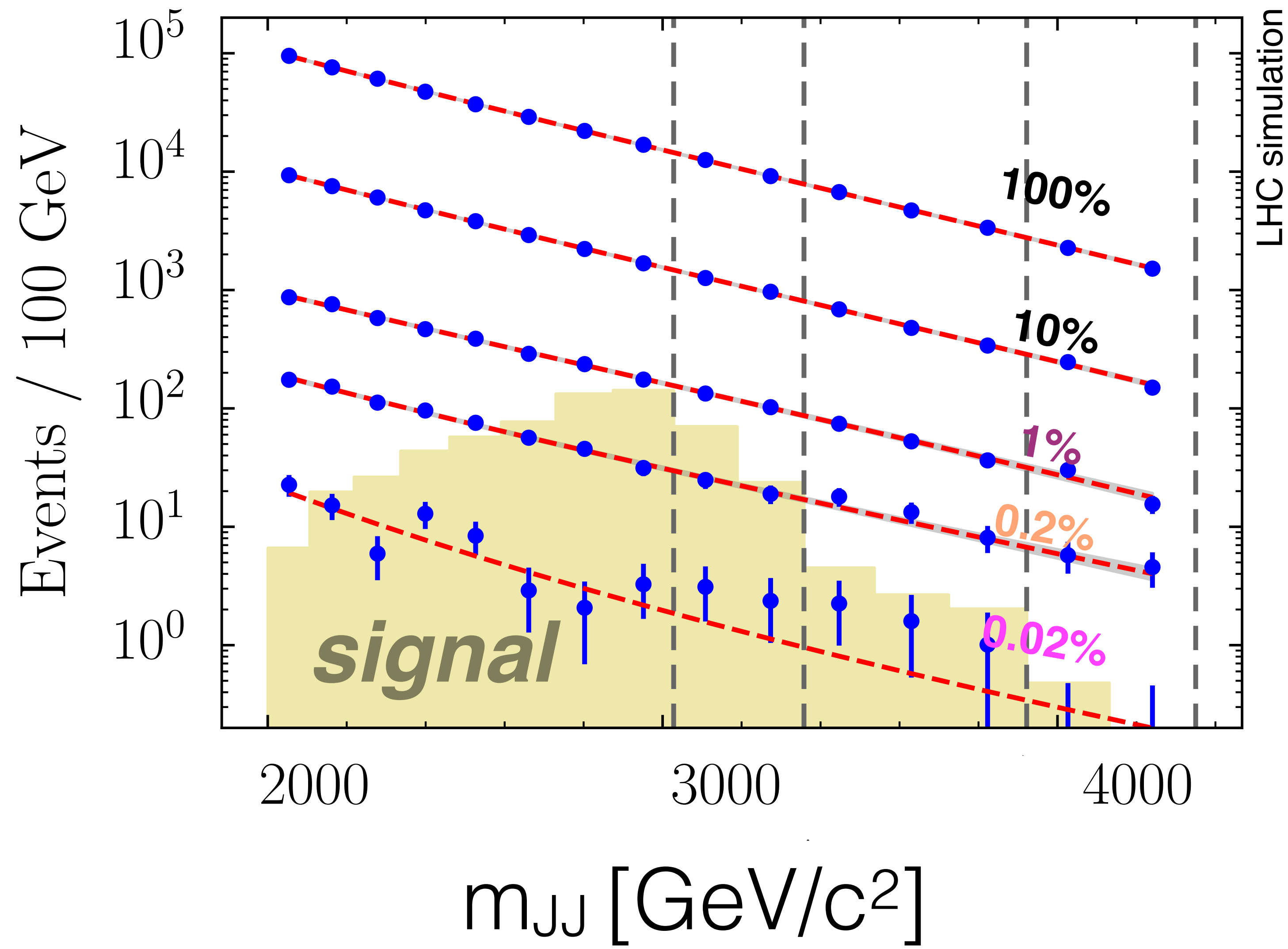
no cut on NN

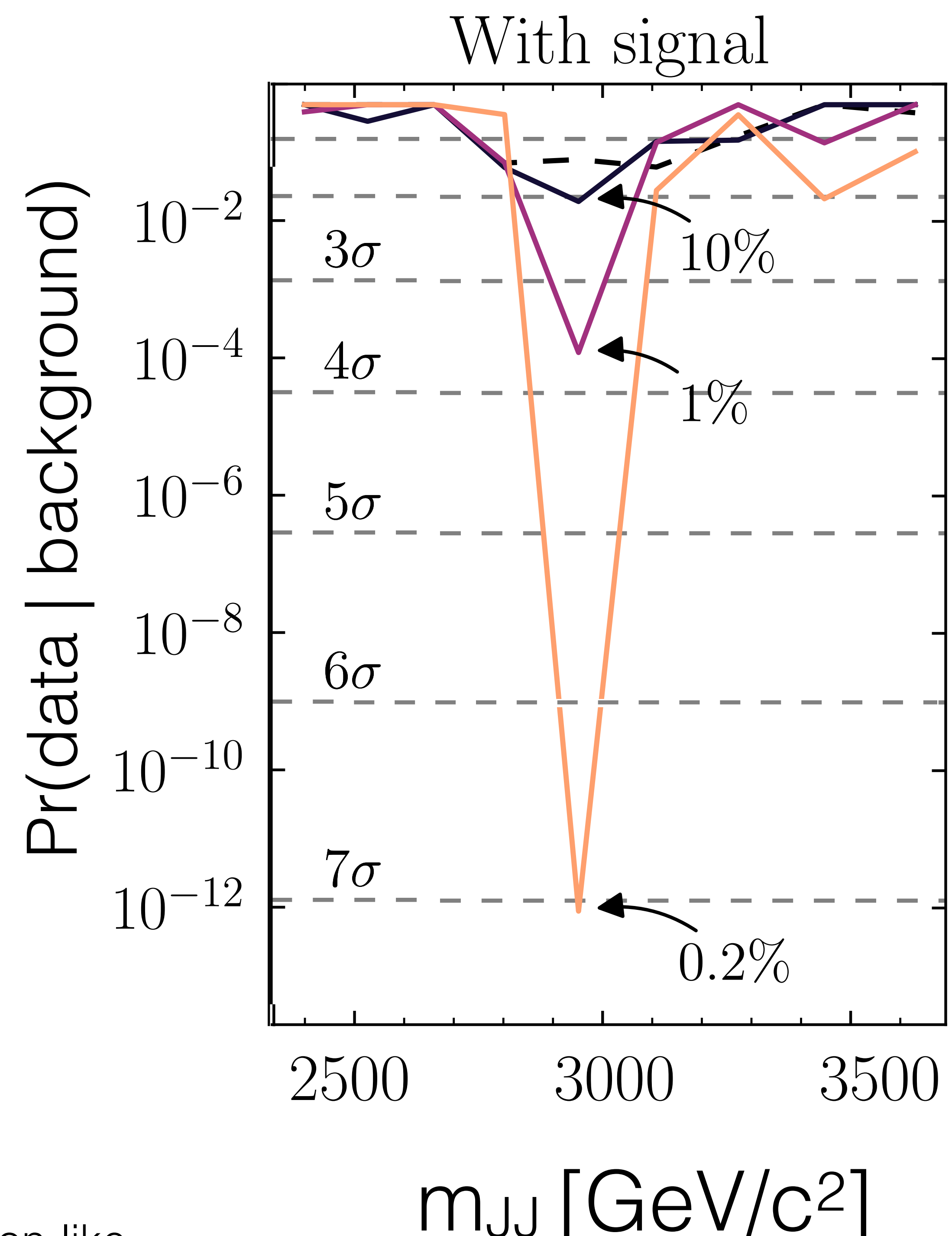
most 10% signal-region-like

most 1% signal-region-like

most 0.2% signal-region-like

…and when there is a signal?

