Extracting the most from collider data with deep learning

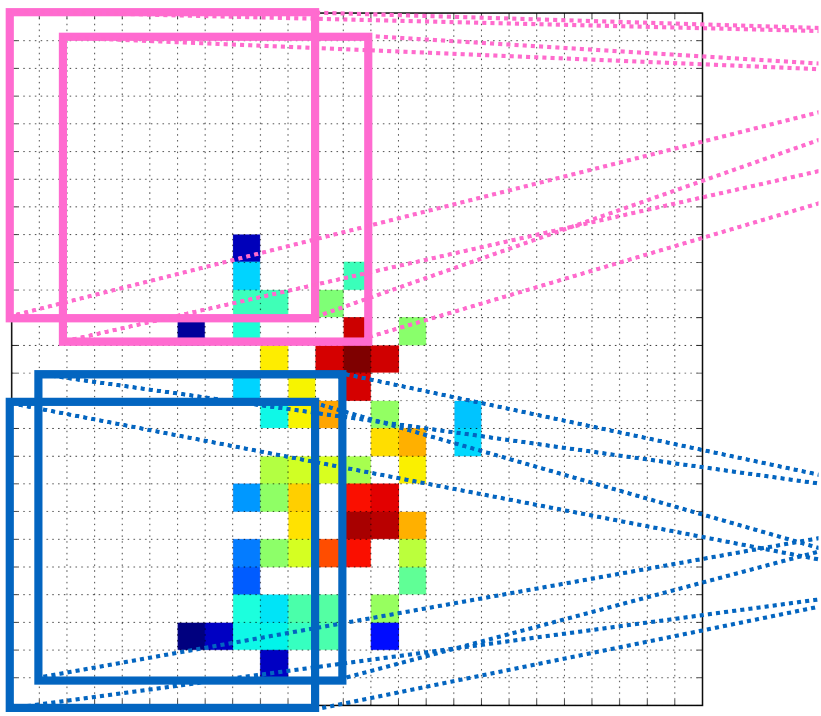
Benjamin Nachman

Lawrence Berkeley National Laboratory

cern.ch/bnachman
bpnachman@lbl.gov
@bpnachman bnachman

BERKELEY LAB

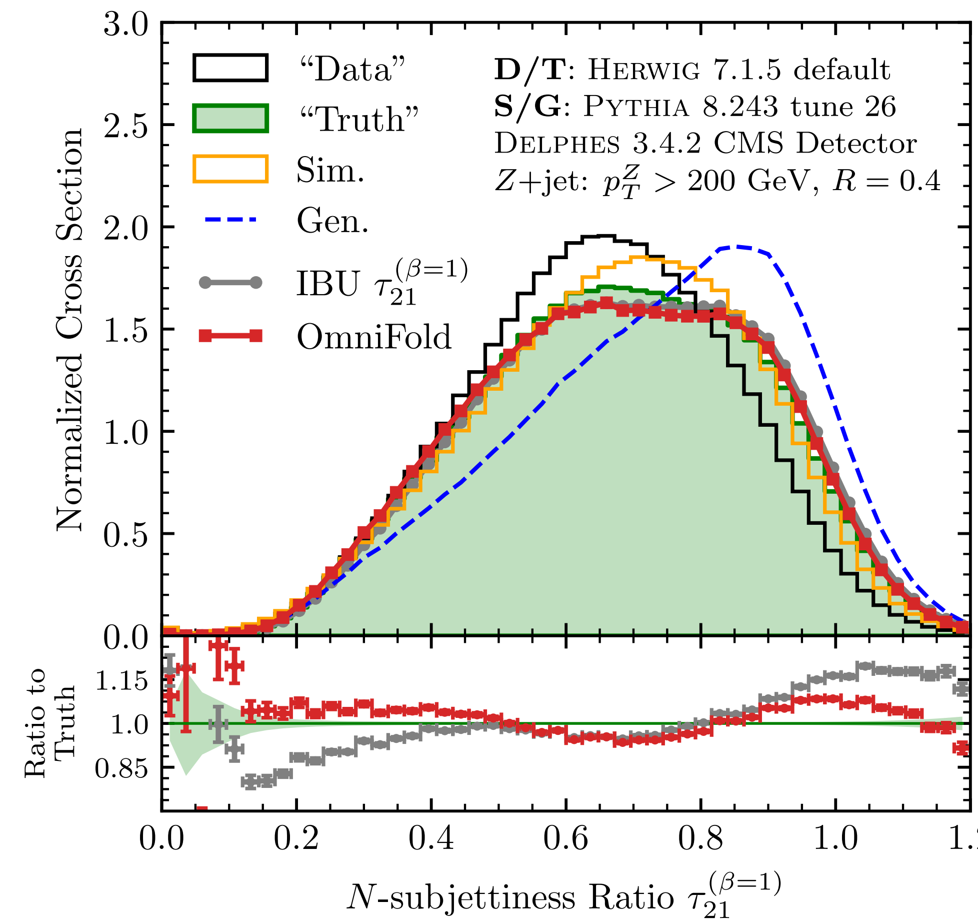
BERKELEY EXPERIMENTAL PARTICLE PHYSICS



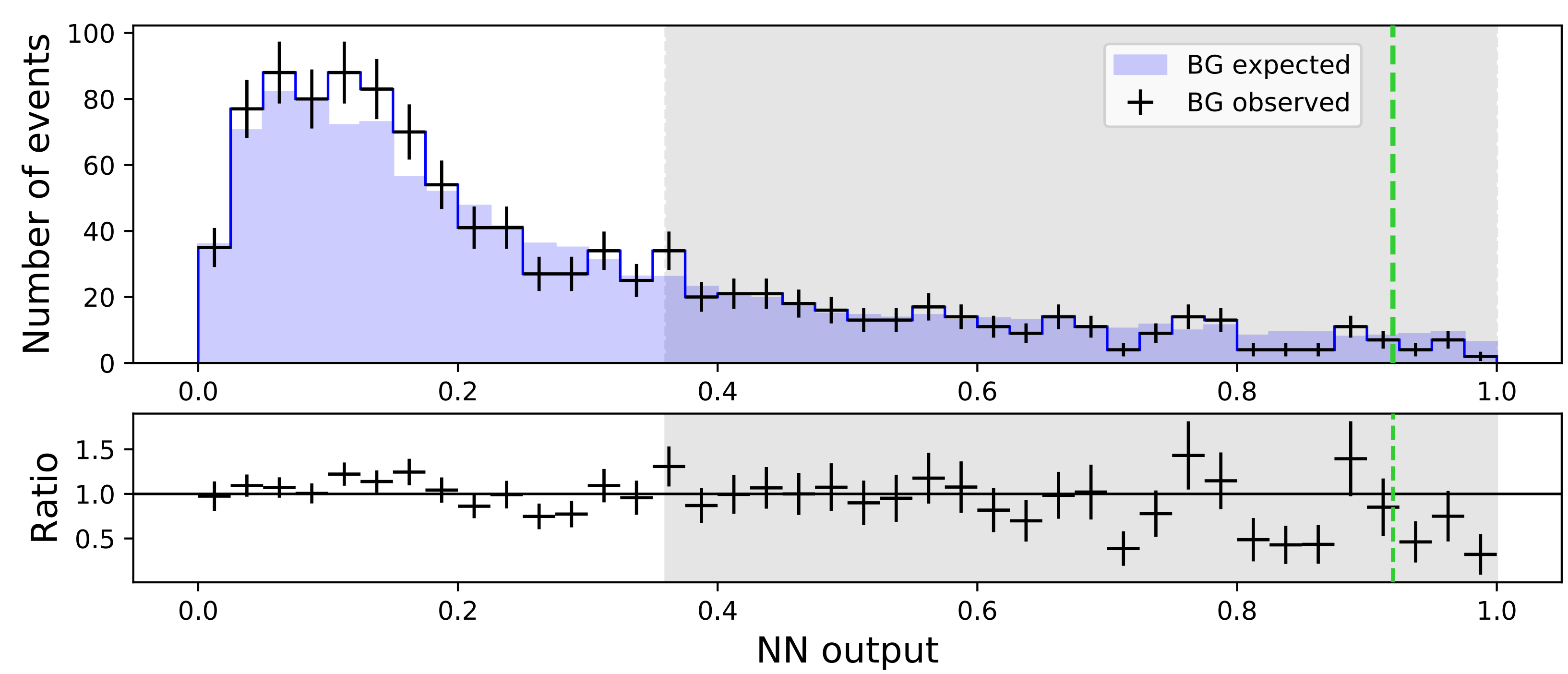
KMI, Nagoya
Machine Learning at LHC
Feb. 5, 2020

Outline for today

- (Optimally) using NNs for analysis
  - Improving search sensitivity
  - Enhancing SM measurements
- Uncertainties with NNs
  - What are they?
  - How to improve (or avoid)?
- Anomaly detection



A. Andreassen, E. Metodiev, P. Komiske, **BPN**, J. Thaler, 1911.09107



BPN & C. Shimmin, 1910.08606

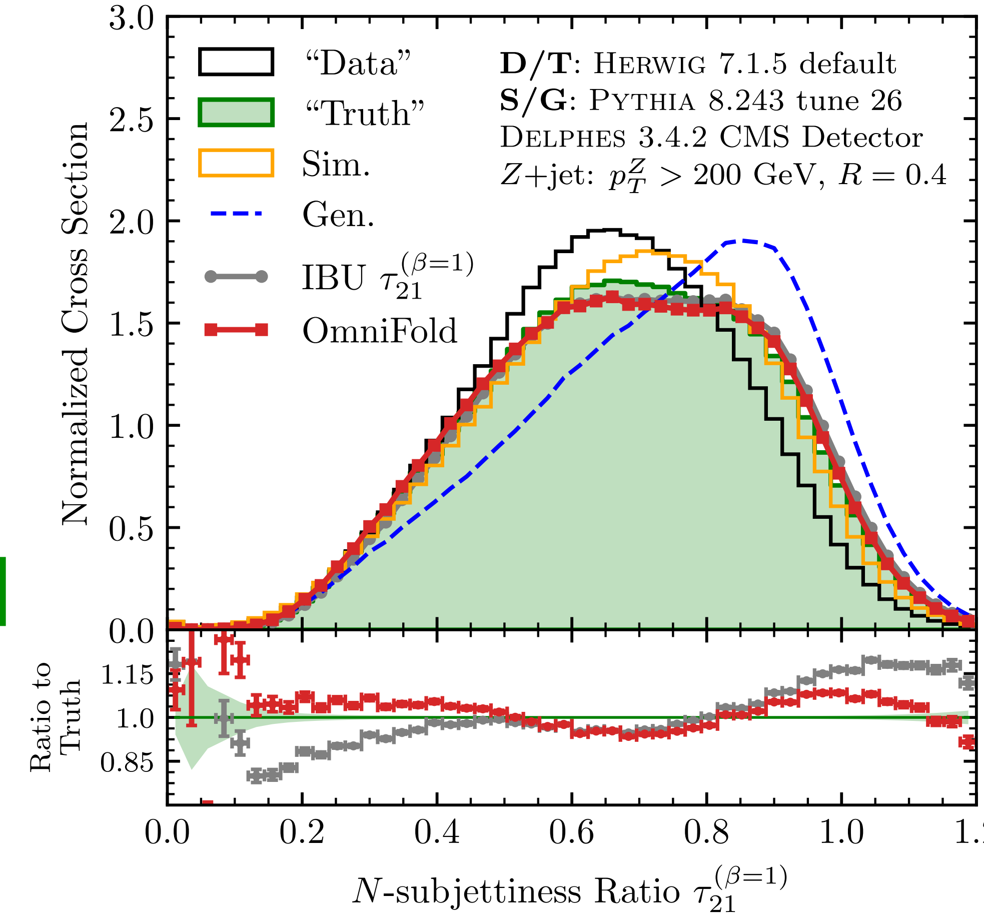
Outline for today

- (Optimally) using NNs for analysis
  - Improving search sensitivity
  - Enhancing SM measurements

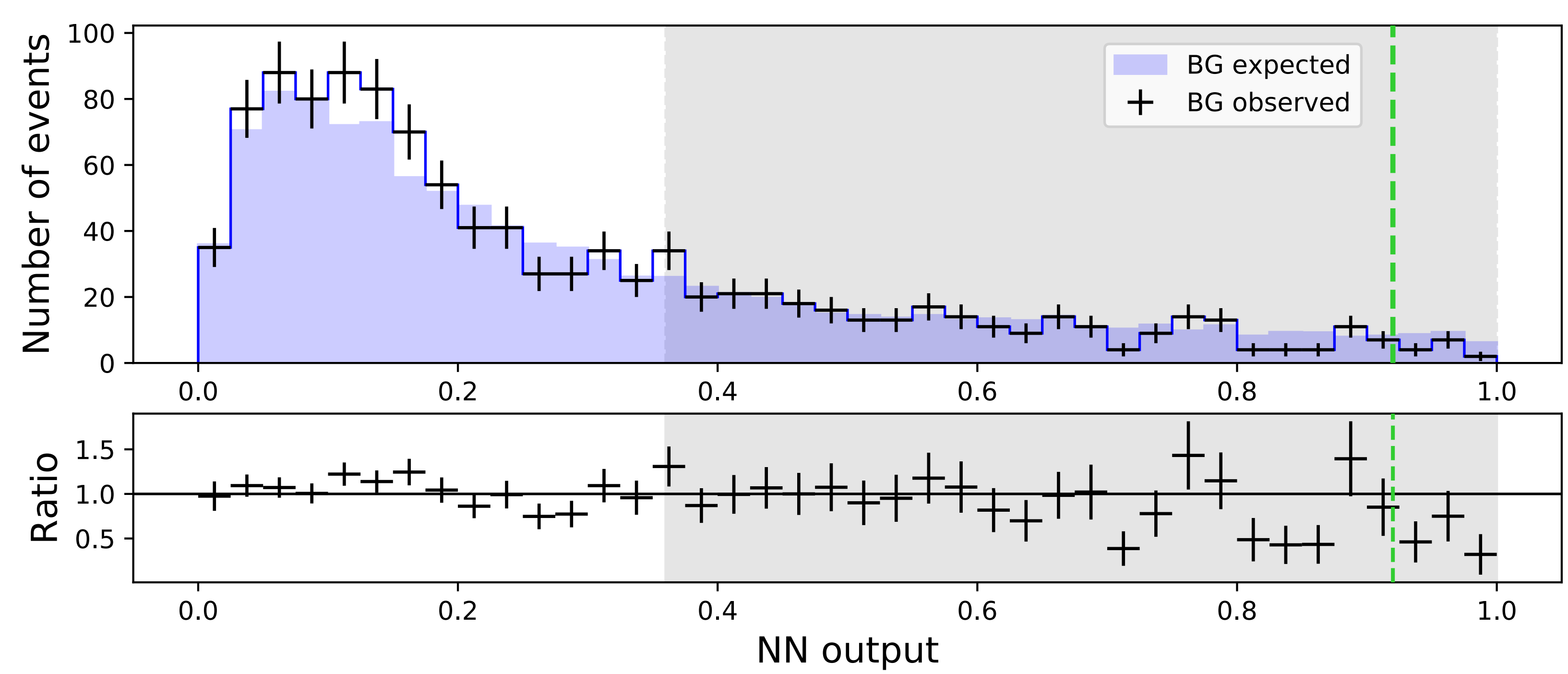
~first third

- Uncertainties with NNs
  - What are they? **~second third**
  - How to improve (or avoid)?
- Anomaly detection

~last third



A. Andreassen, E. Metodiev, P. Komiske, **BPN**, J. Thaler, 1911.09107



BPN & C. Shimmin, 1910.08606

Data analysis in HEP

Theory of everything

↓

Physics simulators

↓

Detector-level observables

↓

Pattern recognition

↔

Nature

↓

Experiment

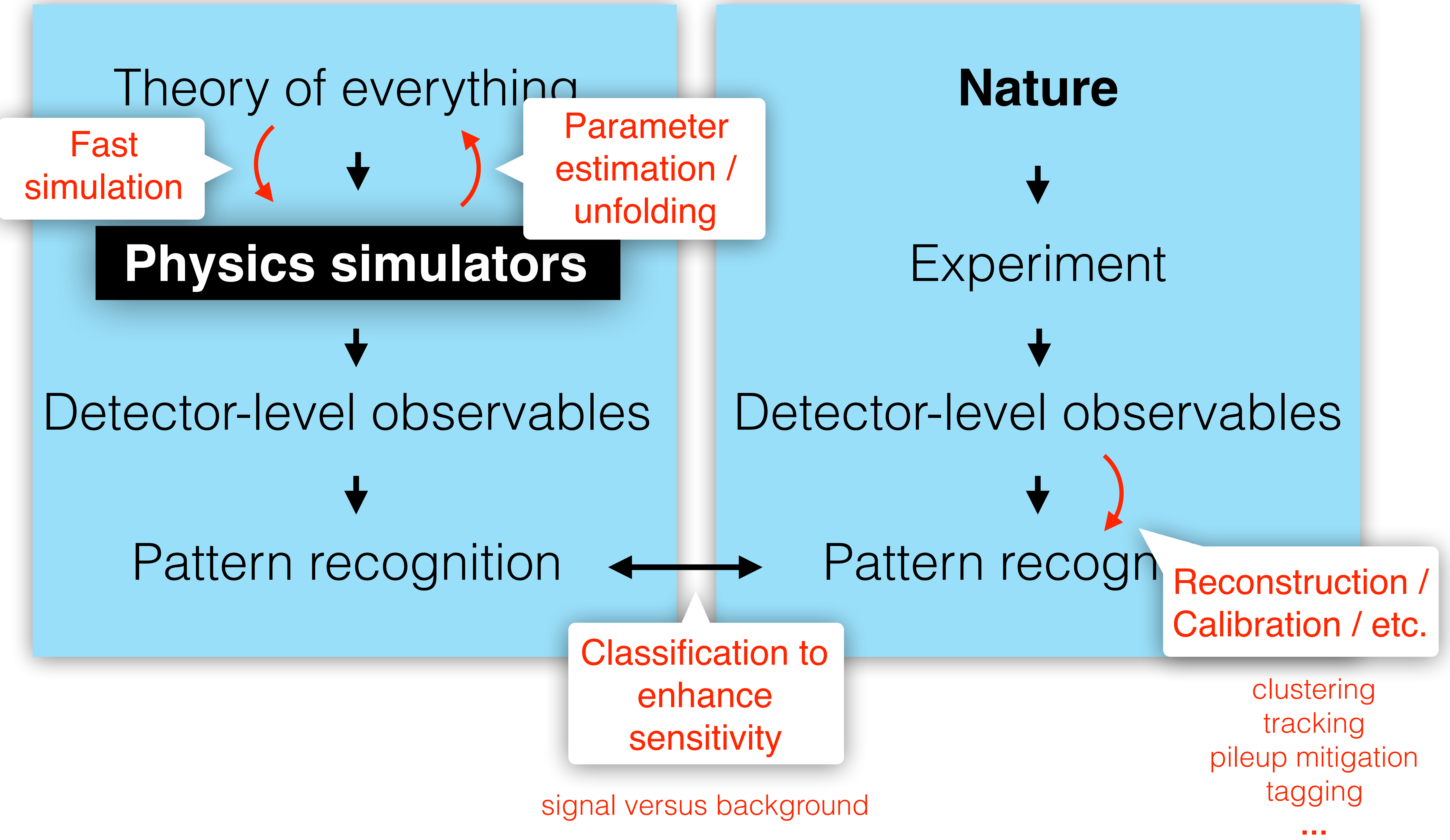
↓

Detector-level observables

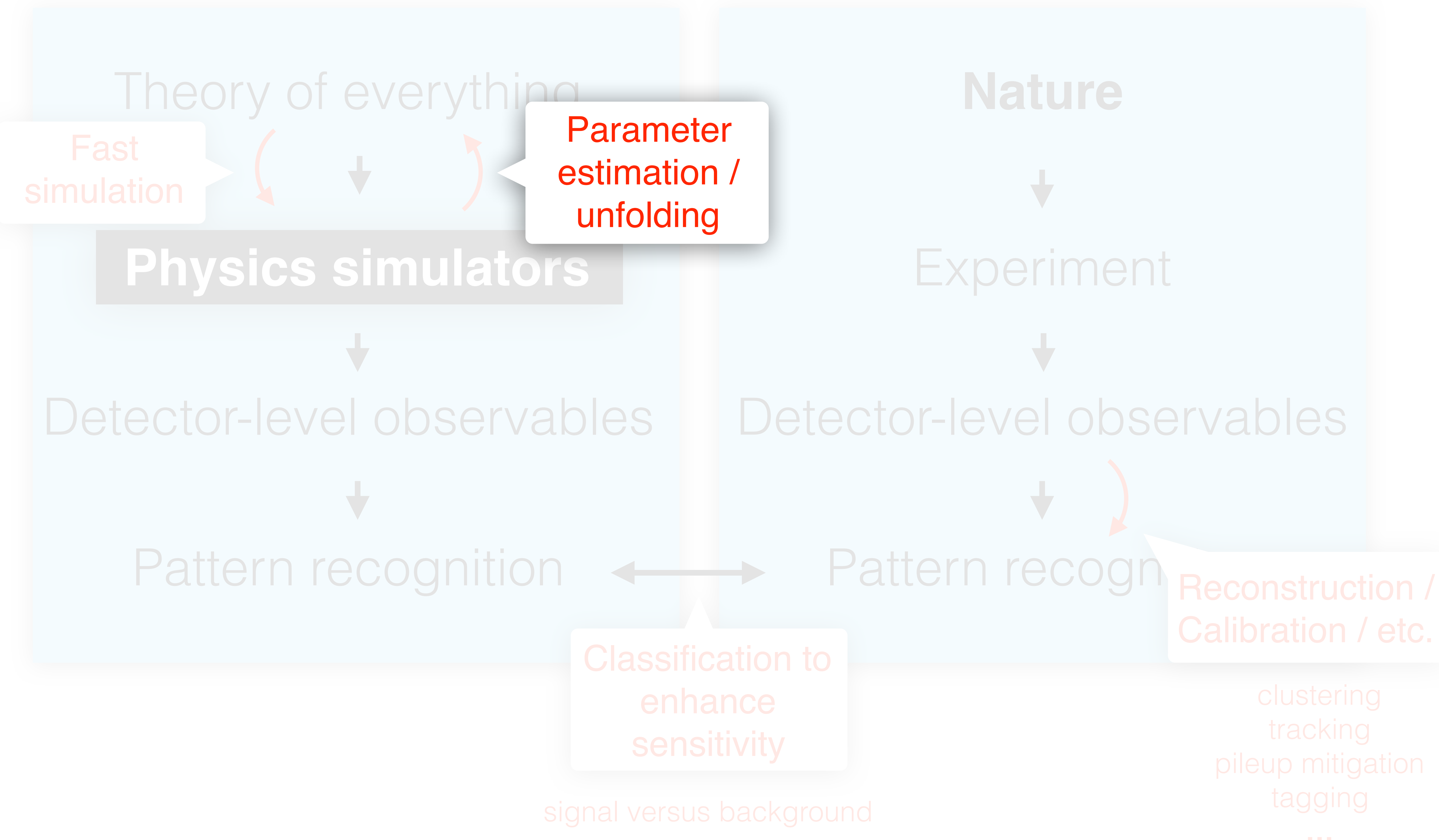
↓

Pattern recognition

Data analysis in HEP + Deep Learning



Data analysis in HEP + Deep Learning



Data analysis in HEP + Deep Learning

Theory of everything

Fast simulation

Parameter estimation / unfolding

Physics simulators

Nature

Experiment

In low-dimensions, can try to explicitly model p(x|params).
In high-dimensions, p is intractable. Need likelihood-free!

Pattern recognition

Pattern recogn

Reconstruction / Calibration / etc.

Classification to enhance sensitivity

signal versus background

clustering
tracking
pileup mitigation
tagging
...

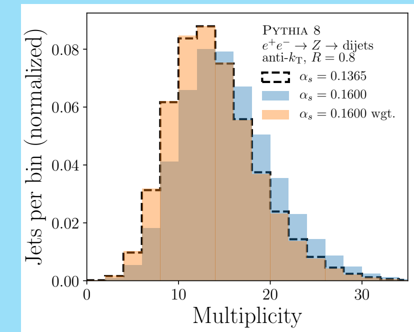
All of the measurable information

Full phase space + likelihood free

"DCTR"

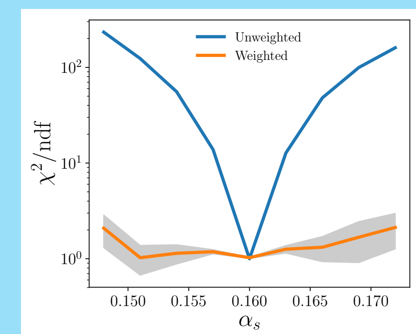
New simulations

morph one simulation into another



Continuous variations

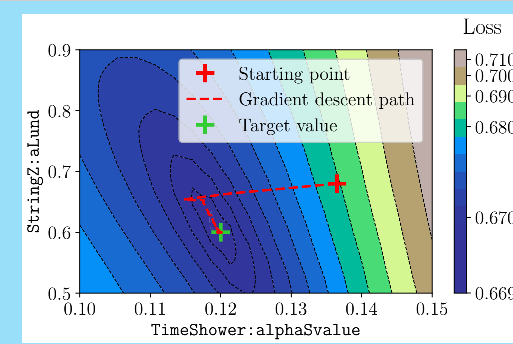
learn the dependence on parameters



A. Andreassen and **BPN**, 1909.03081

Parameter estimation

use classification loss to fit parameters



Unfolding

iterate the morphing to remove distortions

A. Andreassen et al., 1911.09107

Full phase space reweighting

Facts: detector-level simulation is **expensive**; (e.g. **Geant4**)
pre-detector particle simulations (e.g. **Pythia**) are **cheap**.

Imagine we have one high-statistics expensive simulation.
(Pythia + Geant4)

Suppose there is another simulation of the pre-detector dynamics. Can we use the pre-detector parts to achieve a detector version of the new simulation?

Answer: Yes! Full phase space reweighing with neural networks.

Likelihood free reweighting

Let x be a simulated event. It could be composed of many hundreds of particles.

Suppose that **p(x)** and **q(x)** are the densities for the two simulations.

We can **reweight** the first simulation into the second by assigning per-event weights of **q(x)/p(x)**.

Likelihood free reweighting

$\forall \mathcal{O},$

$$\text{weighted } \langle \mathcal{O}(X) \rangle_{X \sim \mathrm{Pr}} \equiv \sum_x \mathrm{Pr}(x) w(x) \mathcal{O}(x)$$

$$\equiv \sum_x \mathrm{Pr}'(x) \mathcal{O}(x) = \langle \mathcal{O}(X) \rangle_{X \sim \mathrm{Pr}'}$$

i.e. any expectation value computed with weighted events (*Pr, w*) is the same as the expectation from a different probability (*Pr'*)

We can **reweight** the first simulation into the second by assigning per-event weights of **q(x)**/**p(x)**.

Likelihood free reweighting

Let x be a simulated event. It could be composed of many hundreds of particles.

Suppose that **p(x)** and **q(x)** are the densities for the two simulations.

We can **reweight** the first simulation into the second by assigning per-event weights of **q(x)**/**p(x)**.

…what if we don't (and can't easily) know **q** and **p**?

Likelihood free reweighting

Solution: train a neural network to distinguish the two simulations. Call this ***f***.

It is not hard to show that if ***f*** is optimal and you train with the most popular loss functions, then

$$\frac{f(x)}{1-f(x)} \propto \frac{q(x)}{p(x)}$$

(for weighting, we don't care about overall constants - in this case, it is the class imbalance during training)

Likelihood free reweighting

Solution: train a neural ne... to distinguish the two ... this ***f***.

It is no... ...at if ***f*** is optimal and you ...most popular loss functions, then

This is great because **classification is easy** while **generation is hard**.

$$\frac{f(x)}{1-f(x)} \propto \frac{q(x)}{p(x)}$$

(for weighting, we don't care about overall constants - in this case, it is the class imbalance during training)

Example: electron-positron collisions

Learn a classifier on the full observable phase space (momenta + particle flavor) and then check with some standard observables.

Our events have a variable number of particles & due to quantum mechanics, are permutation invariant. Thus, we use a deep-sets variant called **particle flow networks**.

PFNs: Komiske, Metodiev, Thaler, JHEP 01 (2019) 121

Deep sets: Zaheer et al., NIPS 2017

Example: electron-positron collisions

Learn a classifier on the full observable phase space (momenta + particle flavor) and then check with some standard observables.

Our events have a variable number of rticles & due to quantum mechanics, a ve use a deep-sets varia .

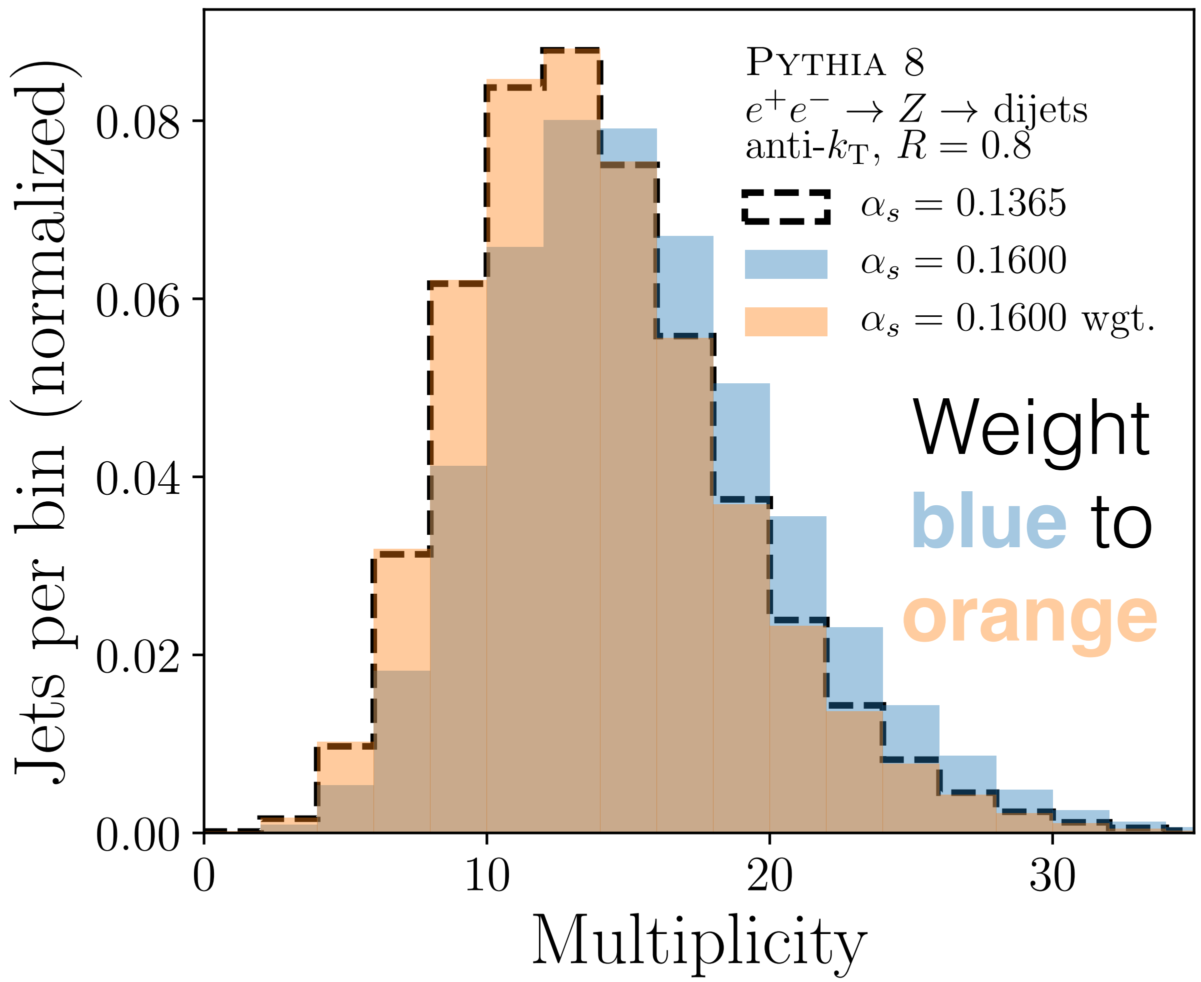
Just to stress: this gives you a new simulation with all the 4-vectors that is statistically indistinguishable.