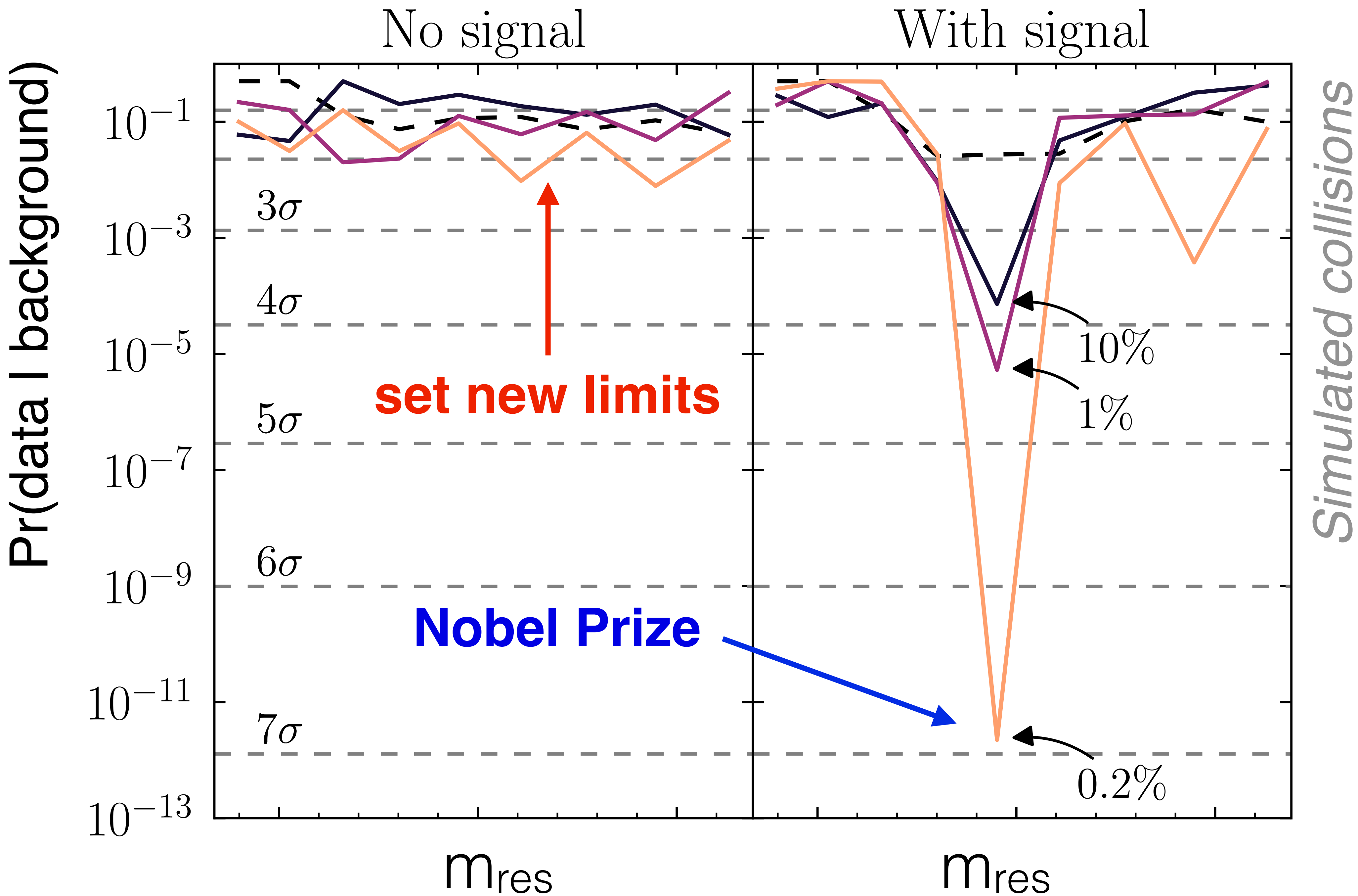




- - - - no cut on NN
—— most 10% signal-region-like
—— most 1% signal-region-like
—— most 0.2% signal-region-like