PFNs: Komiske, Metodiev, Thaler, JHEP 01 (2019) 121

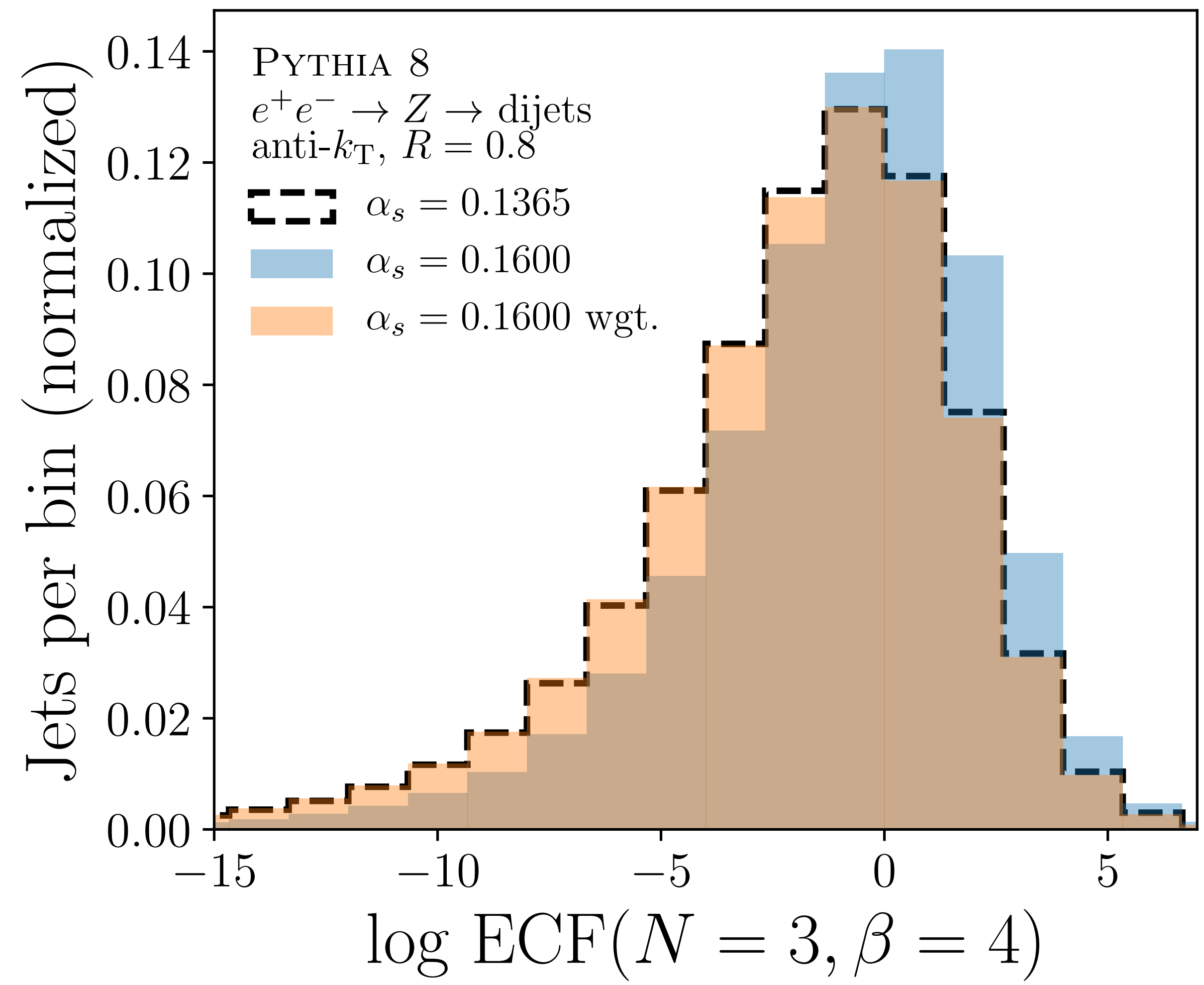
Deep sets: Zaheer et al., NIPS 2017

Example: electron-positron collisions

Learn a classifier on the full observable phase space (momenta + particle flavor) and then check with some standard 1D observables.

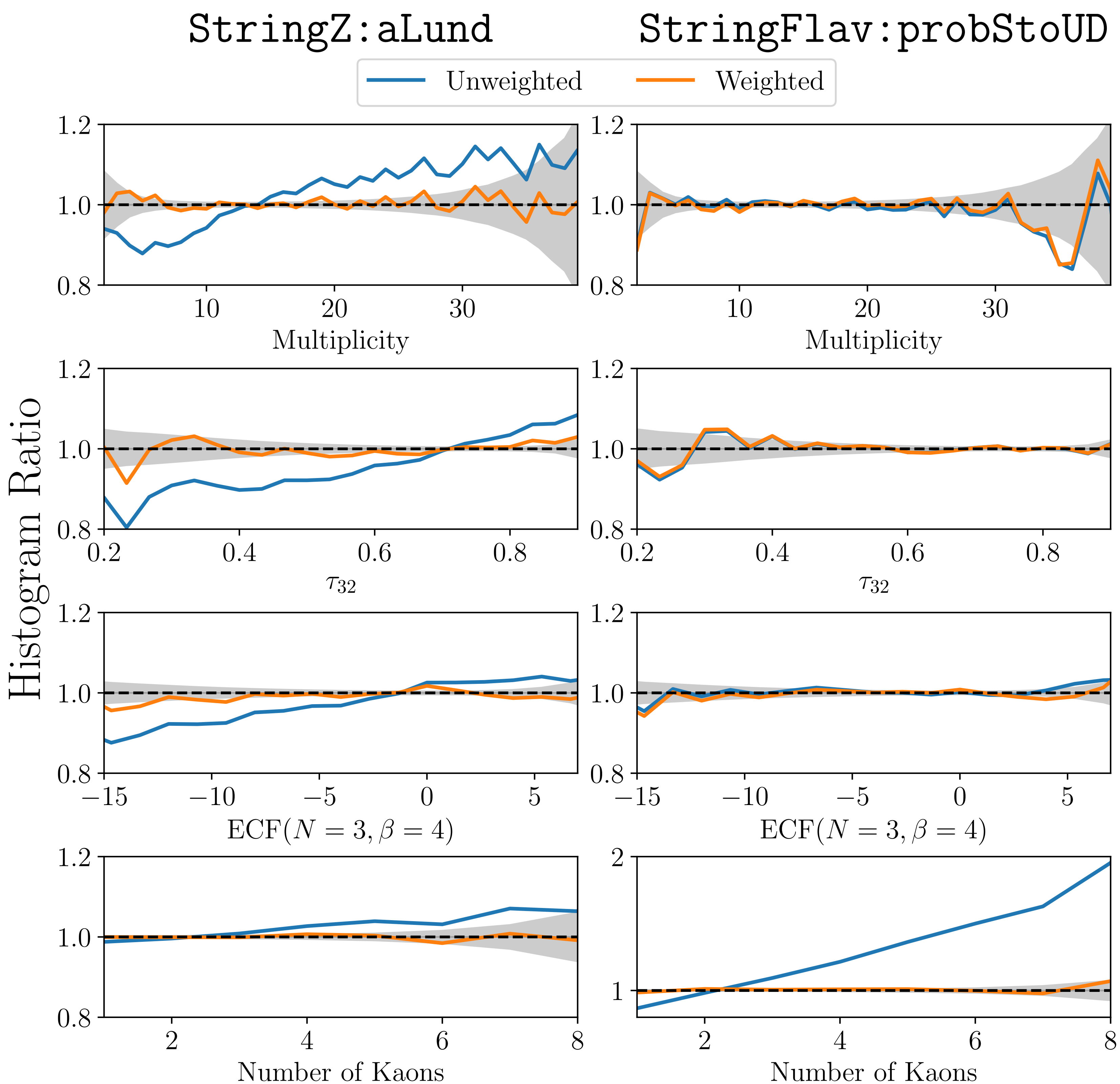


(# of particles)



(3-particle correlation function)

Achieving precision



Works also when the differences between the two simulations are small (left) or localized (right).

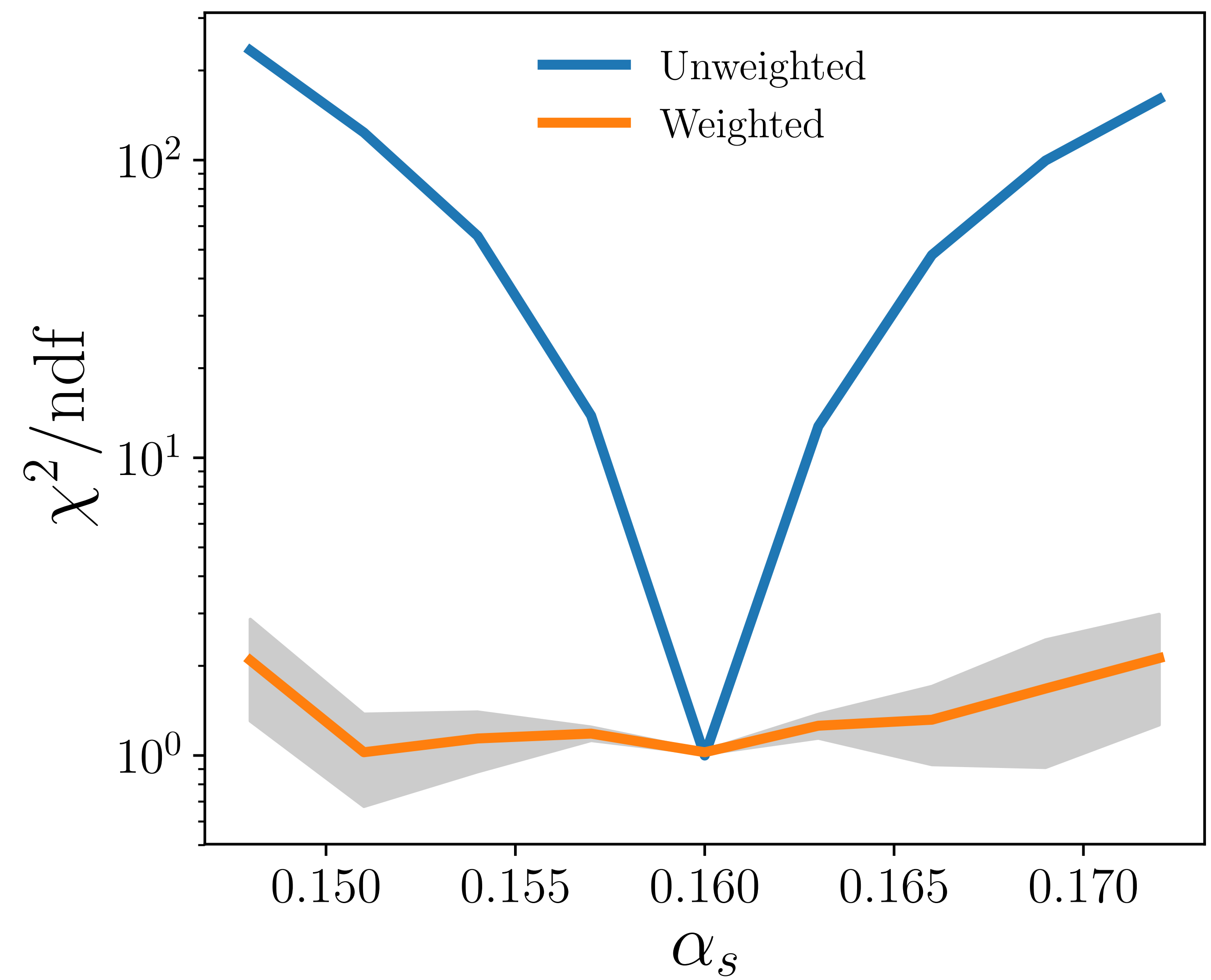
These are histogram ratios for a series of one-dimensional observables

Parameterized reweighting

What if we have a new simulation with multiple continuous parameters θ?

Easy - simply learn a parameterized classifier* !

…simply add the parameter as a feature to the network during training and let it learn to interpolate.



"fine structure constant" of the strong force

*see Cranmer, Pavez, Louppe, 1506.02169

Parameter estimation

What if we want to reweight with **pre-detector particles**, but fit to **detector-level objects**?

$$\theta^* = \underset{\theta'}{\mathrm{argmax}} \min_g \sum_{i \in \theta_0} \log g(x_{D,i}) \quad \text{[data]}$$

[reweighted simulation]

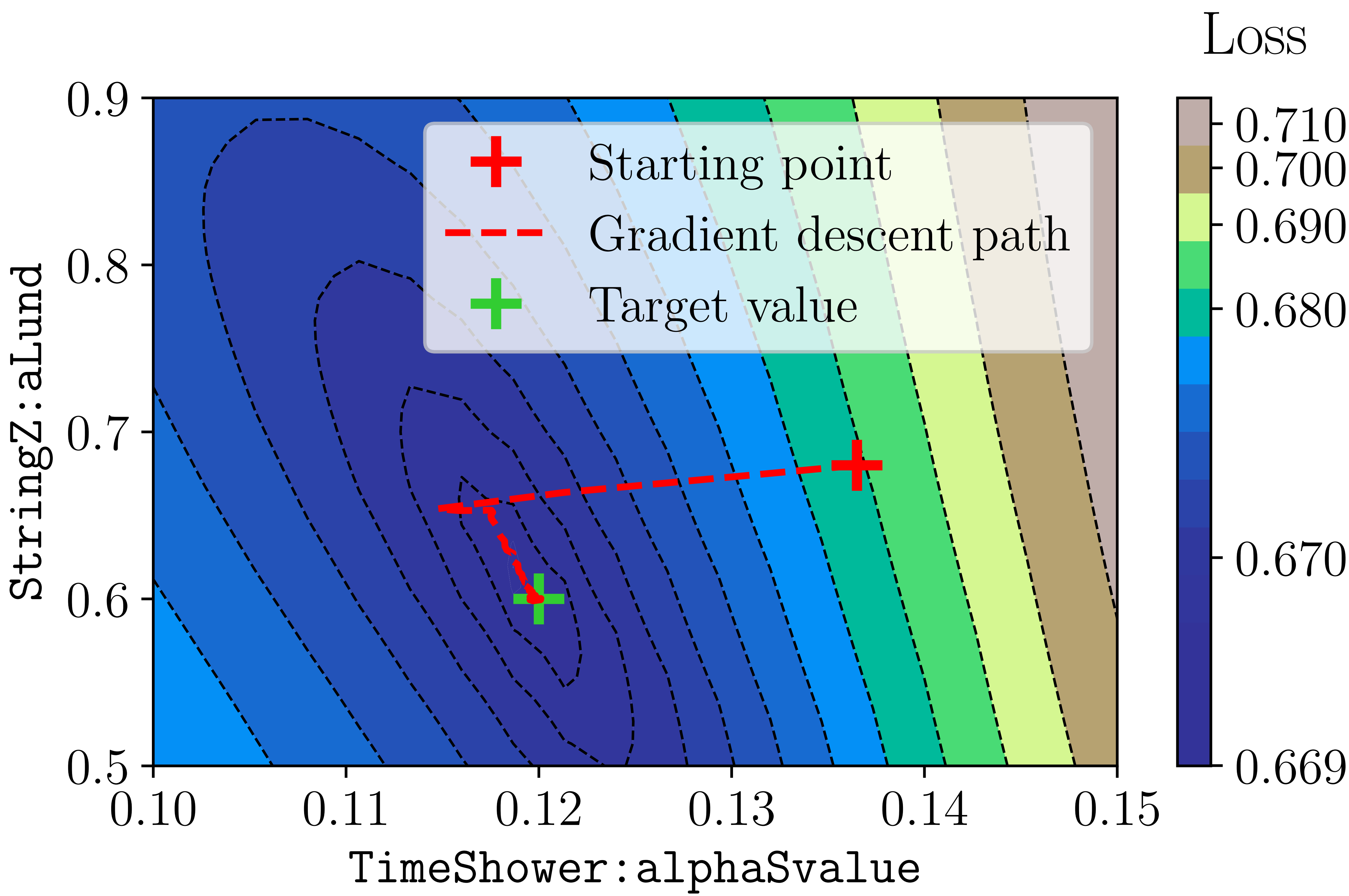
$$+ \sum_{i \in \theta} w(x_{T,i}, \theta) \log(1 - g(x_{D,i}))$$

$$\frac{f(x_T, \theta)}{1 - f(x_T, \theta)}$$

Intuition: reweight until you can't distinguish the data from the (reweighted) simulation!

Parameter estimation

Fit 3 (2 shown) parameters using the full phase space!



*a different method was used for this fit - it is not a minimax procedure but doesn't work at two levels … ask later for details

Parameter estimation

Mean and standard deviation over 20 runs:

	Parameter	Target value	Fit value
Val.	TimeShower:alphaSvalue	0.1200	0.1195 ± 0.0022
	StringZ:aLund	0.6000	0.6276 ± 0.0373
	StringFlav:probStoUD	0.1200	0.1203 ± 0.0071
Blinded	TimeShower:alphaSvalue	0.1700	0.1707 ± 0.0022
	StringZ:aLund	0.7500	0.7425 ± 0.0453
	StringFlav:probStoUD	0.1400	0.1422 ± 0.0065

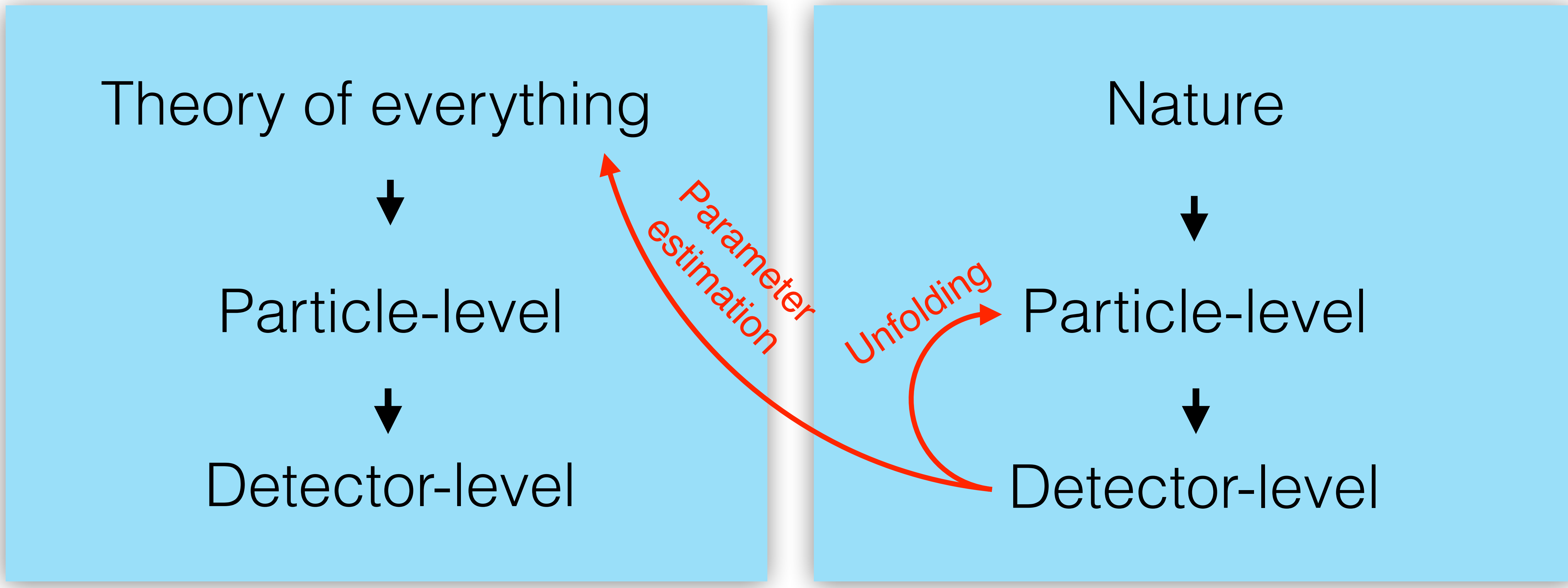
Similar spread

1D:

Parameter	Target value	Fit value
TimeShower:alphaSvalue	0.1600	0.1601 ± 0.0018
StringZ:aLund	0.8000	0.7980 ± 0.0257
StringFlav:probStoUD	0.2750	0.2754 ± 0.0065

The meaning of this “uncertainty” is discussed later.

Unfolding



Can we use this technique to take the full detector-level phase space and correct it to the full particle-level phase space?

…high-dimensional, unbinned unfolding!

Full phase space unfolding: OmniFold

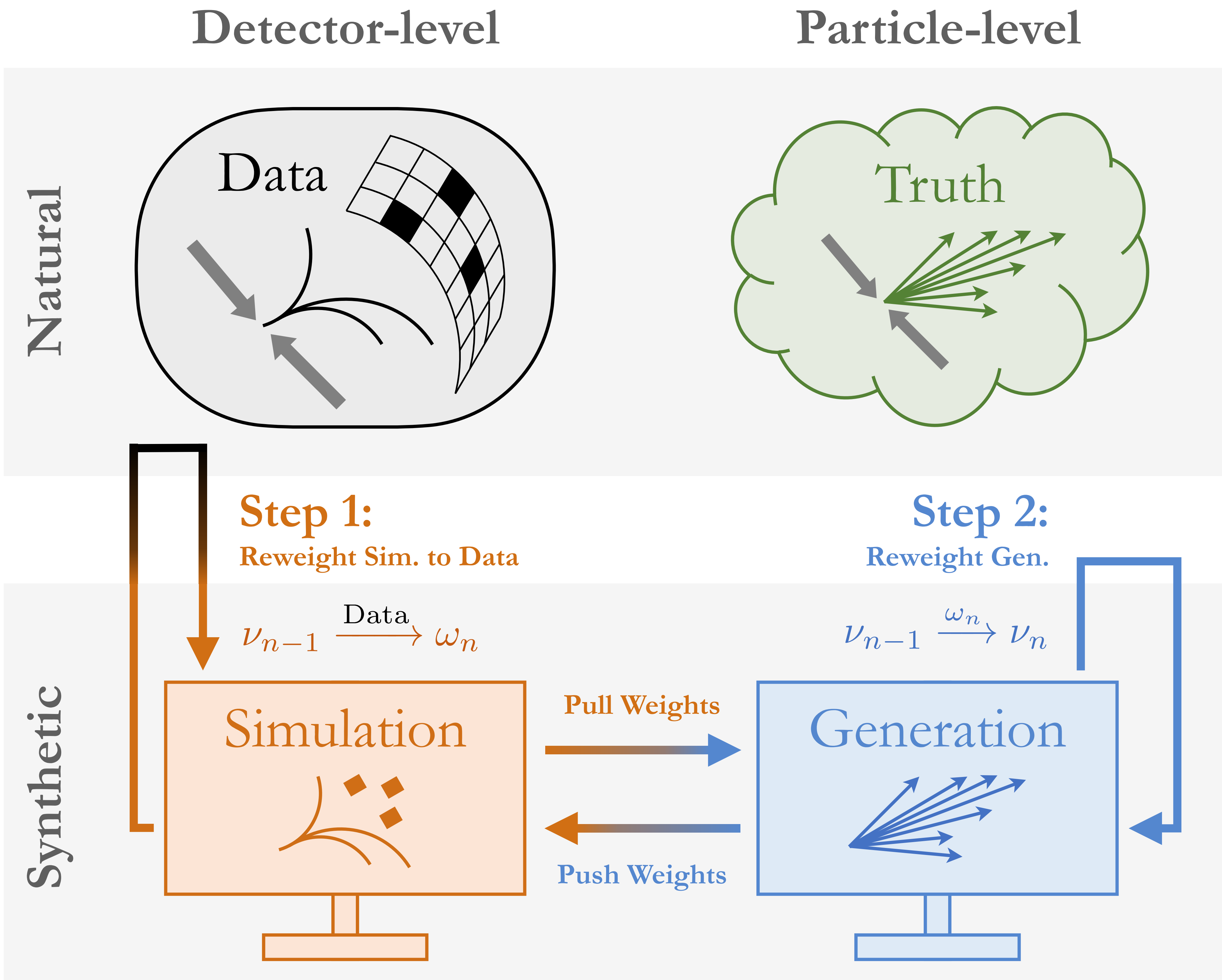
Emily Dickinson, #975

The Mountain sat upon the Plain
In his tremendous Chair –
His observation omnifold,
His inquest, everywhere –

The Seasons played around his knees
Like Children round a sire –
Grandfather of the Days is He
Of Dawn, the Ancestor –

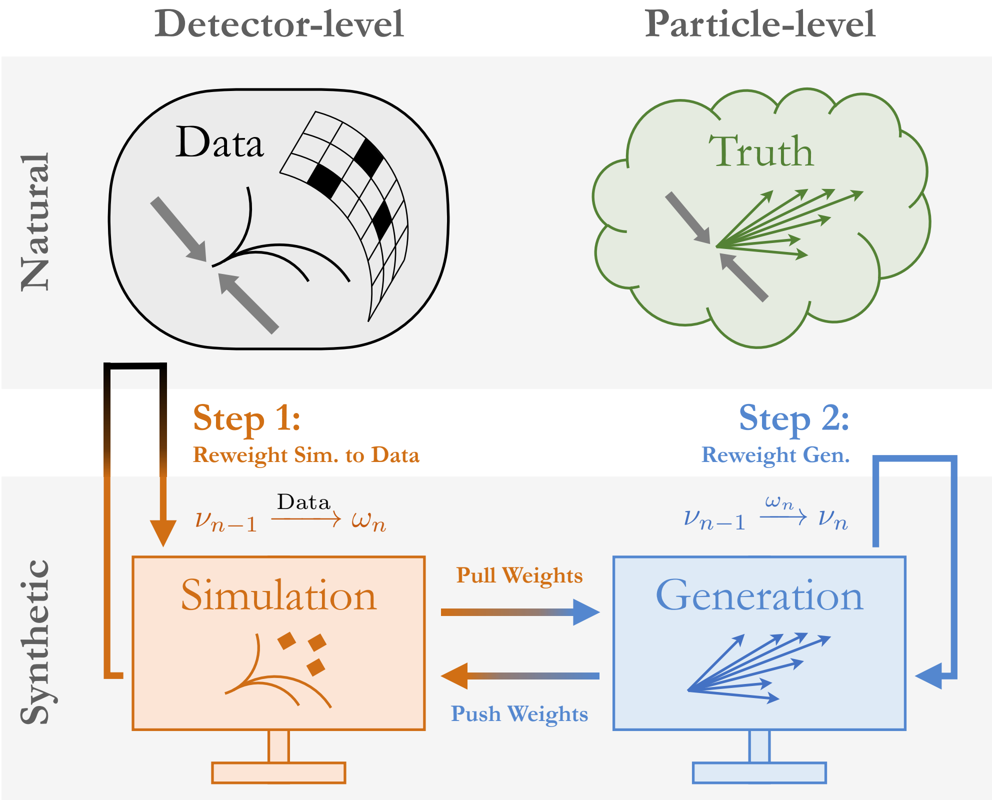
A. Andreassen, E. Metodiev, P. Komiske, **BPN**, J. Thaler, 1911.09107

Full phase space unfolding: OmniFold



A. Andreassen, E. Metodiev, P. Komiske, **BPN**, J. Thaler, 1911.09107

Full phase space unfolding: OmniFold



Notation: m = measured, t = true

$$L[(w,X),(w',X')](x) = \frac{p_{(w,X)}(x)}{p_{(w',X')}(x)}$$

(accomplish with a classifier, as before)

$$\nu_0^{\text{push}}(m) = \nu_0(t) \qquad \omega_n^{\text{pull}}(t) = \omega_n(m)$$

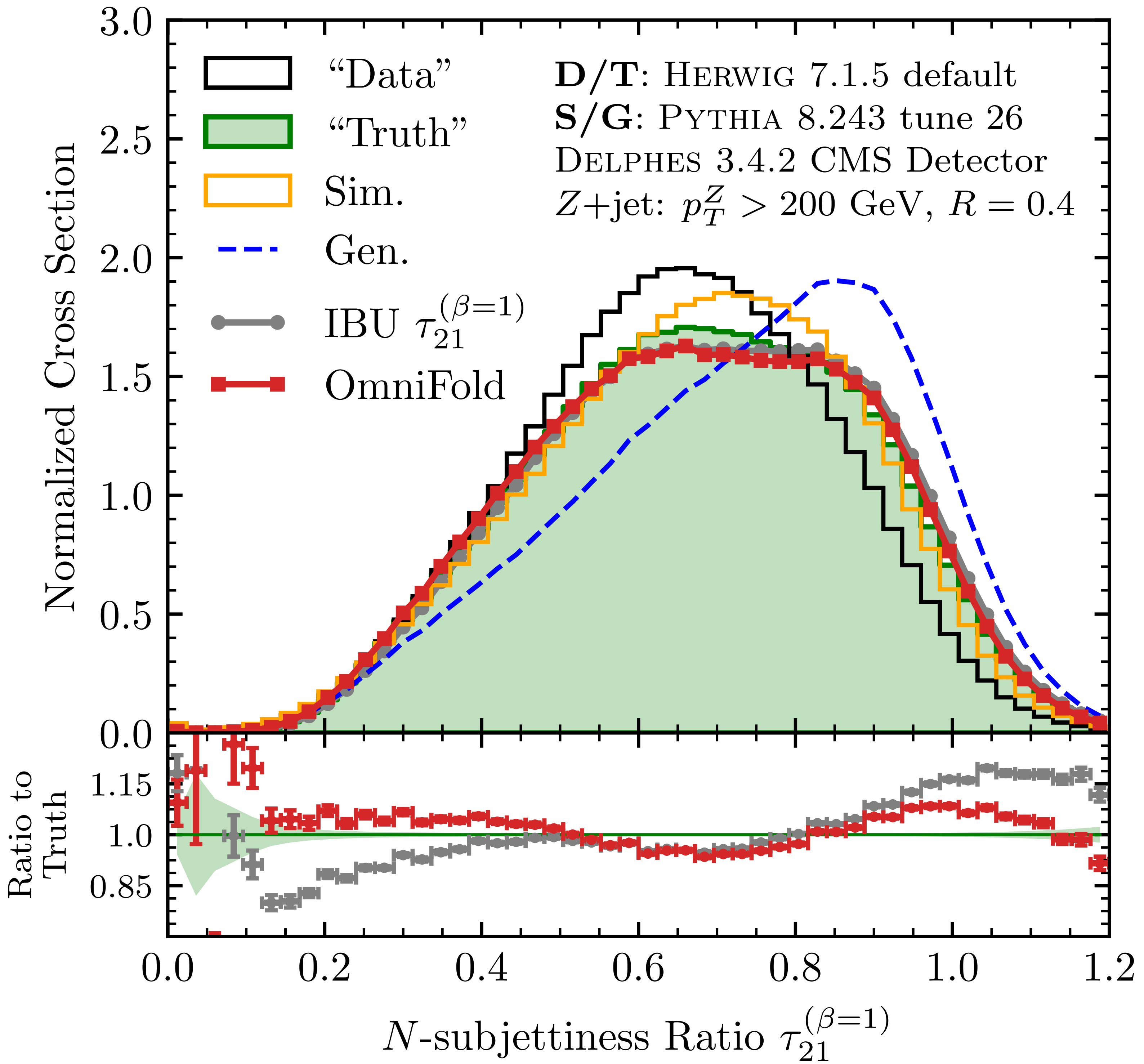
(these are not functions, since t → m is not 1:1)

Iterate:

1. $\omega_n(m) = \nu_{n-1}^{\text{push}}(m)\, L[(1, \text{Data}), (\nu_{n-1}^{\text{push}}, \text{Sim.})](m),$

2. $\nu_n(t) = \nu_{n-1}(t)\, L[(\omega_n^{\text{pull}}, \text{Gen.}), (\nu_{n-1}, \text{Gen.})](t).$