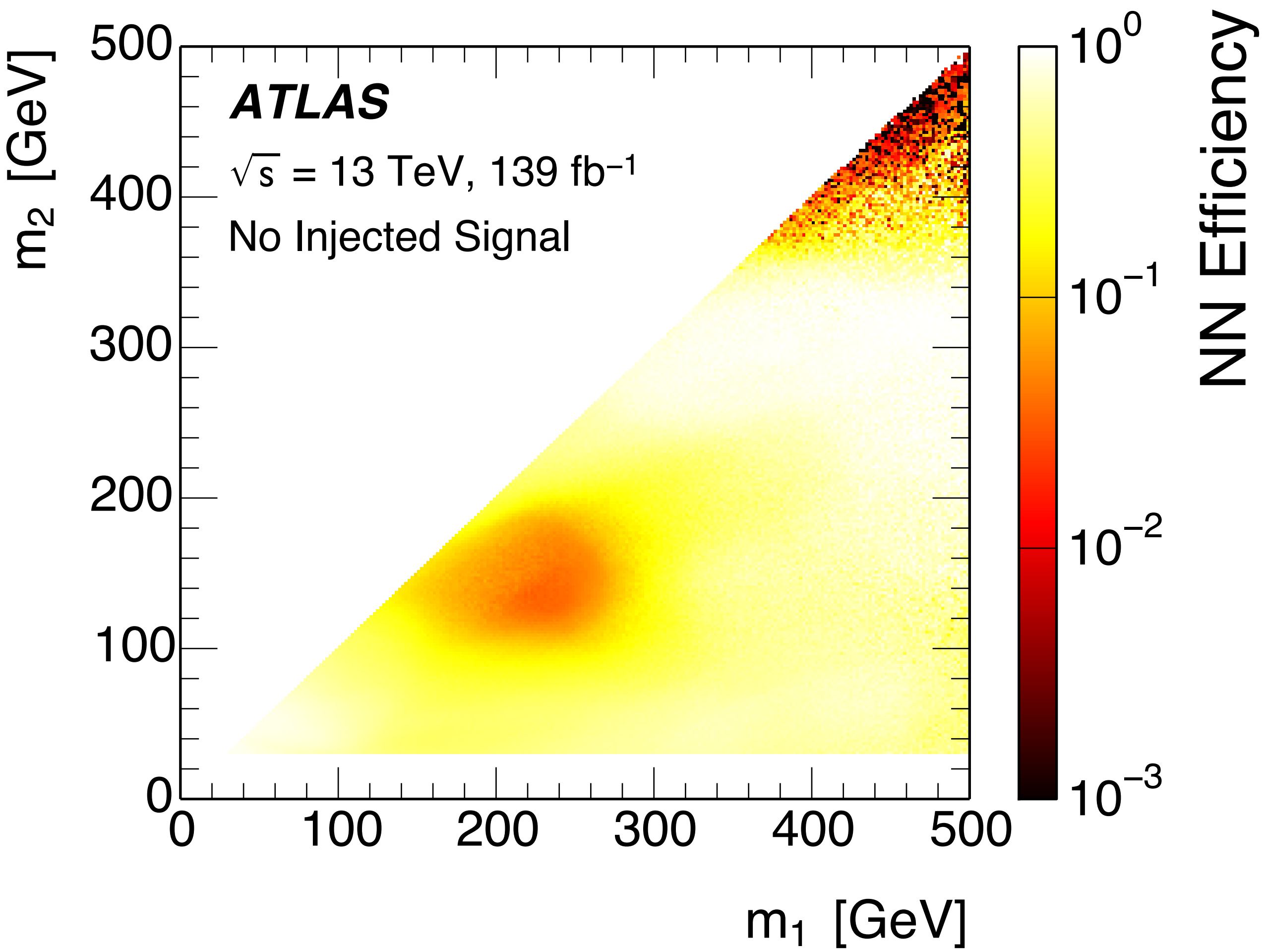
Anomaly detection: Overview

J. Collins, K. Howe, BPN, Phys. Rev. Lett. 121 (2018) 241803, 1805.02664



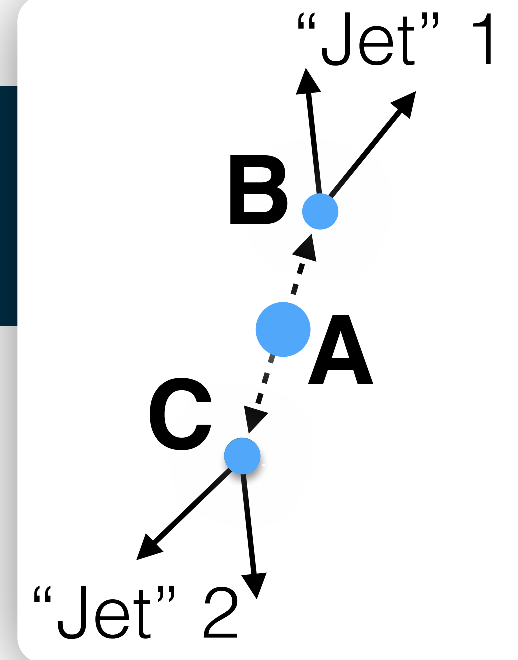
Collision data results New

ATLAS Collaboration, 2005.02983
Analysis Team: A. Cukeriman, BPN

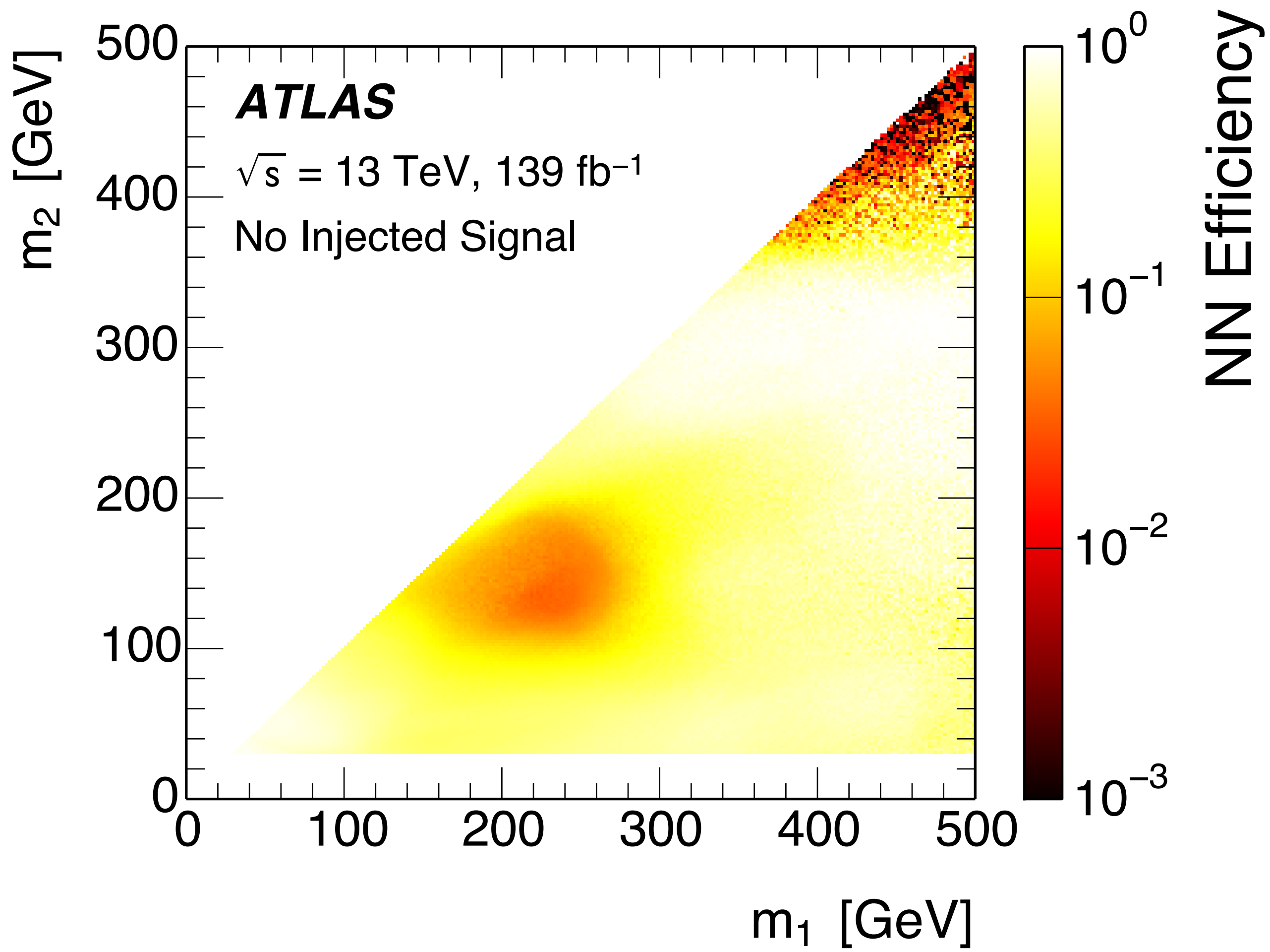


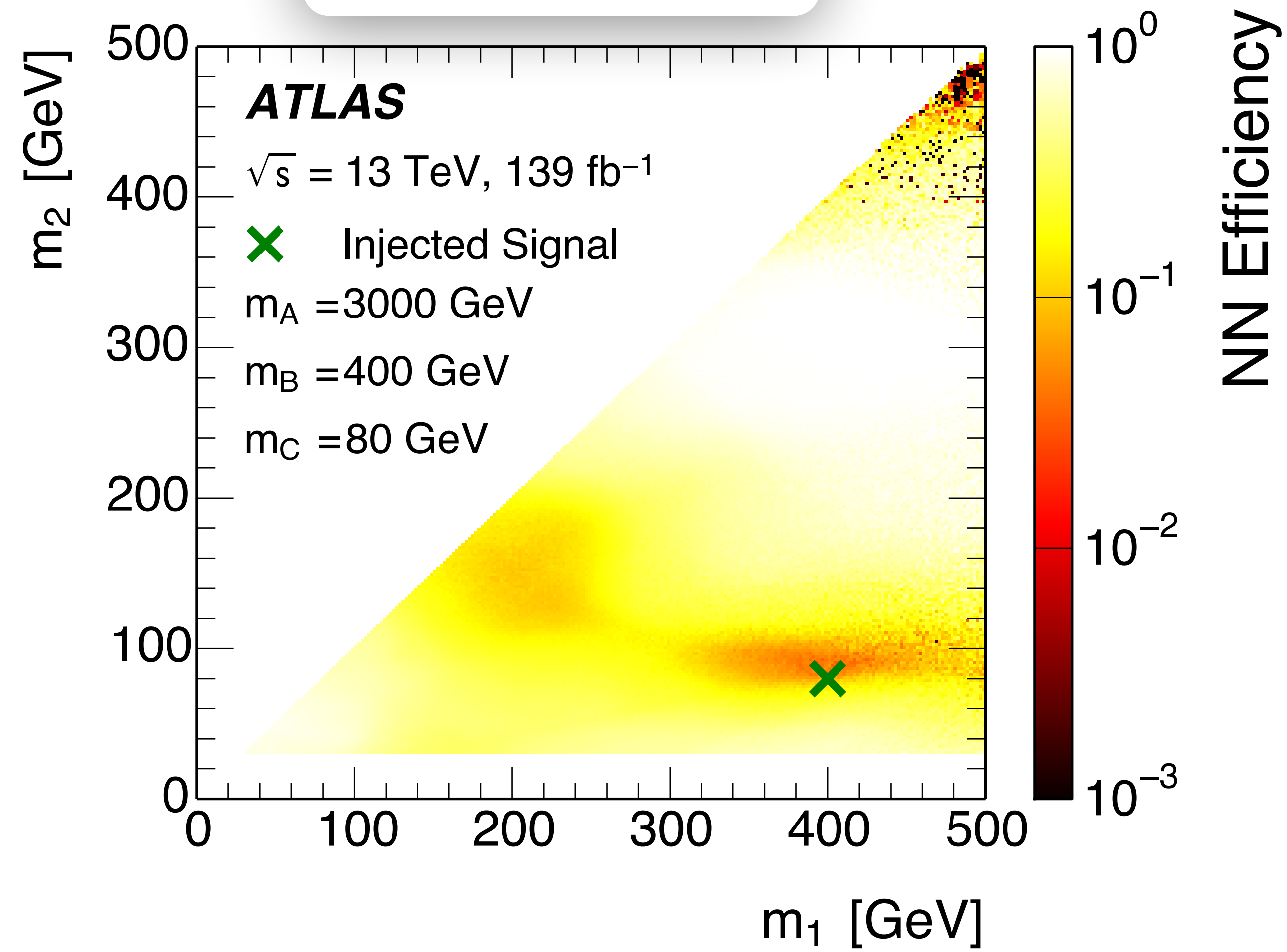
First round, keep it simple: feature space is 2D (jet masses)

Collision data results **New**



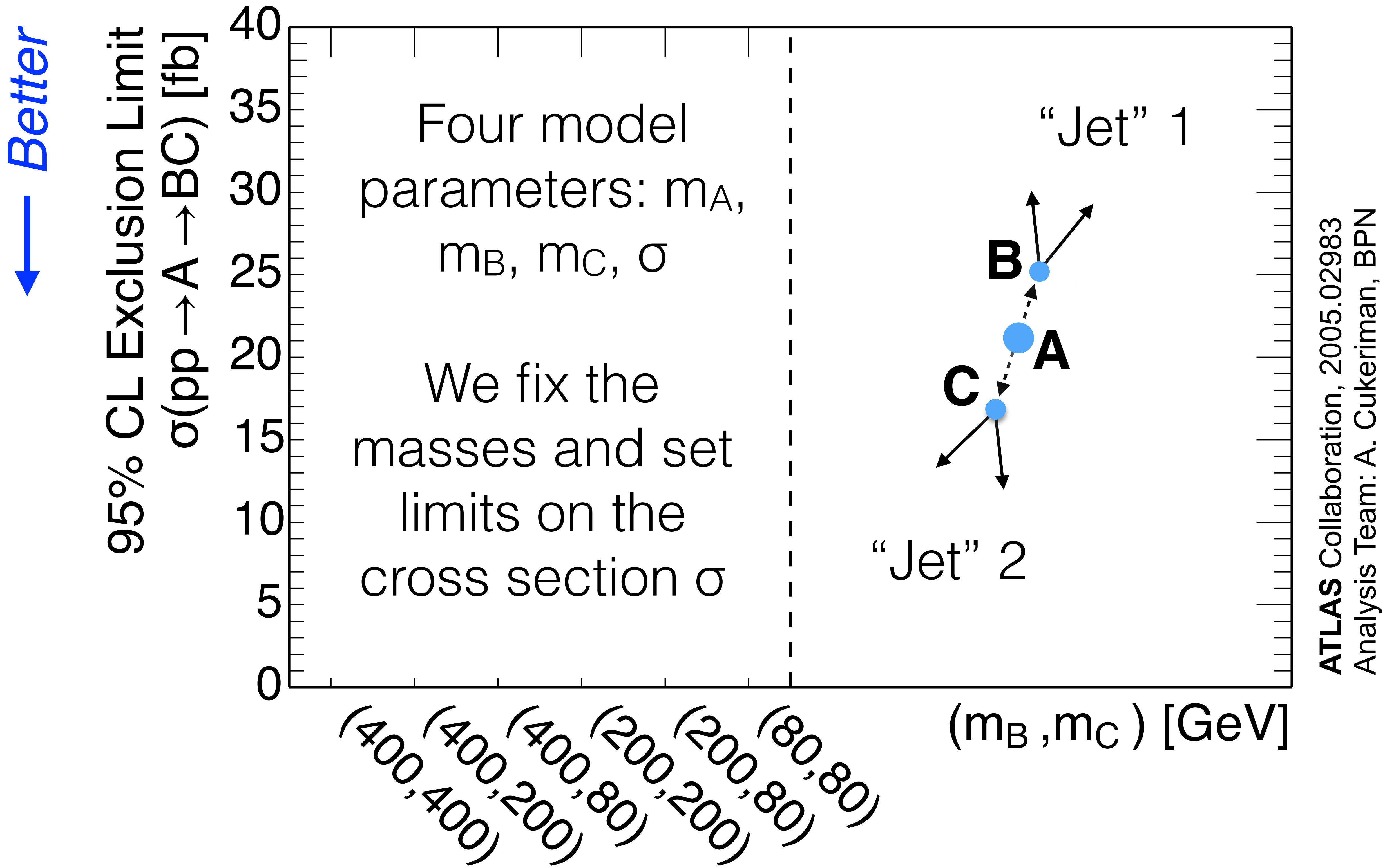
ATLAS Collaboration, 2005.02983
Analysis Team: A. Cukeriman, BPN



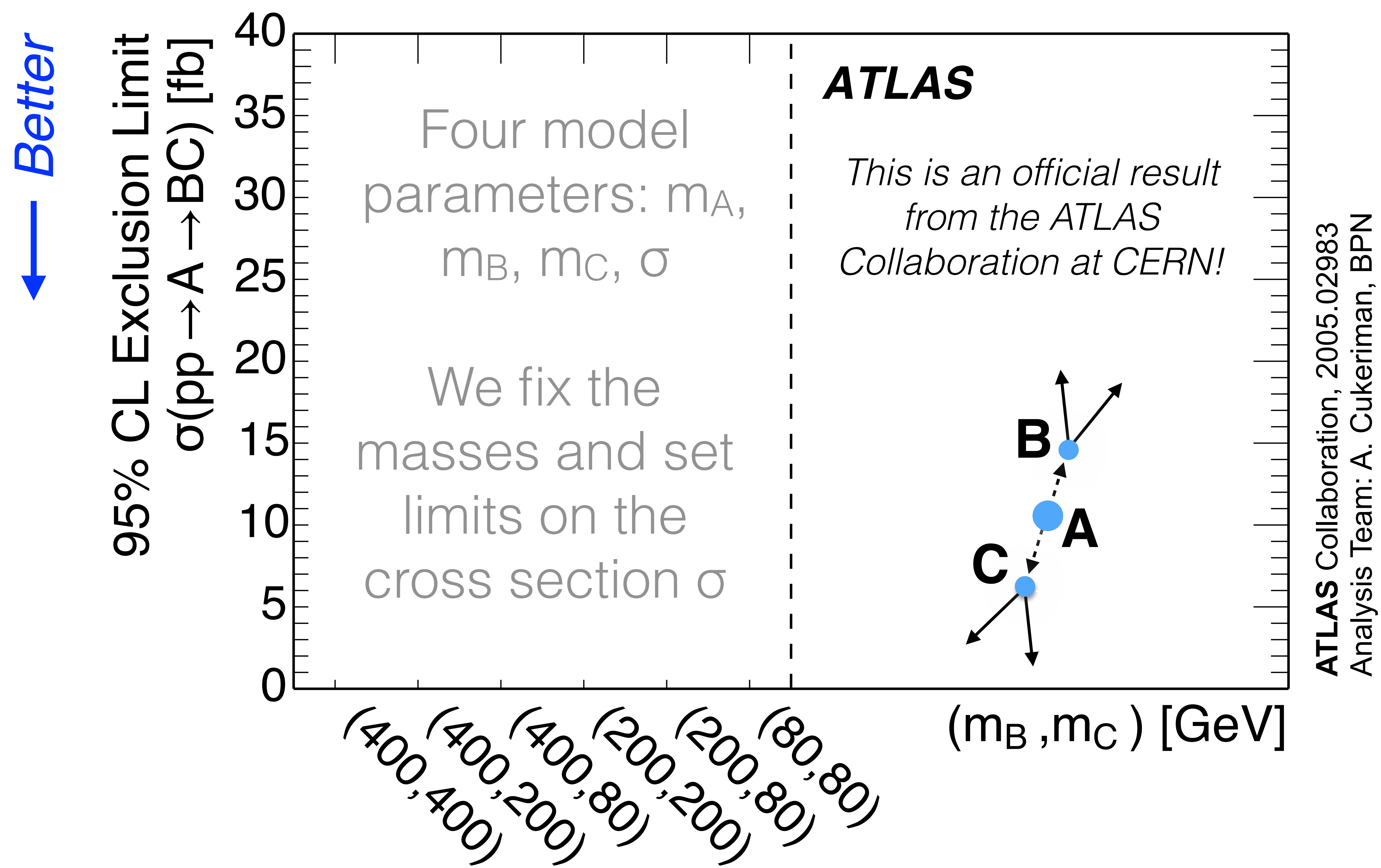


First round, keep it simple: feature space is 2D (jet masses)