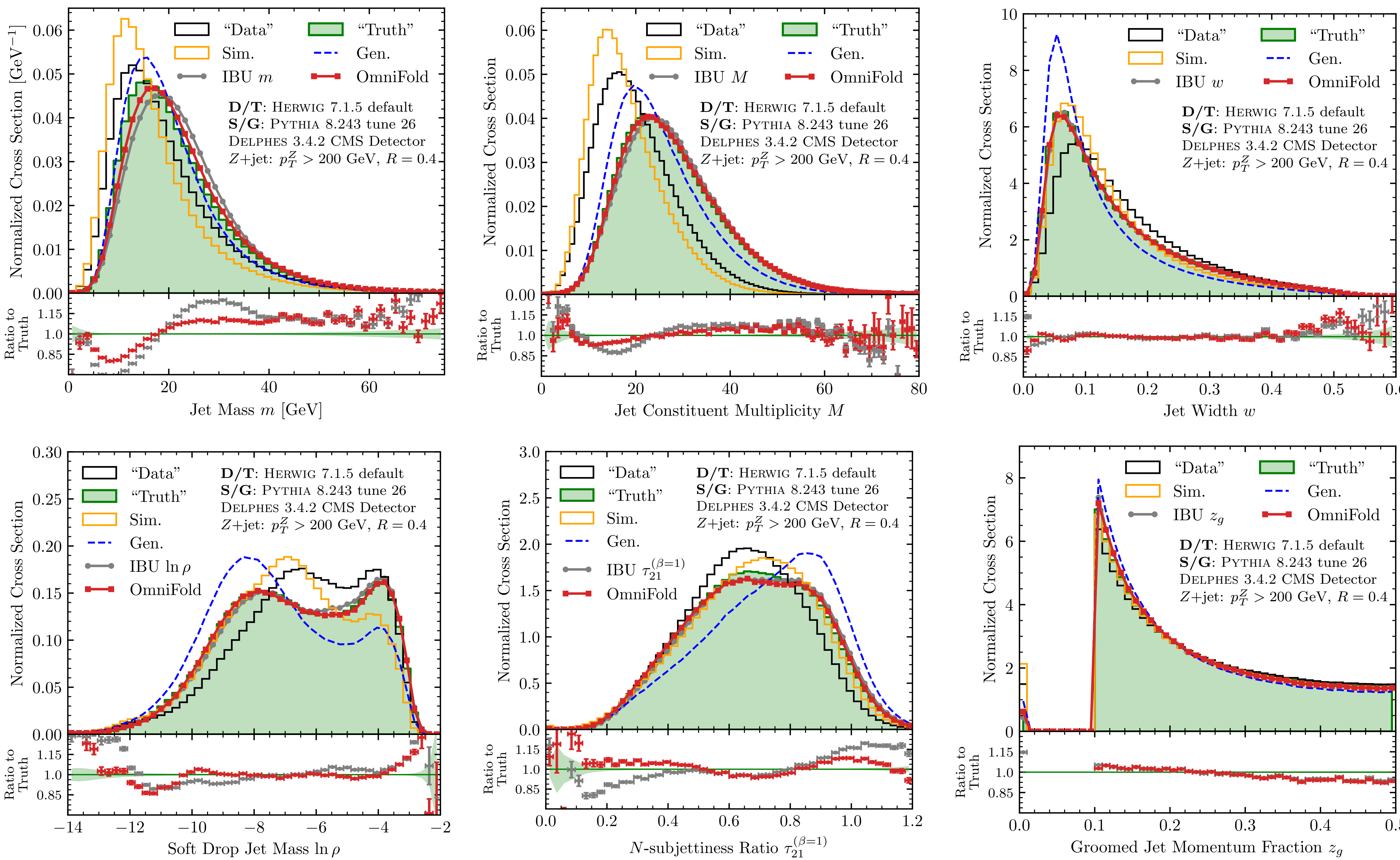
OmniFold



IBU = iterative Bayesian; one of the most widely used methods

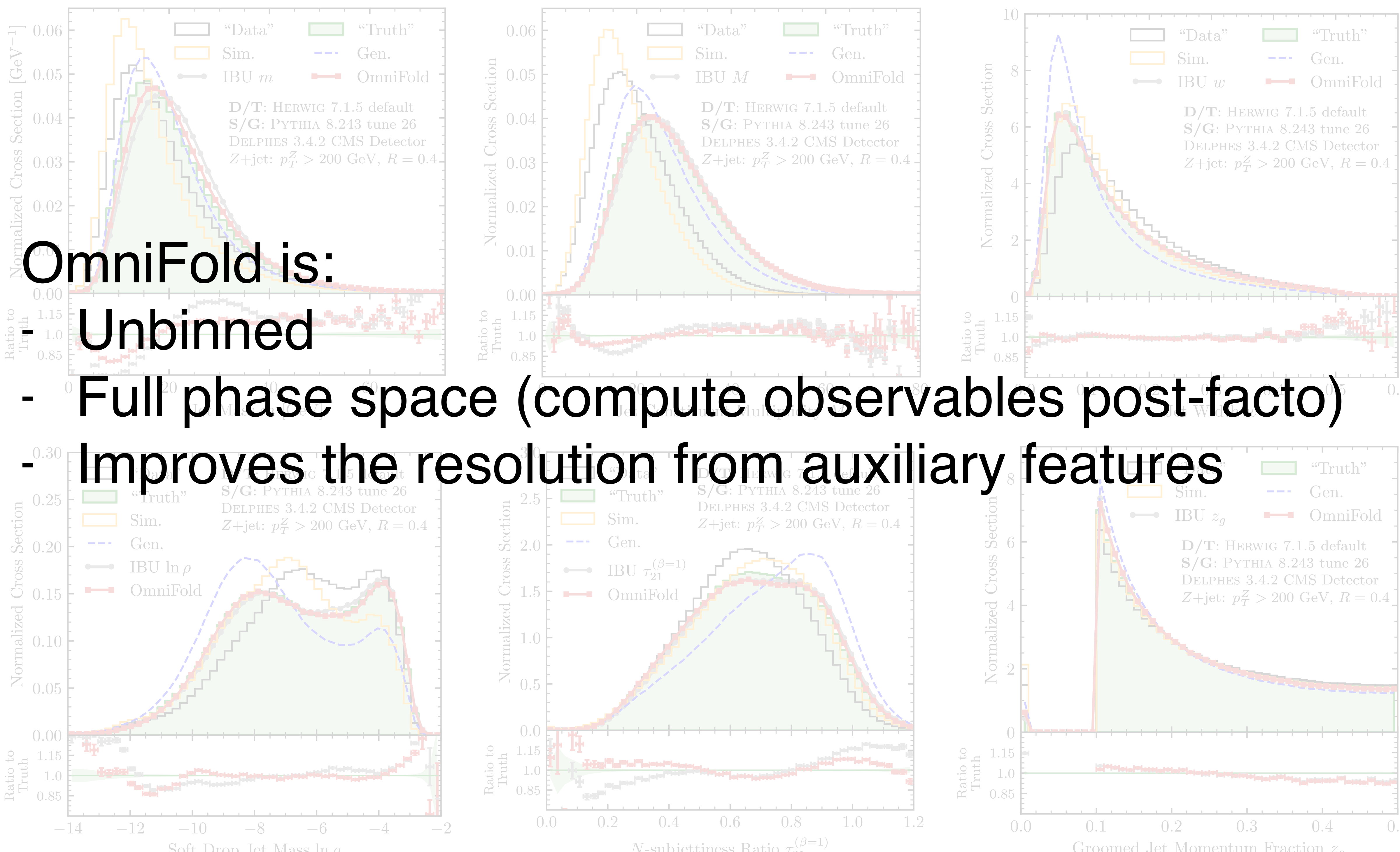
OmniFold

One unfolding for OmniFold, 6 (one per) for IBU.



OmniFold

One unfolding for OmniFold, 6 (one per) for IBU.

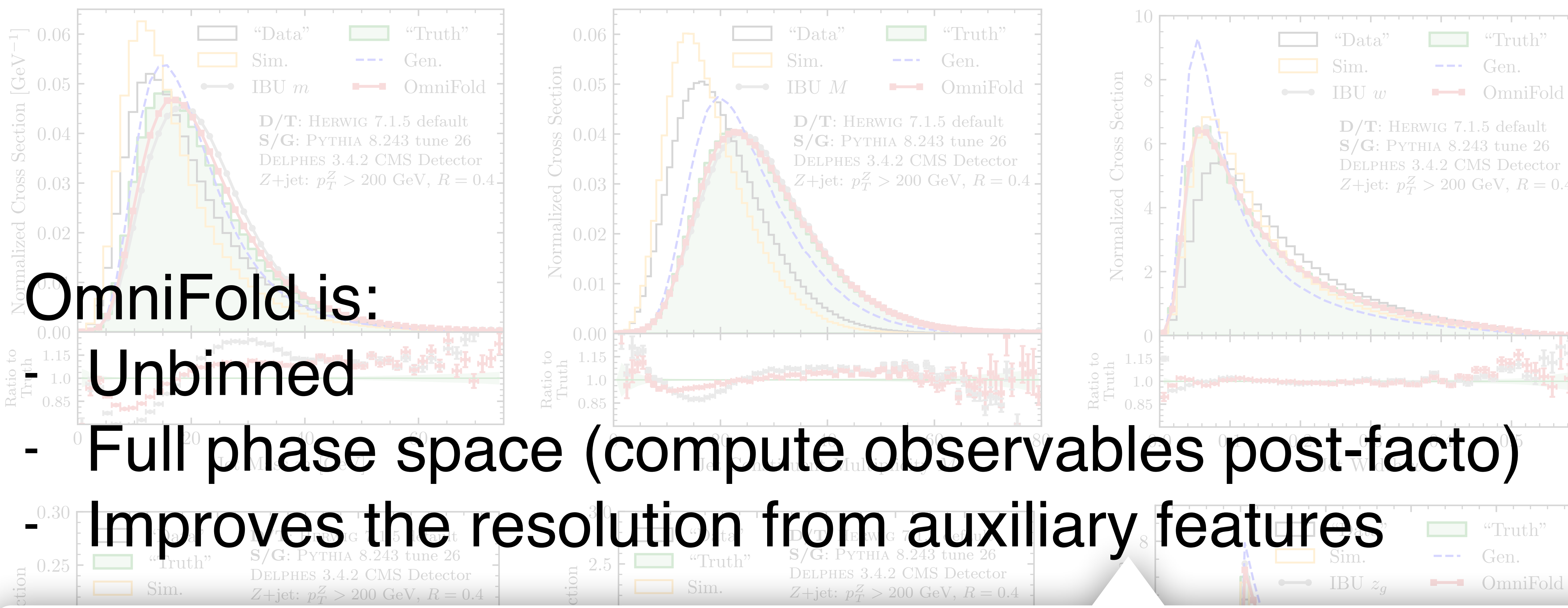


OmniFold is:

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

OmniFold

One unfolding for OmniFold, 6 (one per) for IBU.



OmniFold is:

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

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

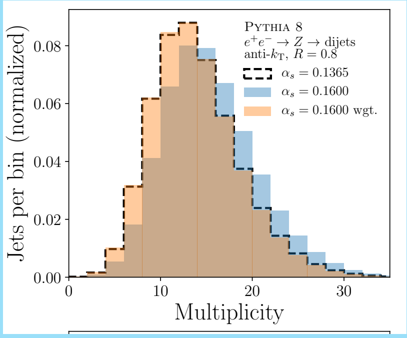
$$X \sim \mathcal{N}(\mu, \sigma)$$

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

Intermediate summary

New simulations

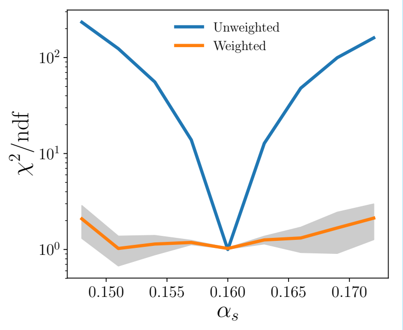
morph one simulation into another



“DCTR”

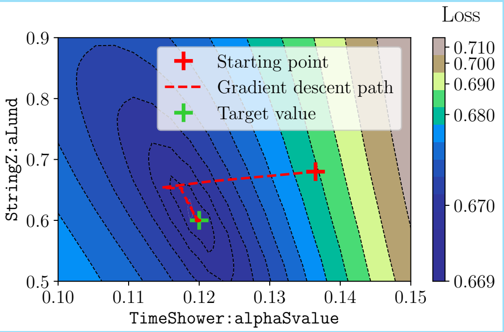
Continuous variations

learn the dependence on parameters



Parameter estimation

use classification loss to fit parameters



A. Andreassen and **BPN**, 1909.03081

Unfolding

iterate the morphing to remove distortions

A. Andreassen et al., 1909.03081

Uncertainties

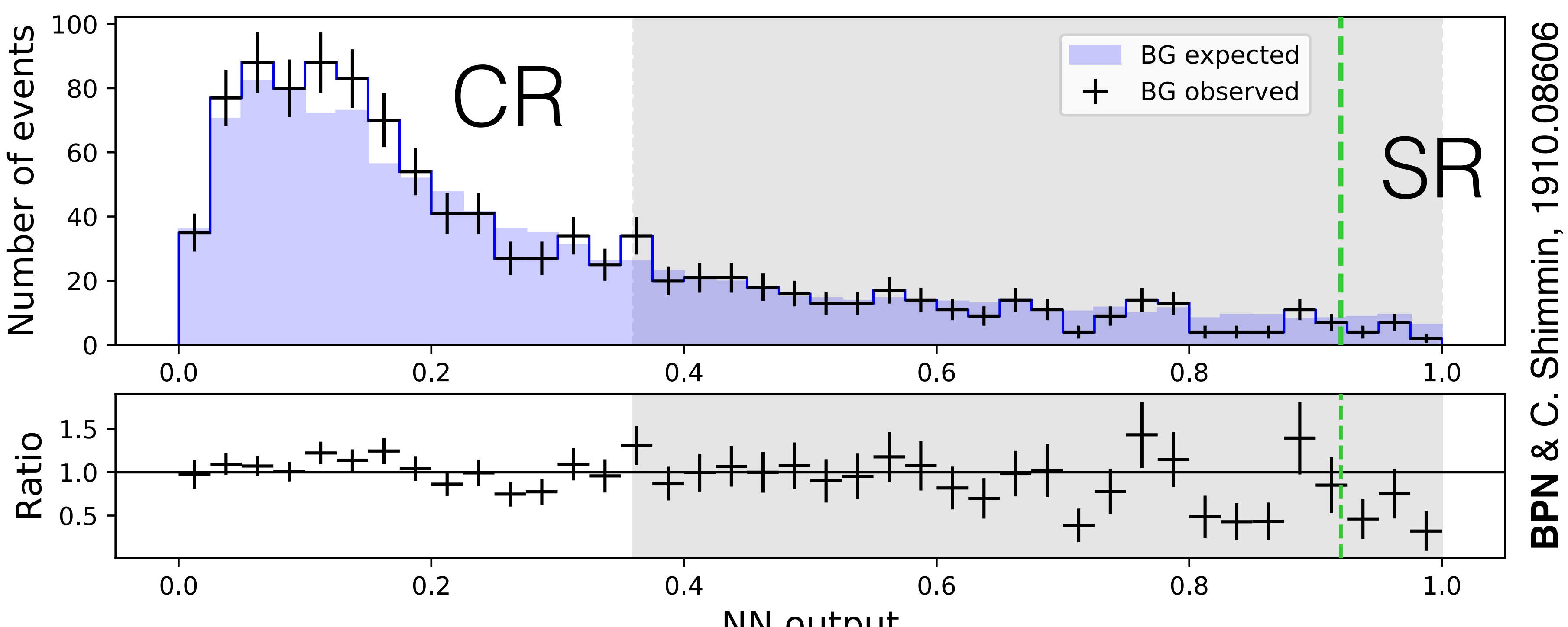
"But what are the uncertainties on the NN"?

- question asked by every reviewer

Setup

To keep things simple, let's use the following common example:

1. Train a classifier (in sim.) for signal vs. background.
2. Define a control region and a signal region using (1).
3. Normalize simulation in CR.
4. Compare data and scaled simulation in SR.
5. Significantly different? go to Stockholm; else 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 scaled sim. is not correct.

Uncertainties for a NN-based analysis

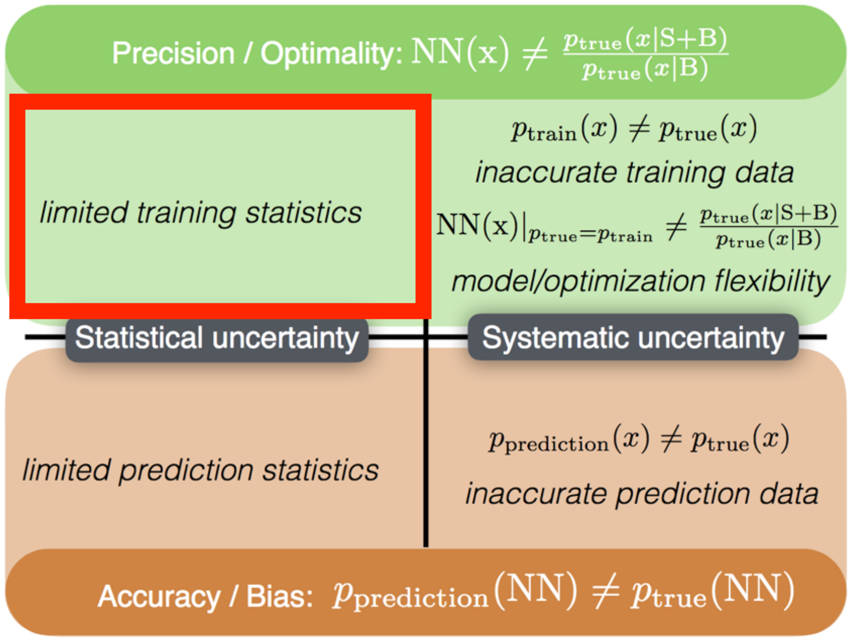
Precision / Optimality: $\mathrm{NN}(\mathrm{x}) \neq \frac{p_{\mathrm{true}}(x|\mathrm{S+B})}{p_{\mathrm{true}}(x|\mathrm{B})}$

Statistical uncertainty	Systematic uncertainty
limited training statistics	$p_{\mathrm{train}}(x) \neq p_{\mathrm{true}}(x)$ *inaccurate training data* $\mathrm{NN}(\mathrm{x})\vert_{p_{\mathrm{true}}=p_{\mathrm{train}}} \neq \frac{p_{\mathrm{true}}(x\vert\mathrm{S+B})}{p_{\mathrm{true}}(x\vert\mathrm{B})}$ *model/optimization flexibility*
limited prediction statistics	$p_{\mathrm{prediction}}(x) \neq p_{\mathrm{true}}(x)$ *inaccurate prediction data*

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

BPN, 1909.03081

How to estimate precision stat. uncerts.

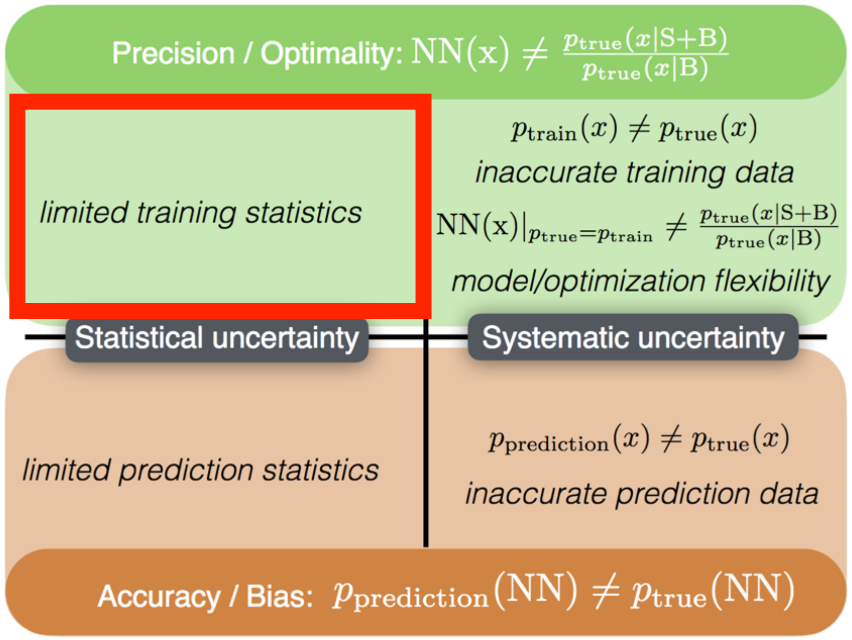


You can always accomplish this by bootstrapping: making pseudo-datasets from resampling and then retraining.