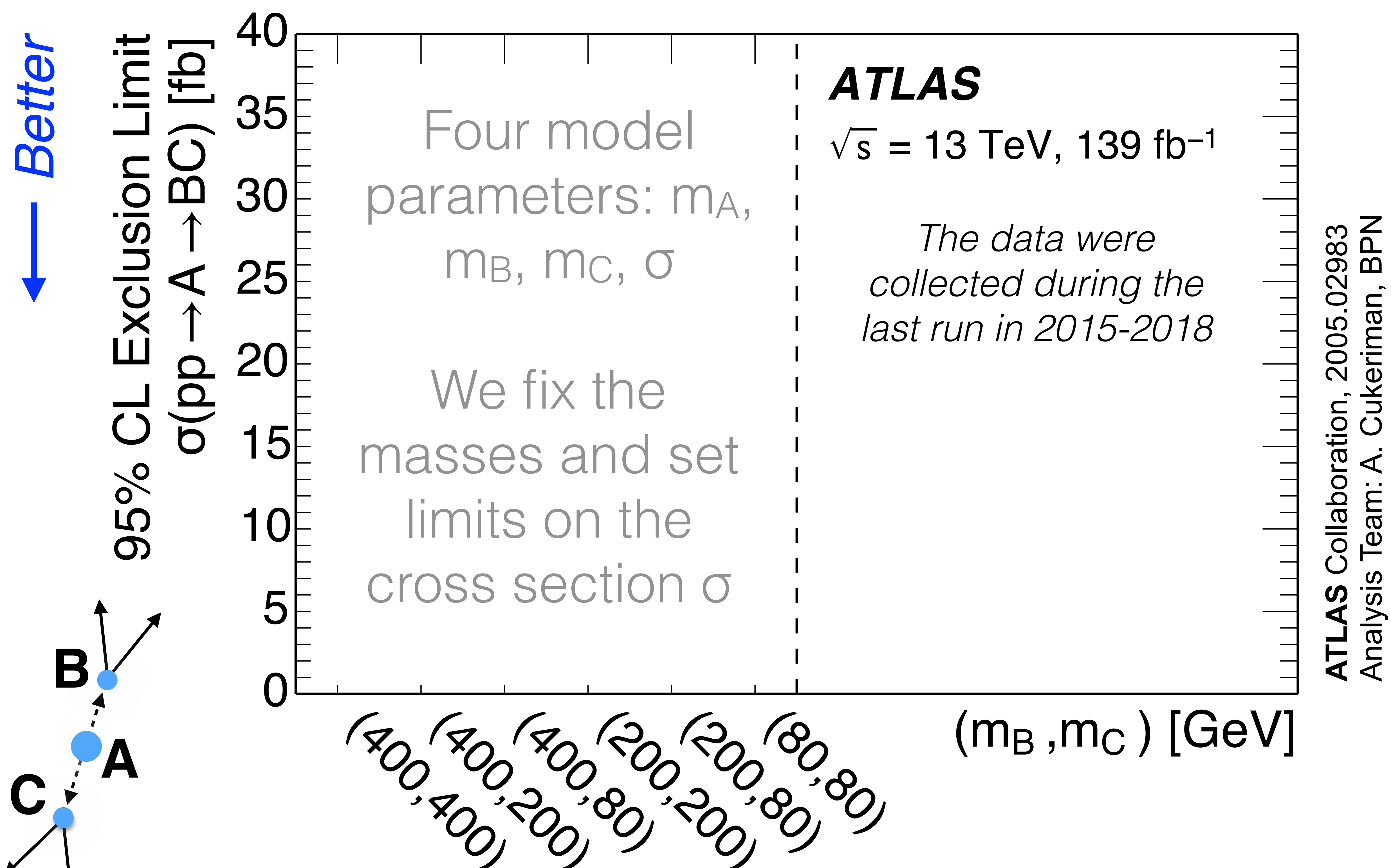
Collision data results New



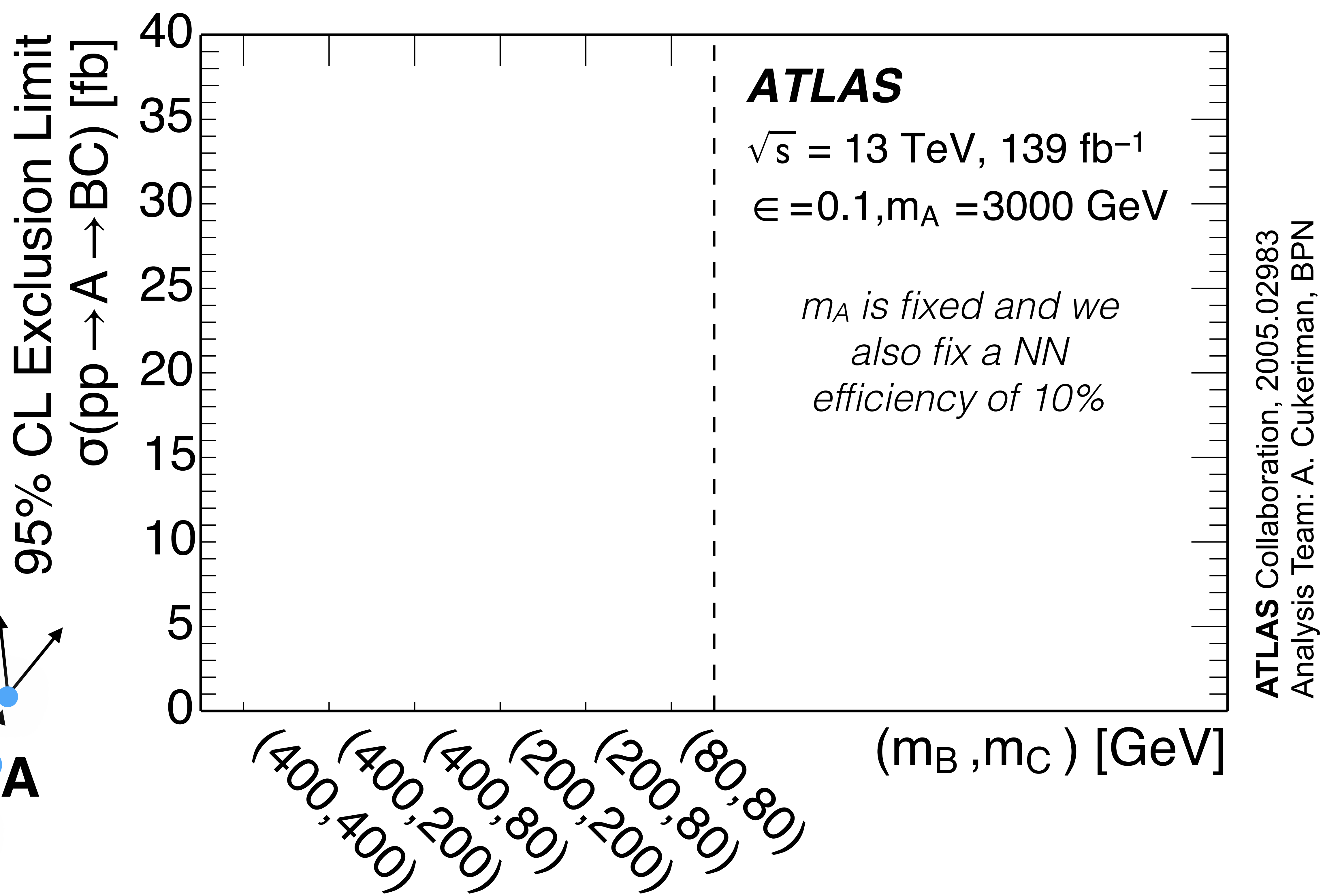
Collision data results New

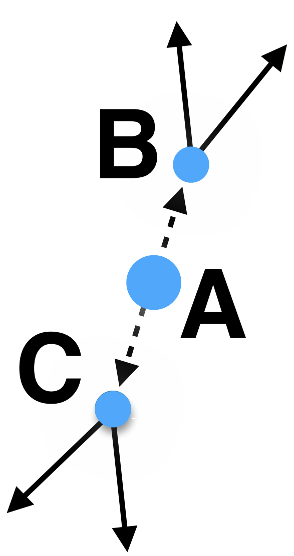


Collision data results New

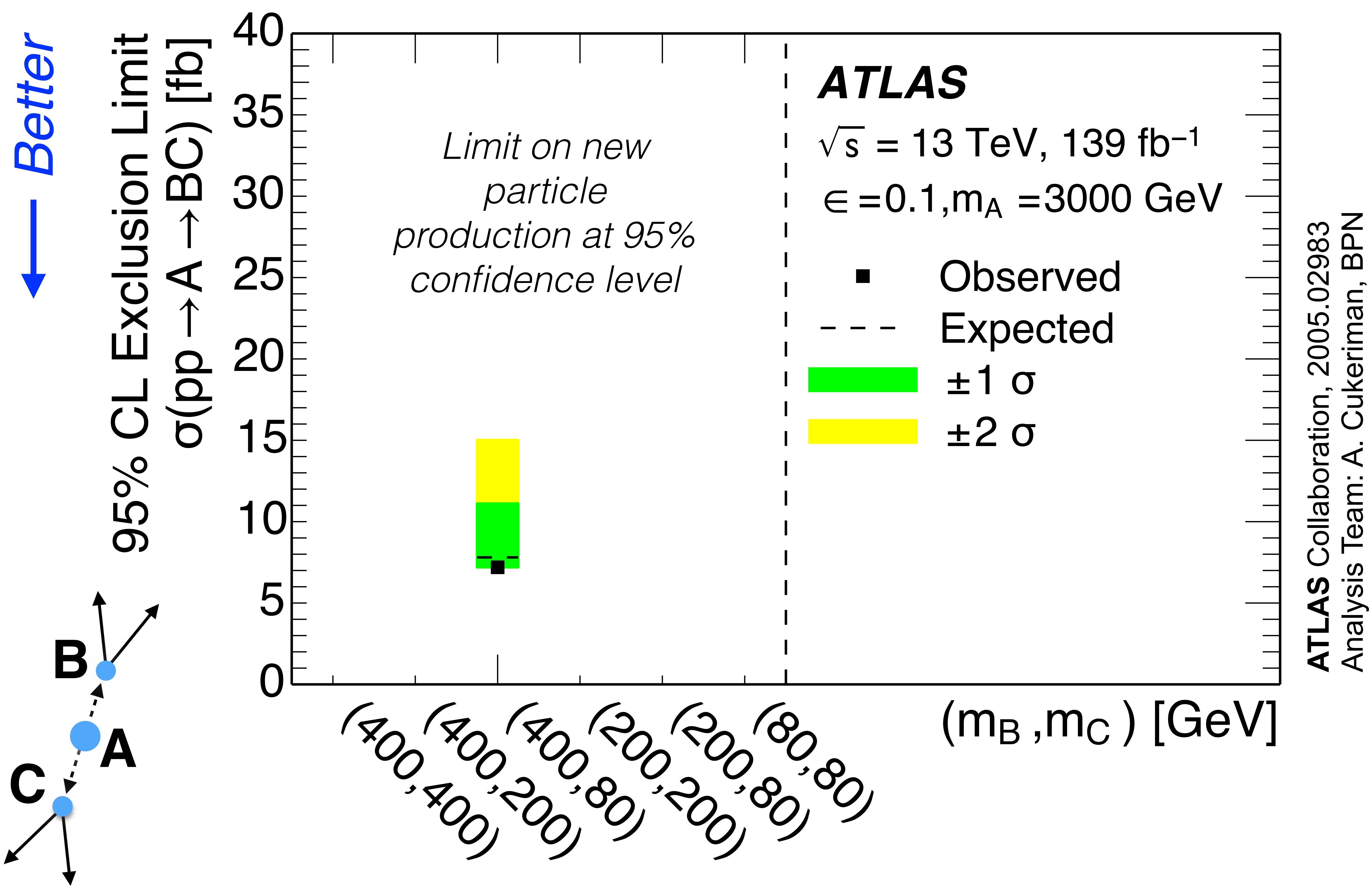


Collision data results New



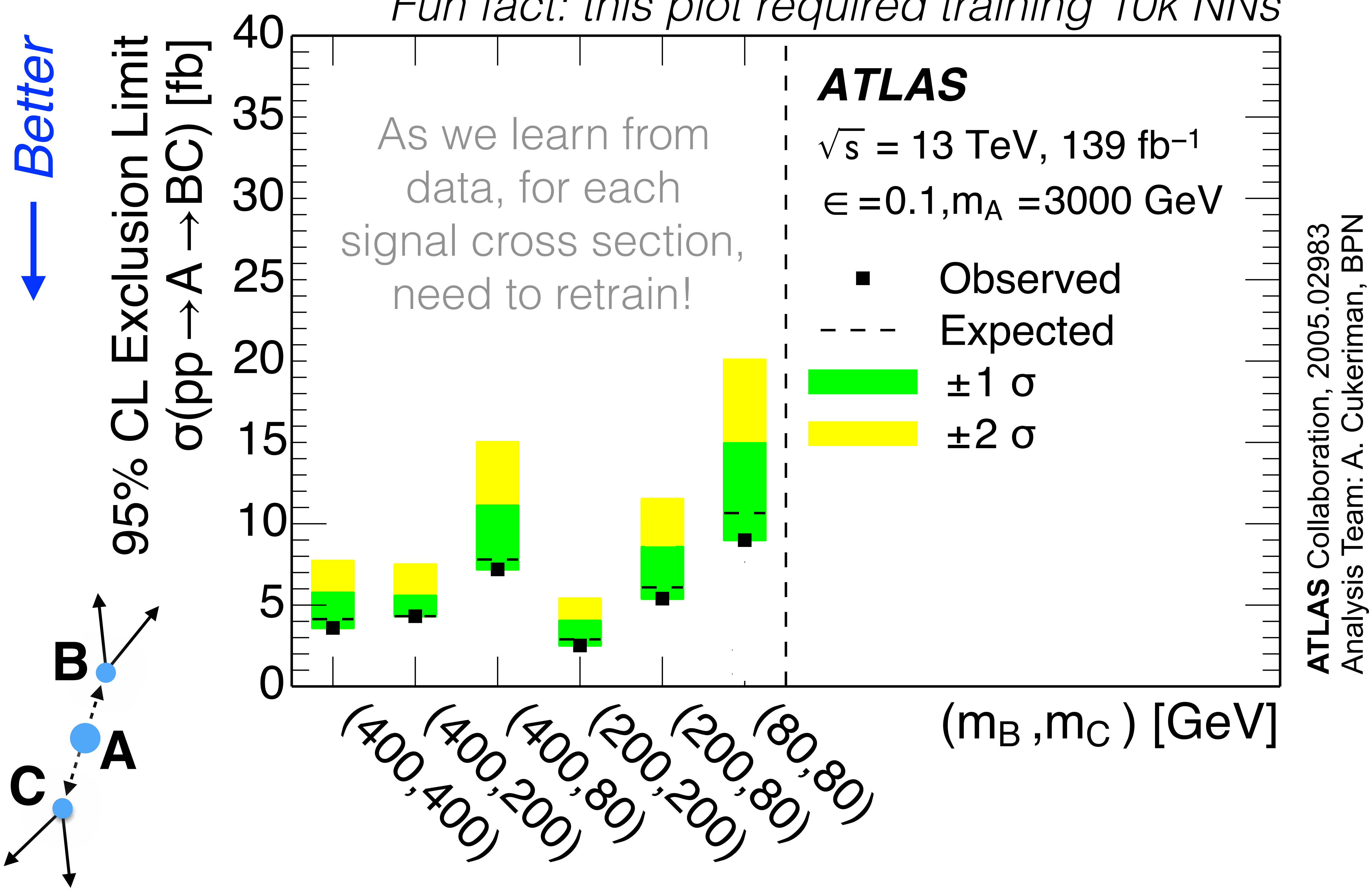


Collision data results New



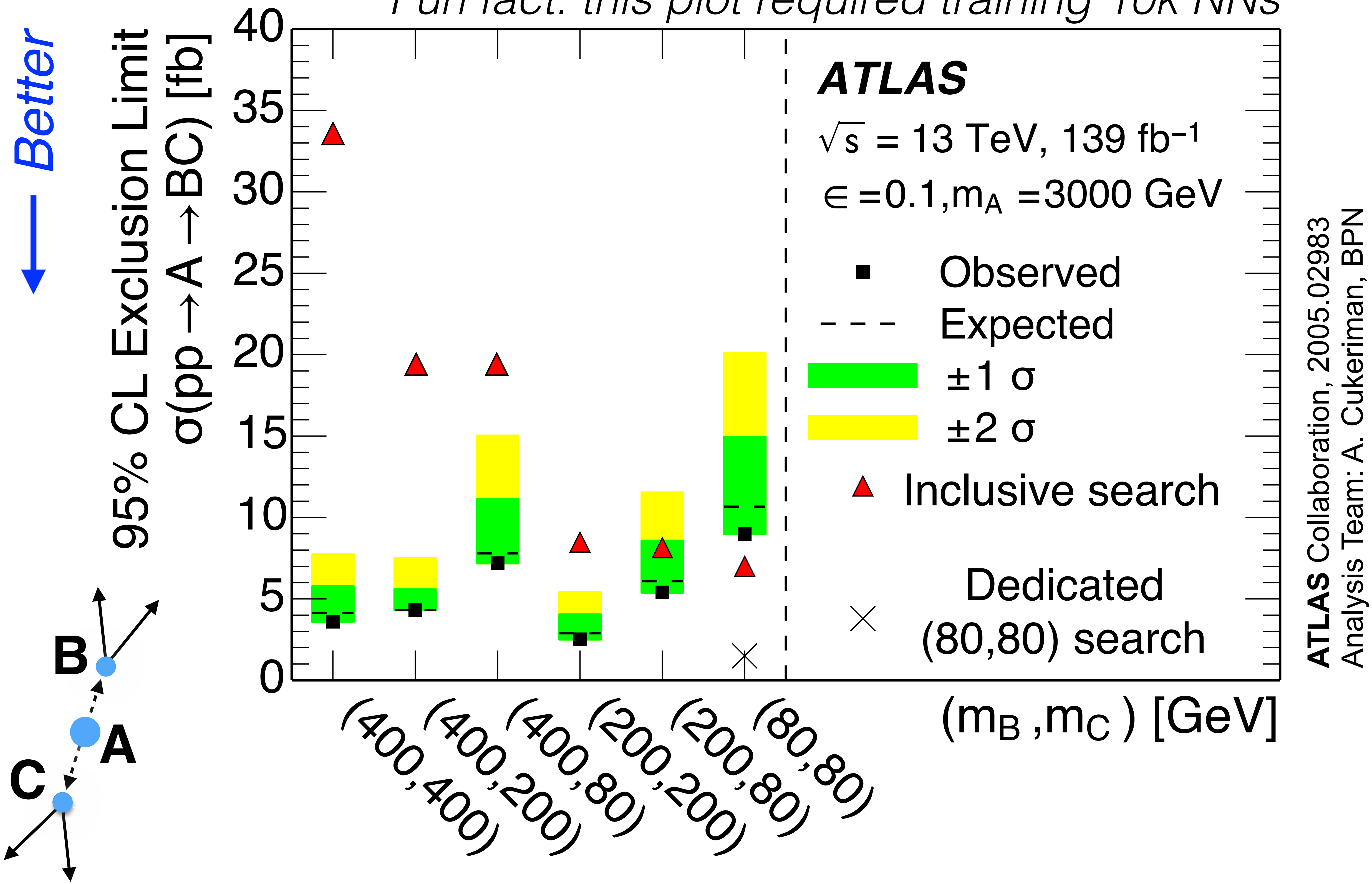
Collision data results New

Fun fact: this plot required training 10k NNs



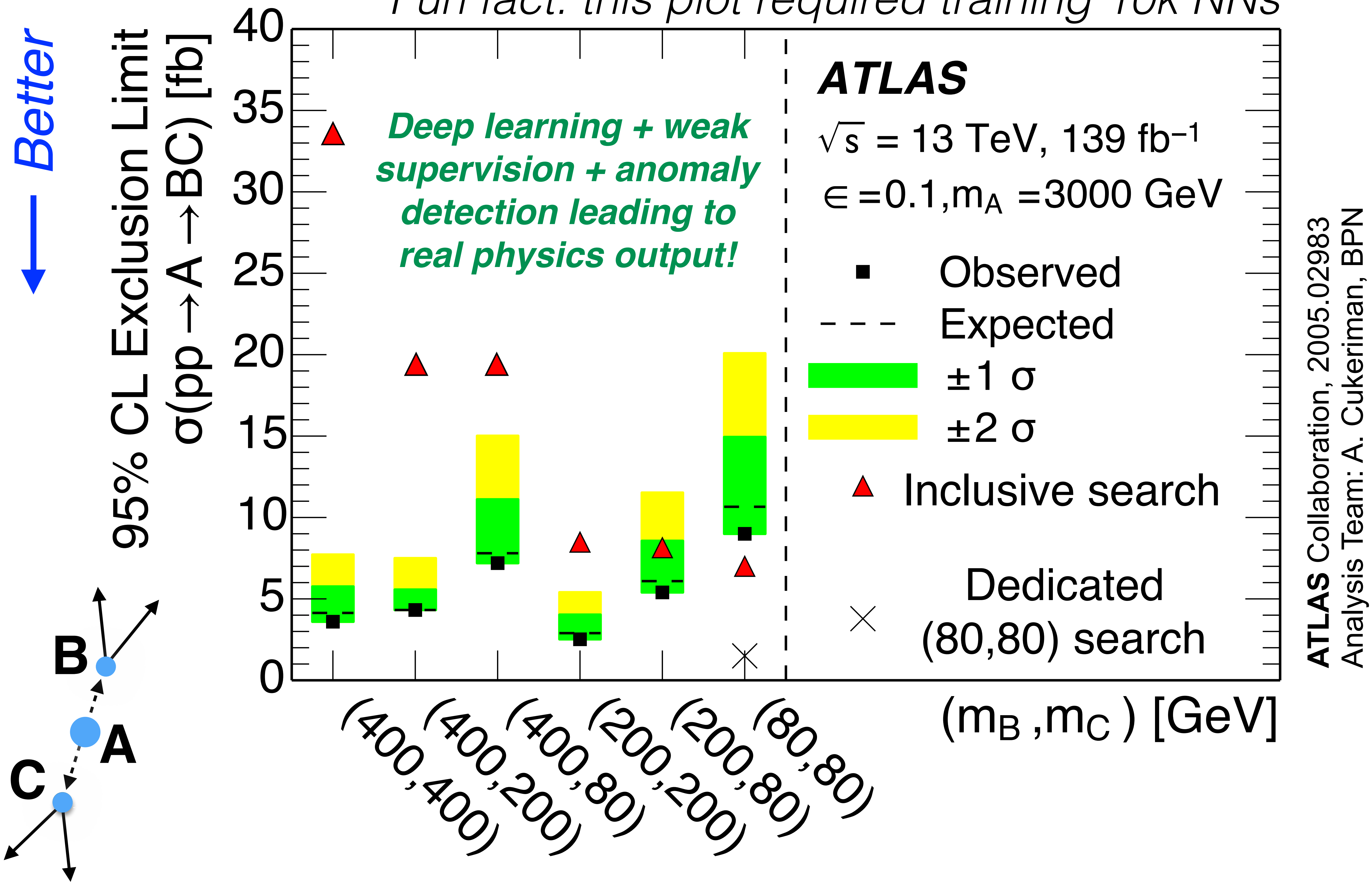
Collision data results New

Fun fact: this plot required training 10k NNs

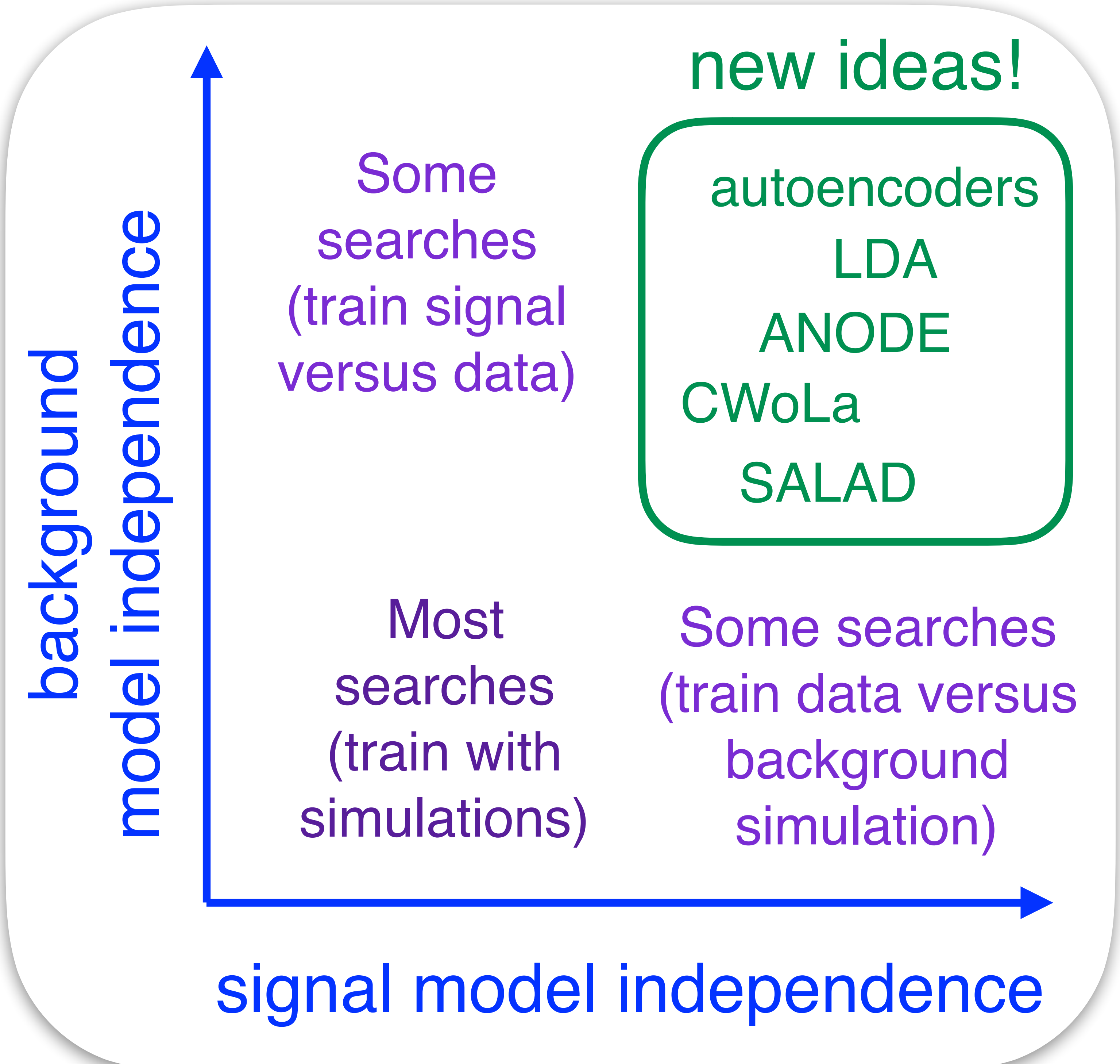


Collision data results New

Fun fact: this plot required training 10k NNs



Anomaly detection future



Rapidly developing area - LHC Olympics 2020 to help facilitate!