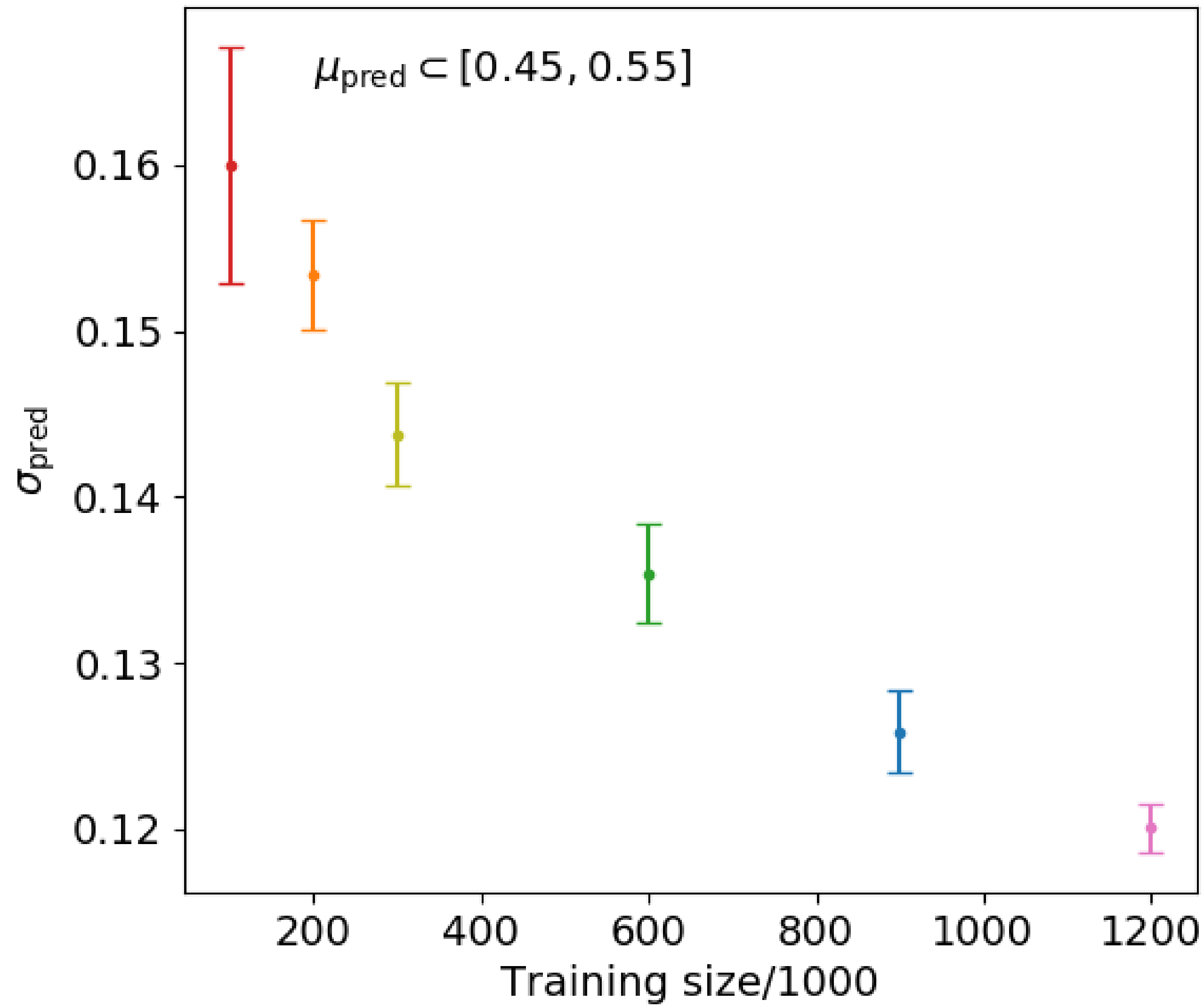
It is important to fix the NN initialization so that you are not also testing your sensitivity to that.

This can be painful because it requires retraining many NNs.

How to estimate precision stat. uncerts.

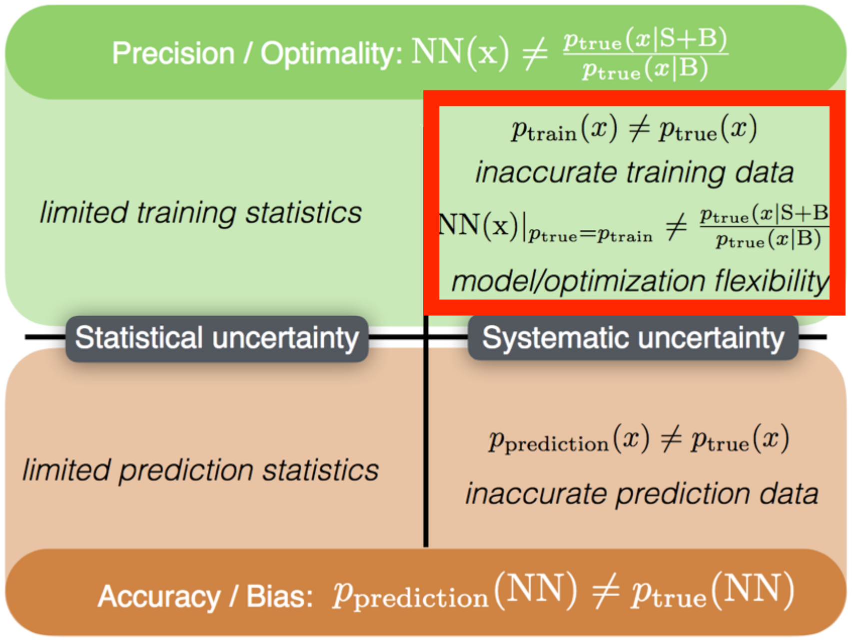


Alternative: train **one** Bayesian NN?!



S. Bollweg, M. Haußmann, G. Kasieczka, M. Luchmann, T. Plehn, J. Thompson, 1904.10004

How to estimate precision syst. uncerts.

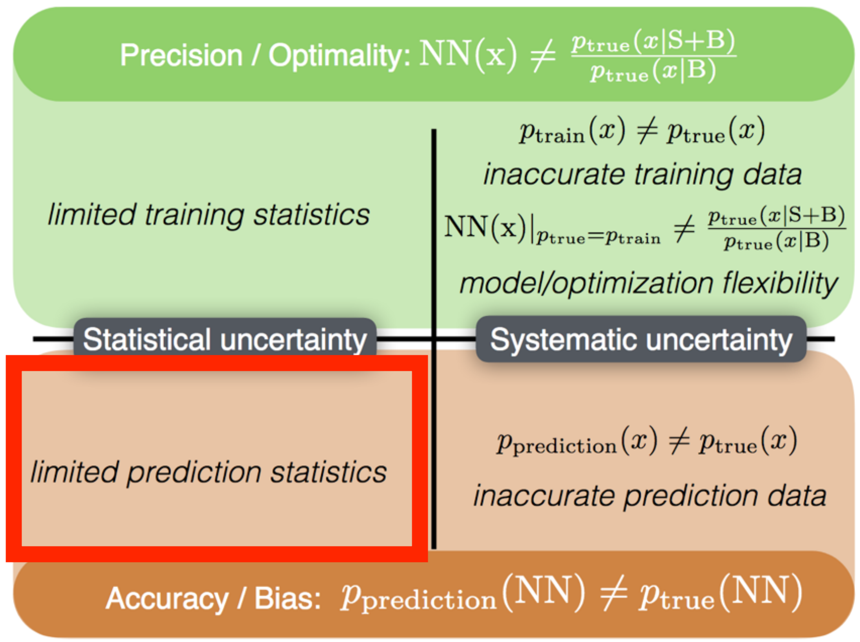


As with all systematic uncertainties, this is hard to quantify.

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.)

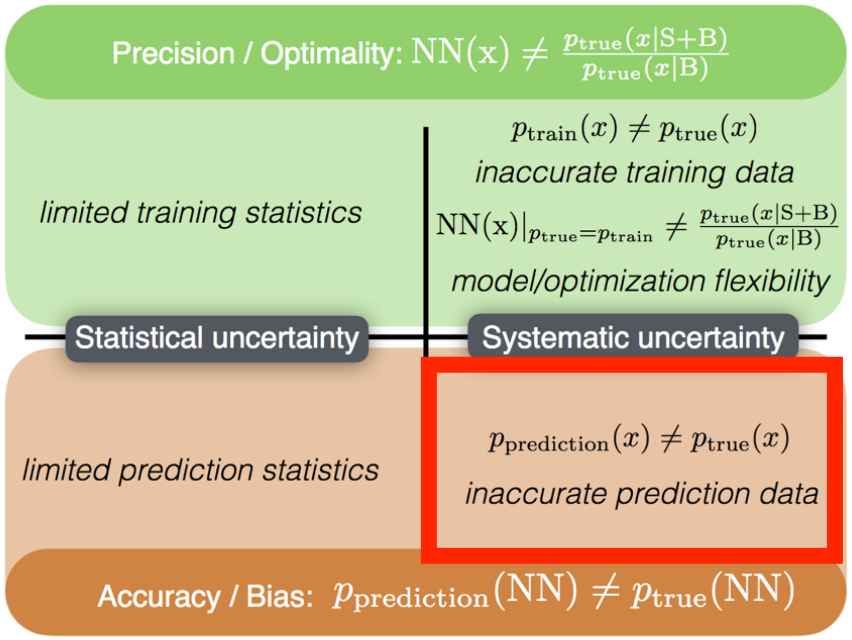
How to estimate bias stat. uncerts.



This uncertainty is post-training - Bayesian NNs unfortunately won't help.

Unfortunately, this is often not small given that we are now probing extreme final states and have a limited computing budget.

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 the jet energy is measured and evaluated per jet.

What about physics modeling uncertainties where we usually have a two-point comparison? (e.g. Pythia versus Herwig)

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?

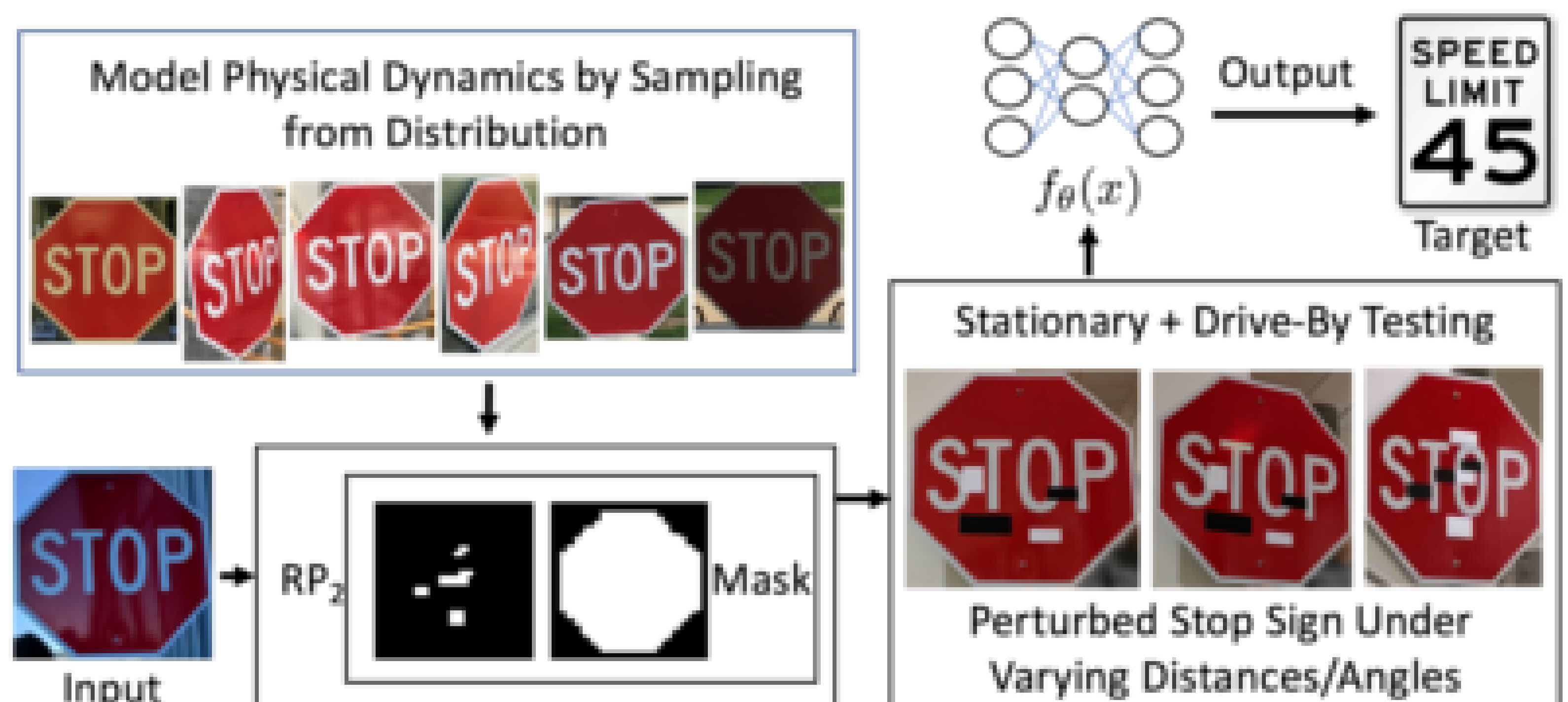
Answer: borrow tools from AI Safety

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 = jet (in all of its high-dimensional glory)

f = fixed classifier for signal vs. background

Loss

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

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

BPN & C. Shimmin, 1910.08606

g is a learned NN that maps J to J + δJ.

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

Bounding high-dim. uncerts: strategy

J = jet (in all of its high-di

f = fixed classifier for signa

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

$$\mathcal{L}_{\text{bg}} = \lambda_{\text{cls}}(f(J) - f(g(J)))$$

$$+ \sum_i \lambda_{\text{obs}}^{(i)}(\mathcal{O}^{(i)}(J) - \mathcal{O}^{(i)}(g(J))^2$$

BPN & C. Shimmin, 191

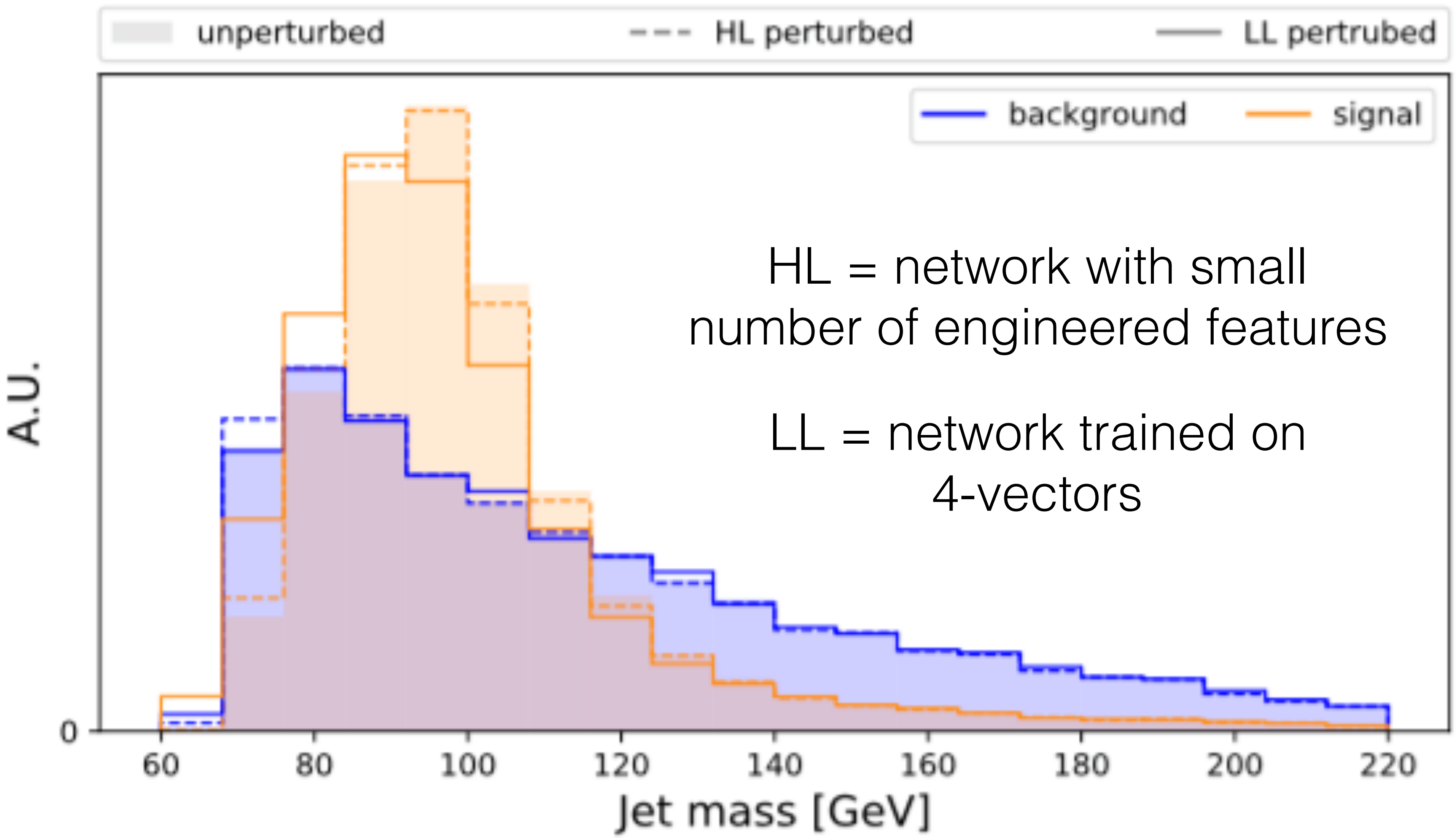
Fun fact: this requires computing ***O*** in the NN - now have GPU implementations of many standard observables!

g is a learned NN that maps J to J + δJ.