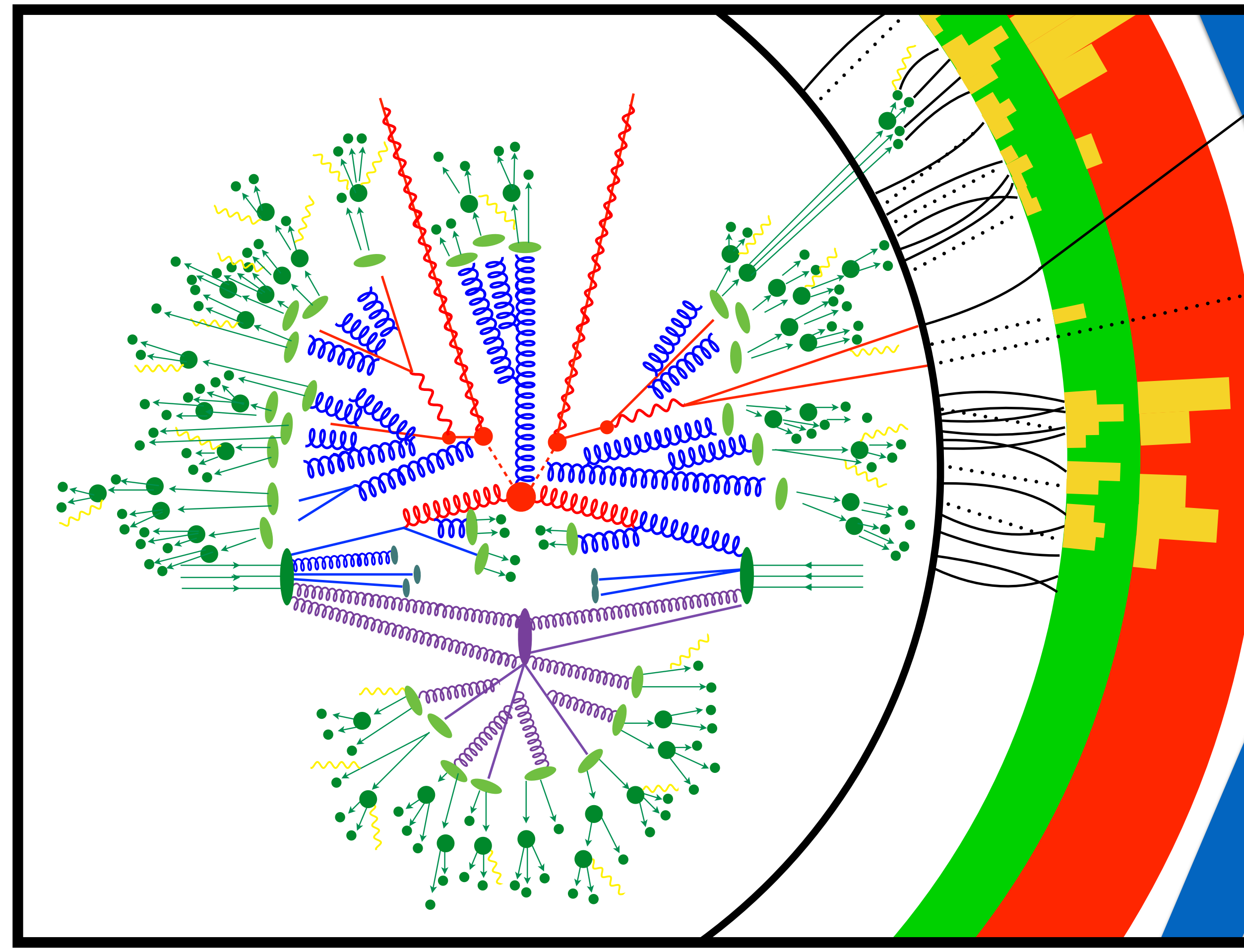
G. Kasieczka. BPN, D. Shih
https://lhco2020.github.io/homepage/

We need your great ideas!

A. Andreassen. ***BPN****, D. Shih, PRD 101 (2020) 095004*
BPN*, D. Shih, PRD 101 (2020) 075042*
J. Collins, K. Howe, ***BPN****, PRL 121 (2018) 241803*
M. Farina, Y. Nakai, D. Shih, PRD 101 (2020) 075021
T. Heimel, G. Kasieczka, T. Plehn, J. Thompson, SciPost Phys. 6 (2019) 030
B. Dillon, D. Faroughy, J. Kamenik, PRD 100 (2019) 056002

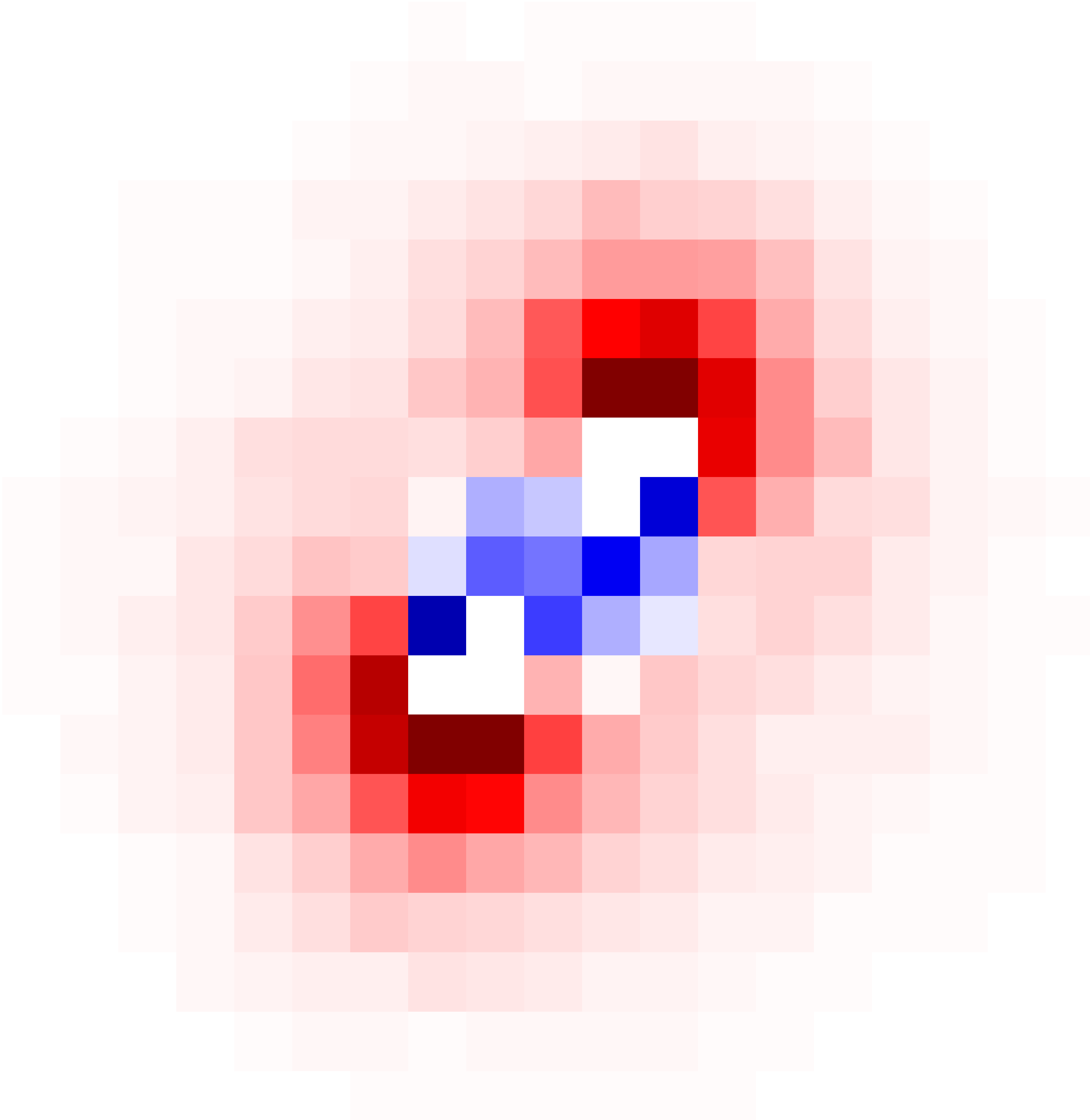
Conclusions and outlook

Deep learning has a great potential to **enhance**, **accelerate**, and **empower** HEP analyses.