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

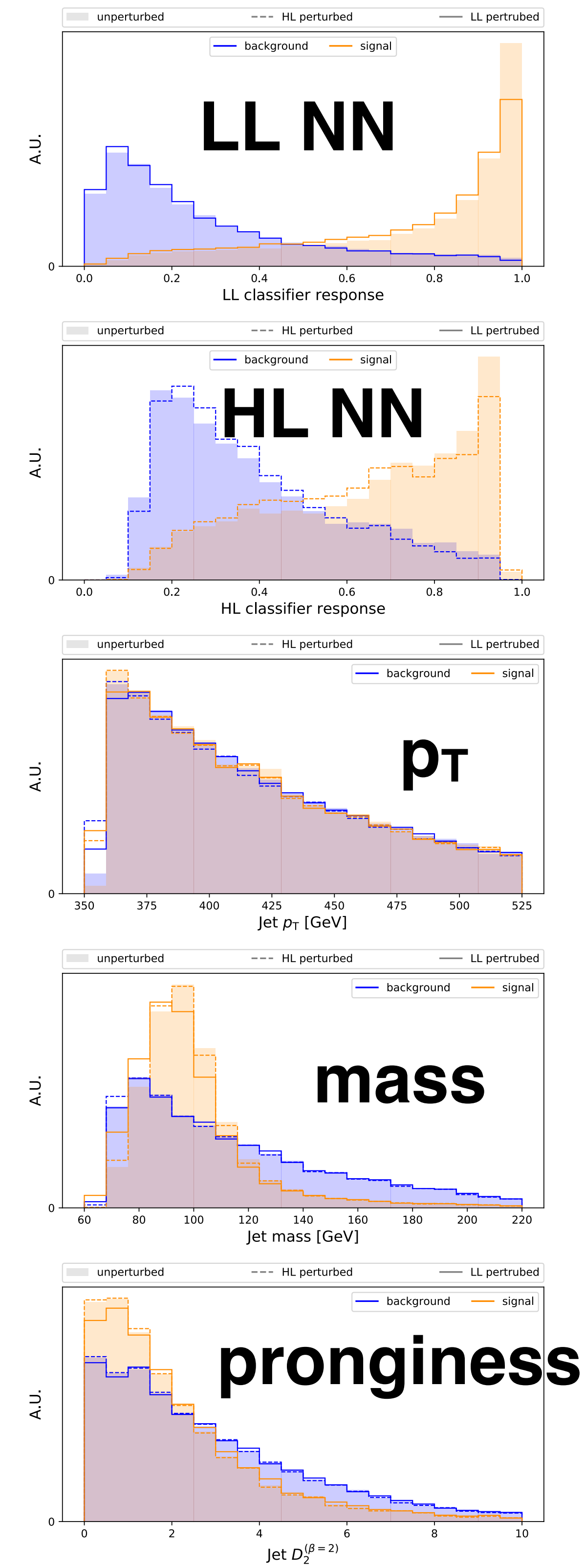
High-dimensional Uncertainty

Example case: Boosted Z's versus QCD



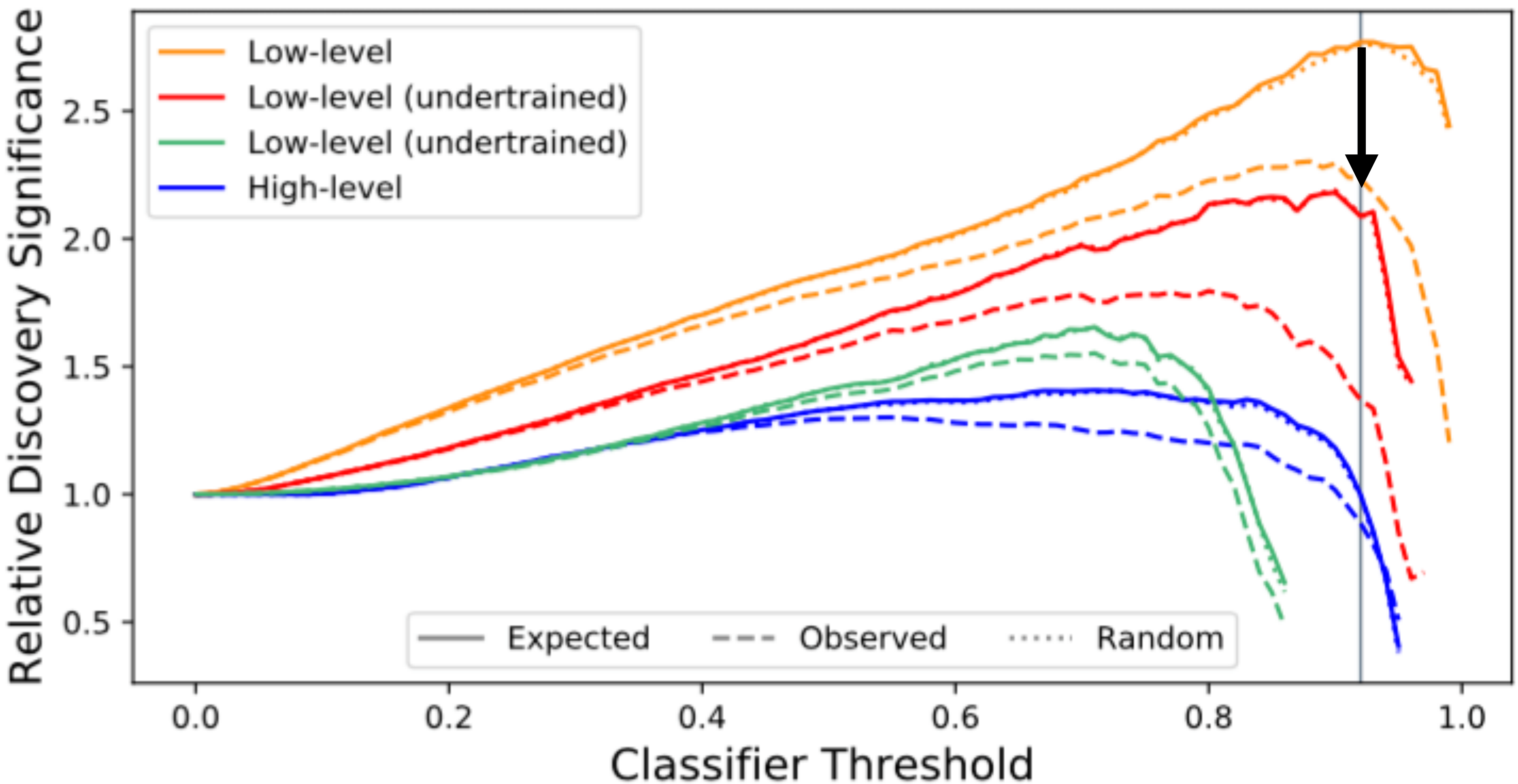
HL = network with small number of engineered features

LL = network trained on 4-vectors



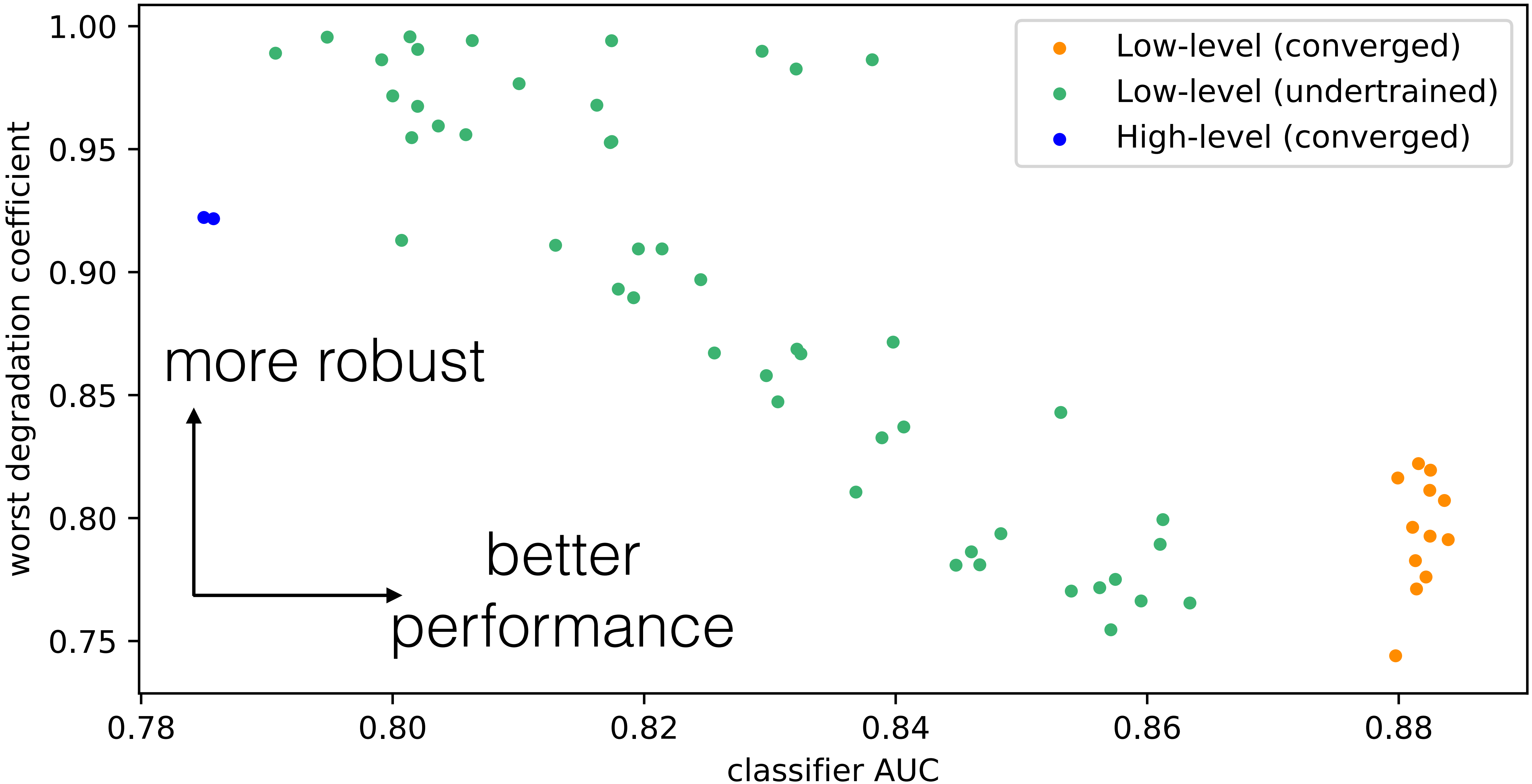
It is "easy" to preserve the blue and modify the orange.

High-dimensional Uncertainty



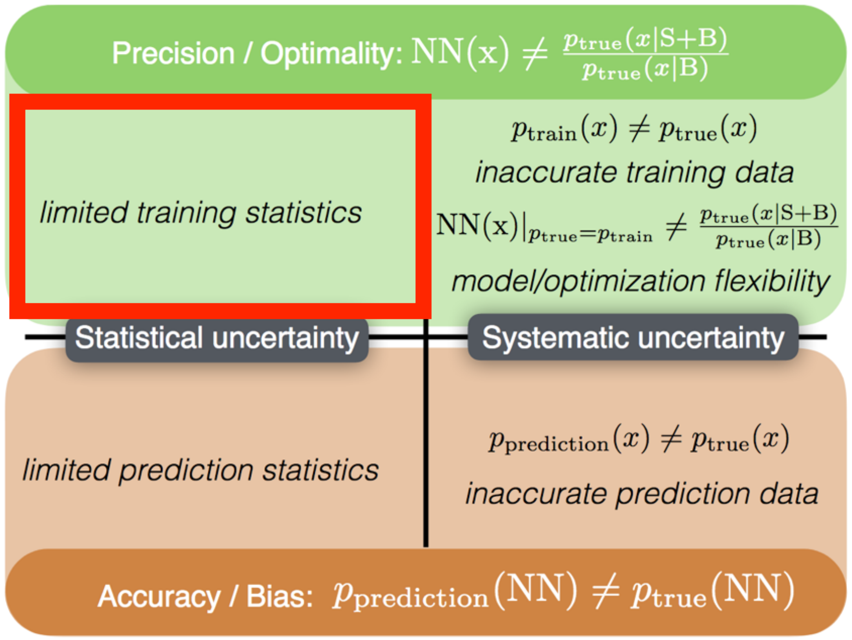
High-dimensional Uncertainty

Under training may help with robustness:



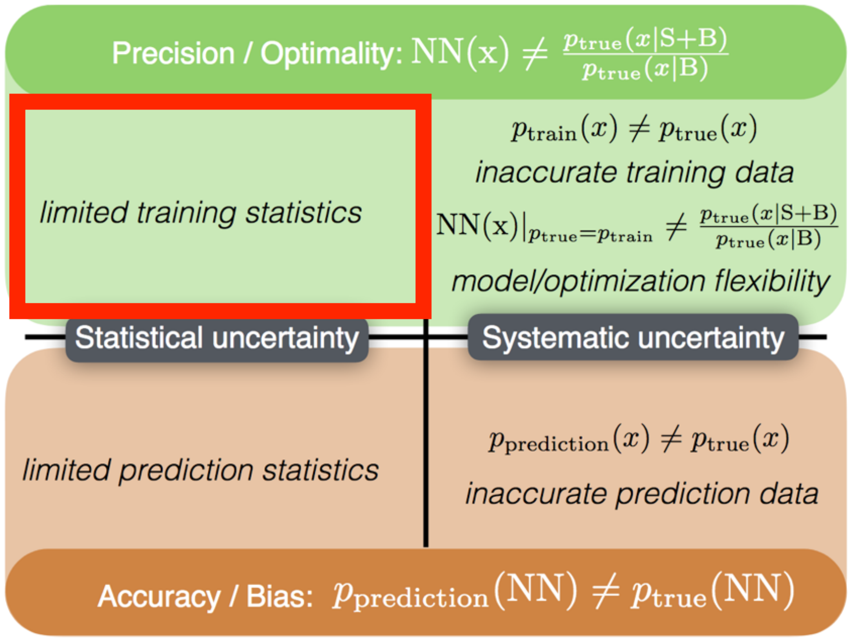
Perturbed significance improvement / nom. sig. improvement

How to reduce precision stat. uncerts.



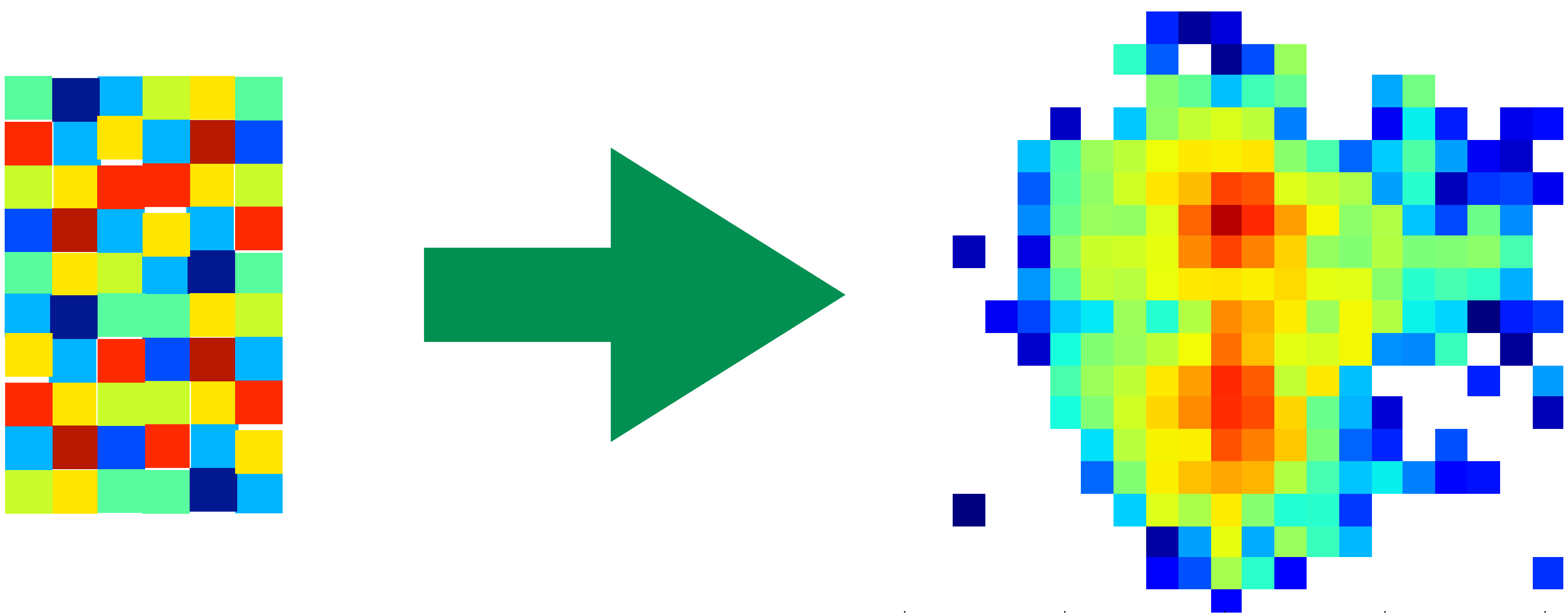
Train with more events!