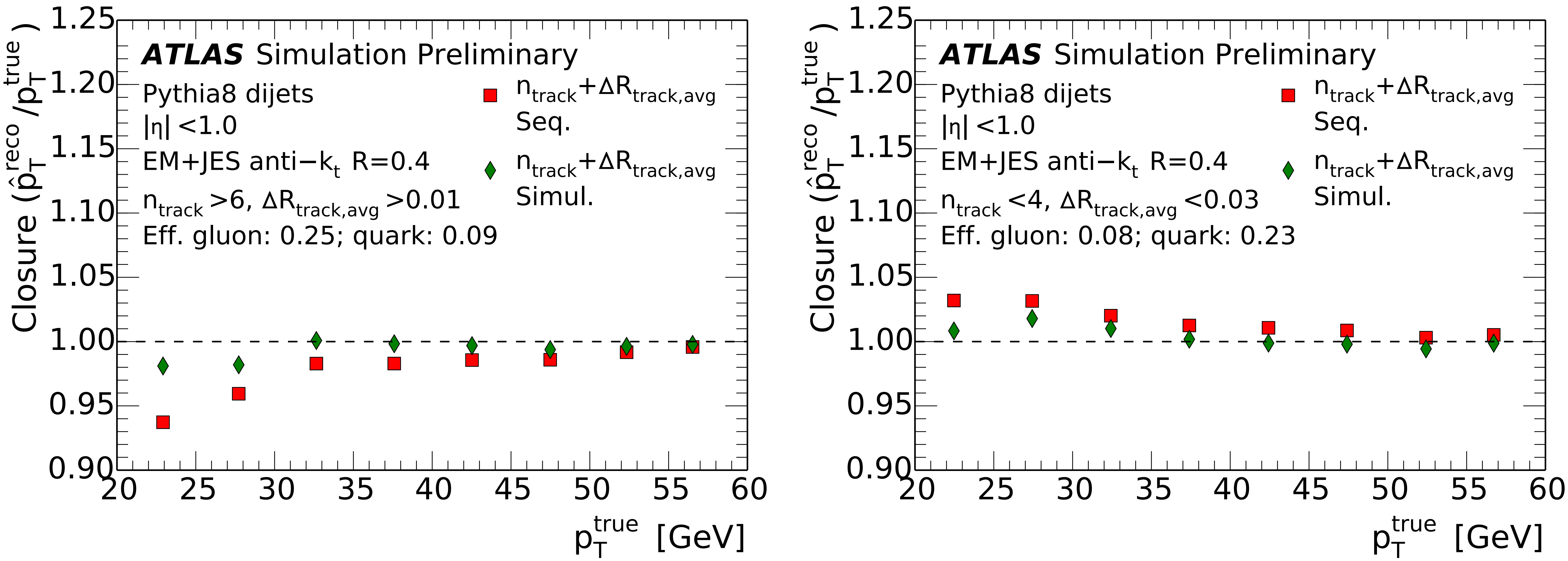
Disclaimer: I have given you a biased perspective of new developments - there is a growing community within HEP!

The **full phase space** of our experiments is now explorable and deep learning will allow us this information to discover fundamental properties of nature!



Fin.

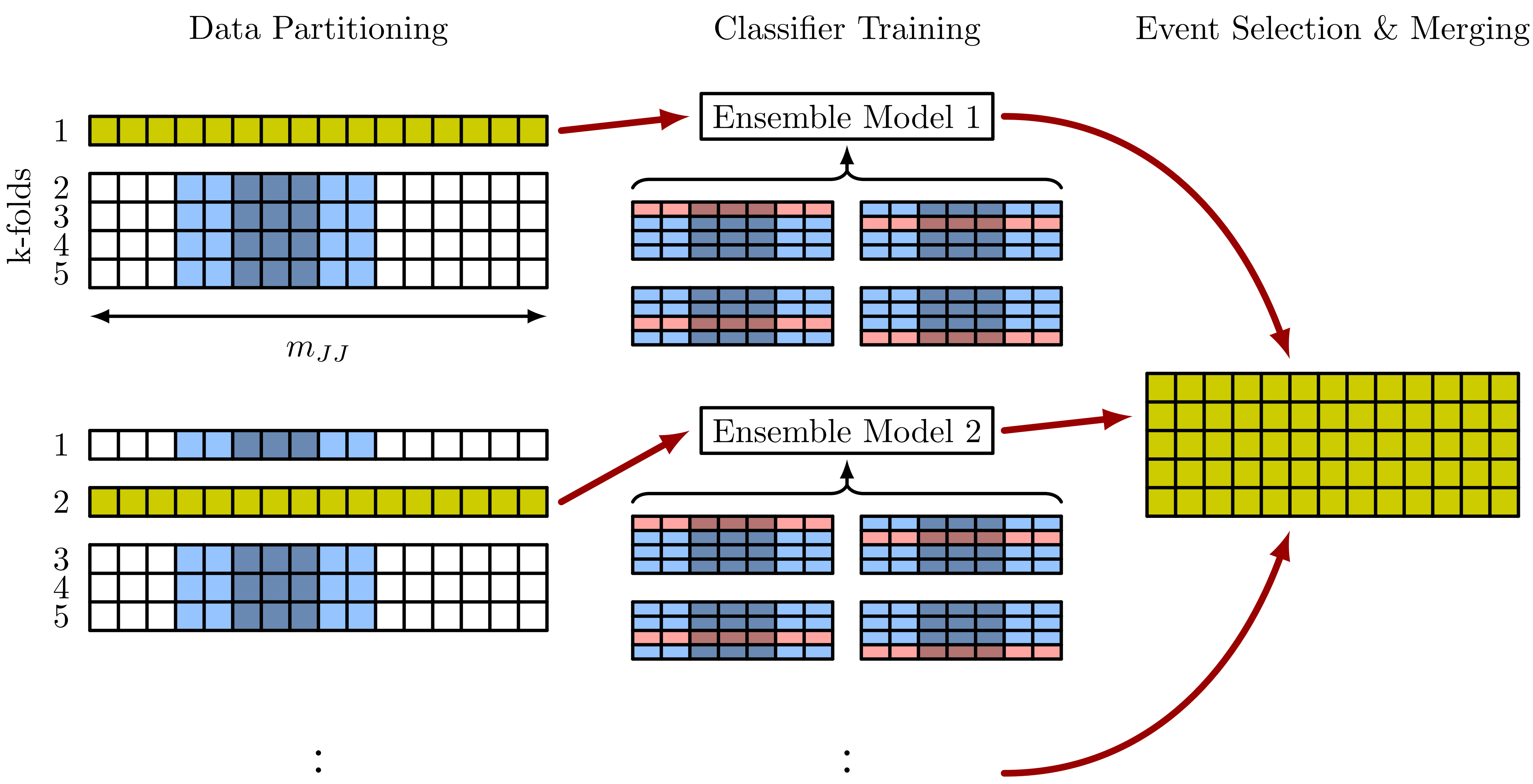
GNI in action



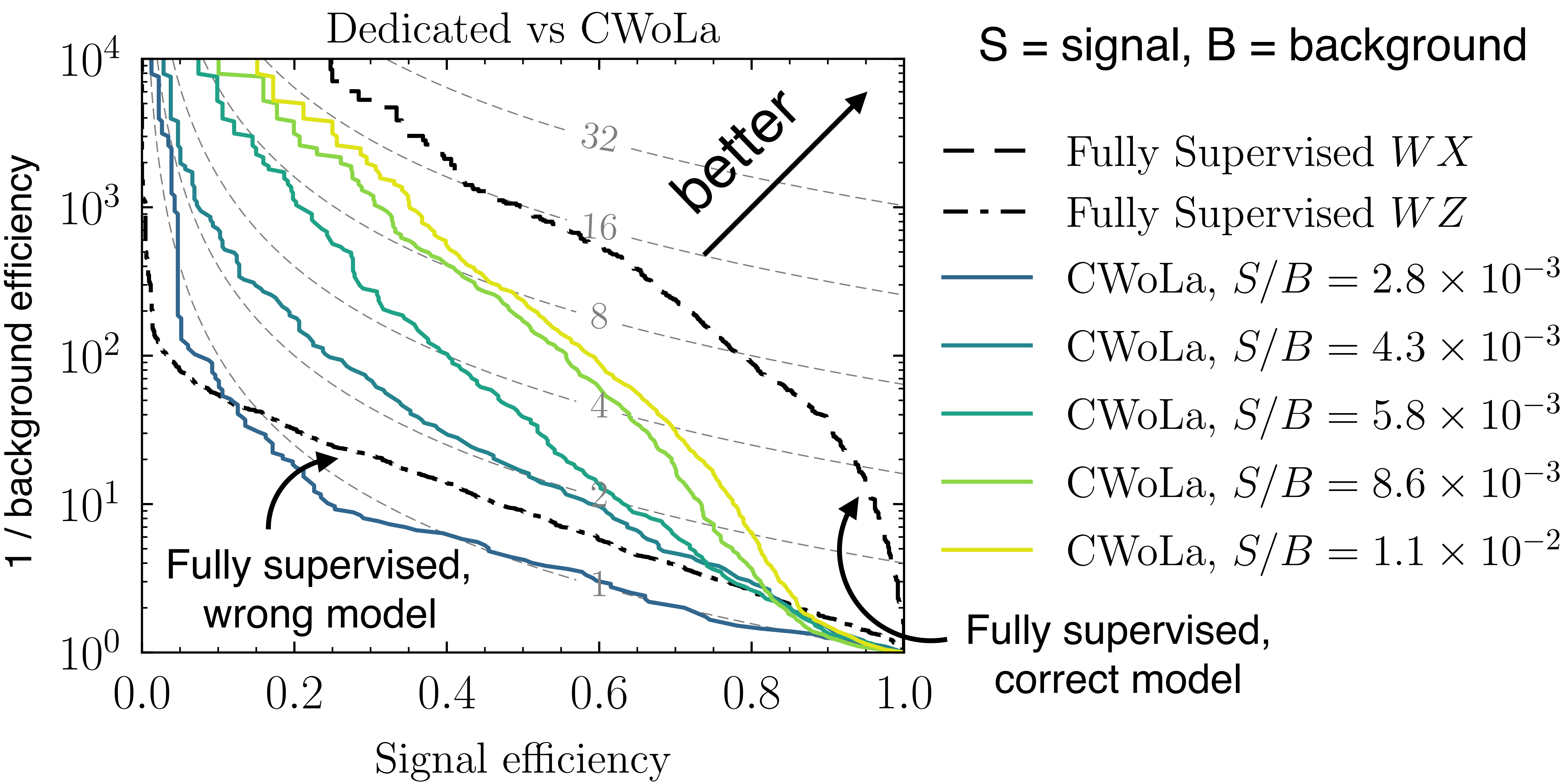
Slightly better closure for the simultaneous calibration.

Weak/unsupervised learning for anomalies

Need to be careful about testing/training on the same data.



CWoLa hunting vs. Full Supervision



If you know what you are looking for, you should look for it. If you don't know, then CWoLa hunting may be able to catch it!