How to reduce precision stat. uncerts.



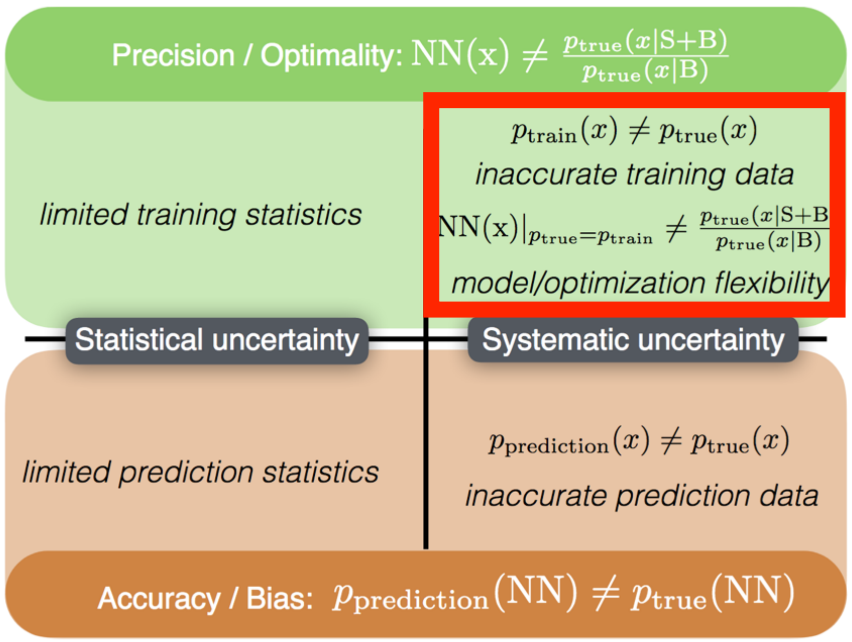
Train with more events!

…maybe use NN's to help with that



M. Paganini, L. de Oliveira, **BPN**, PRL 120 (2018) 042003 + many others (including some from other people here!)

How to reduce precision syst. uncerts.



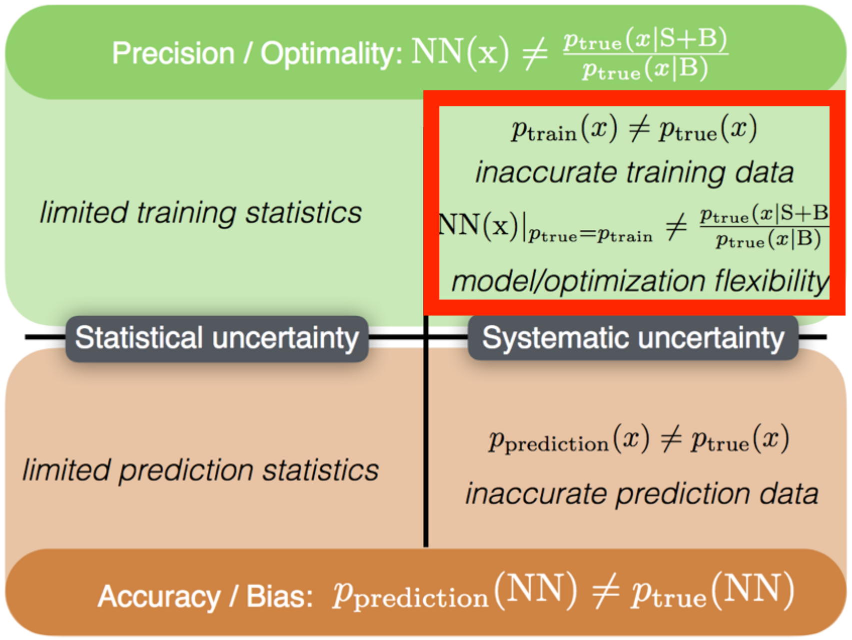
You might be tempted to force the NN to not depend on some uncertain parameters.

There are many ways to do this, e.g. adversarial techniques[1] or DisCo[2]

Unless this is needed to estimate the background[3], this is usually suboptimal and may not even reduce the uncertainty.

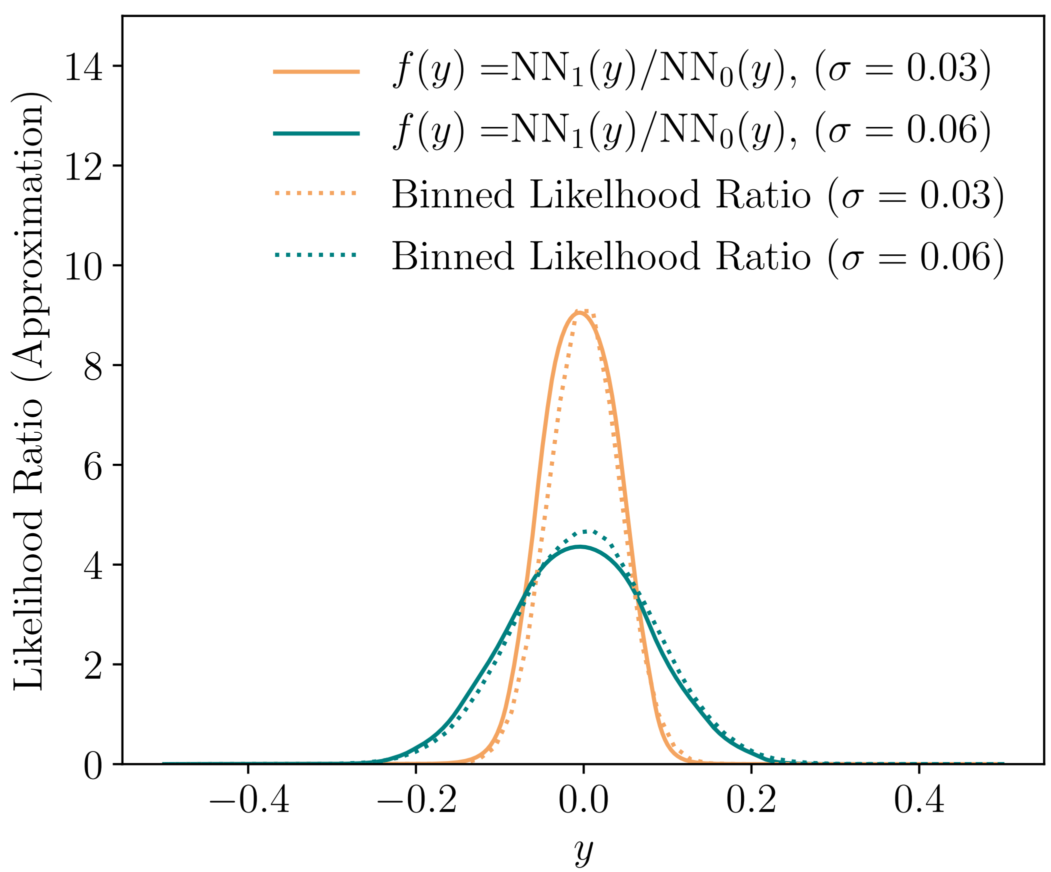
1. G. Louppe, M. Kagan, K. Cranmer, 1611.01046
2. Gregor Kasieczka and D. Shih, 2001.05310
3. C. Shimmin et al. Phys. Rev. D 96, 074034 (2017), and many others including (2)

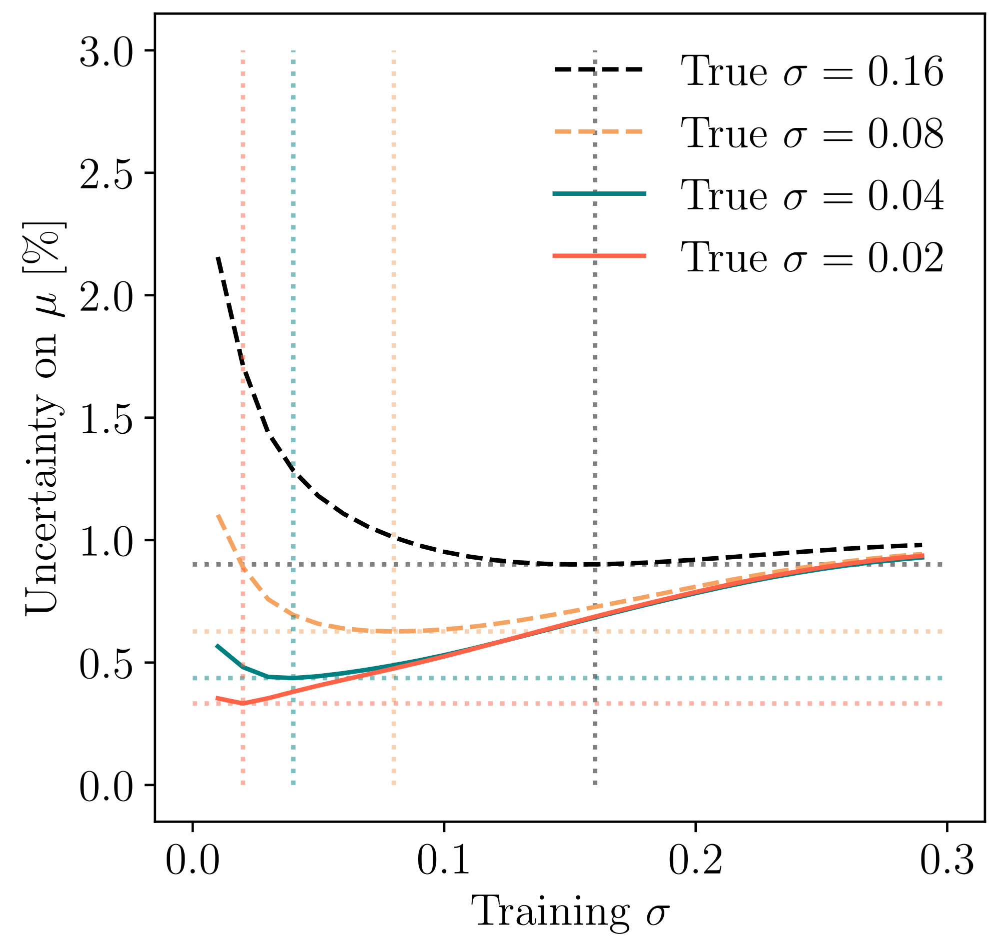
How to reduce precision syst. uncerts.



Profiling instead of pivoting:

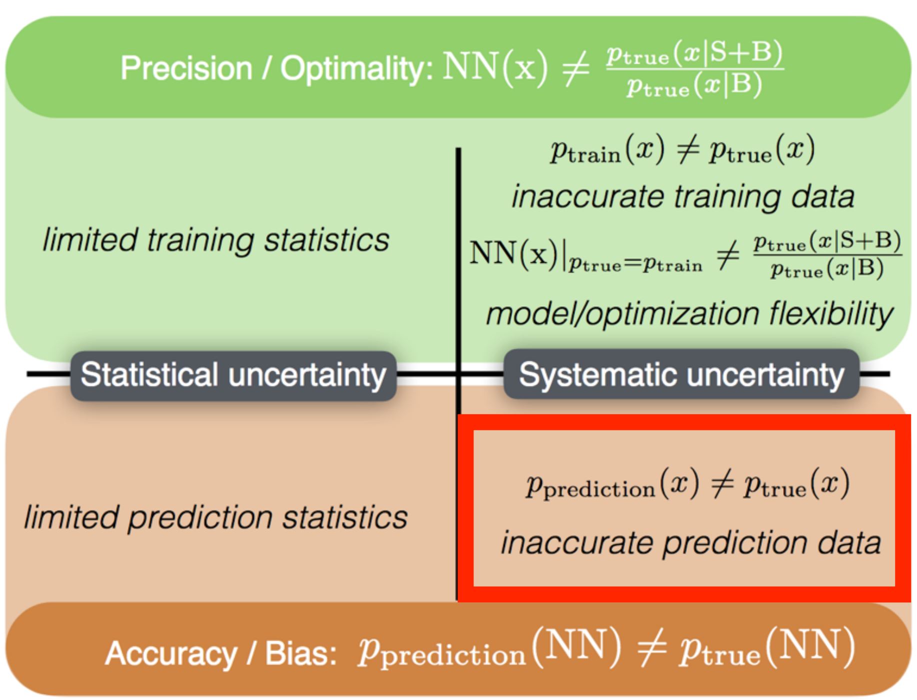
Better to do the opposite: let your NN **depend explicitly** on uncertainty quantities and then **profile them**!





How to get around high-D bias uncerts?

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

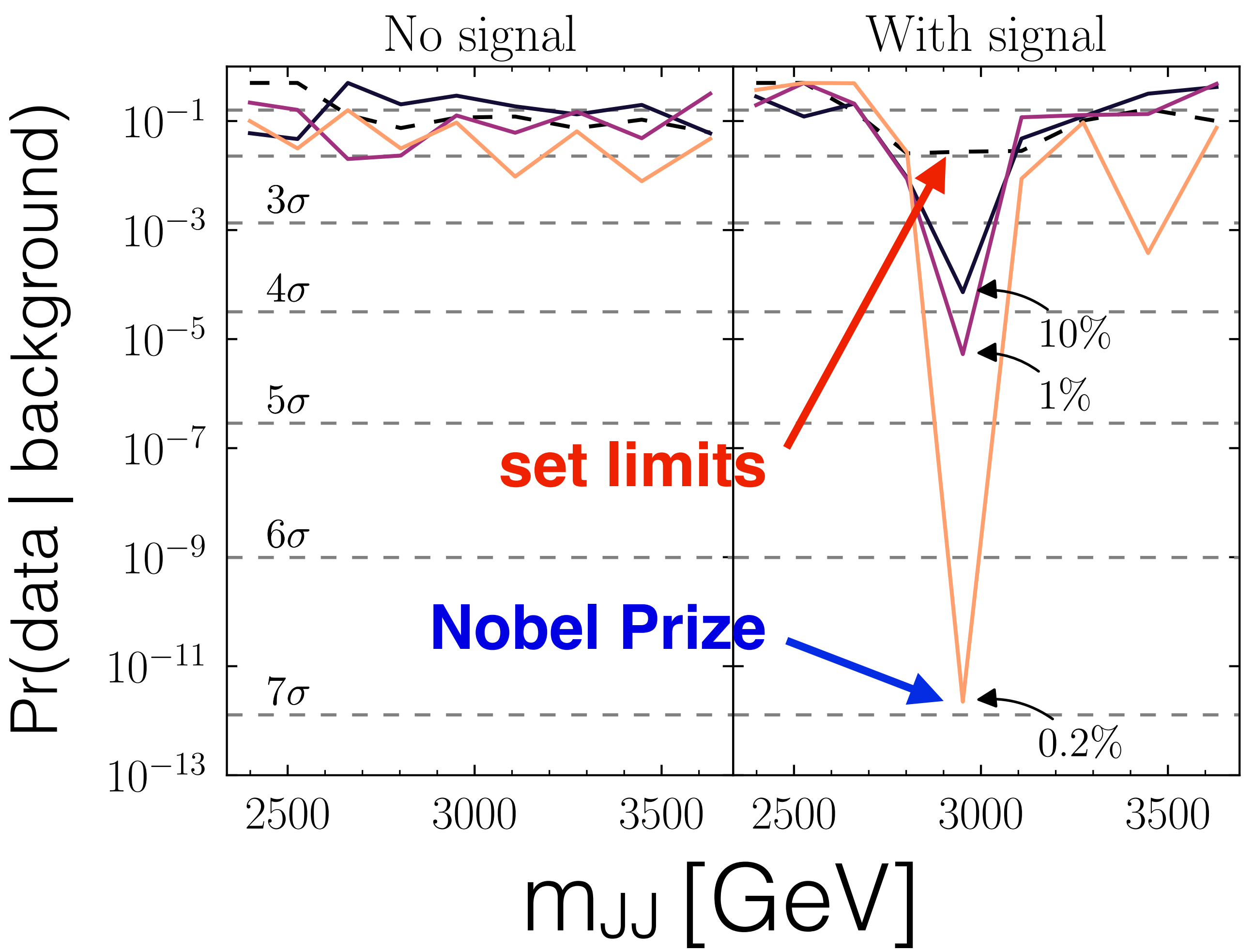


In my opinion, this is **THE** biggest challenge with deploying NN-based analyses … solving it will require hard physics work.

How to get around high-D bias uncerts?

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

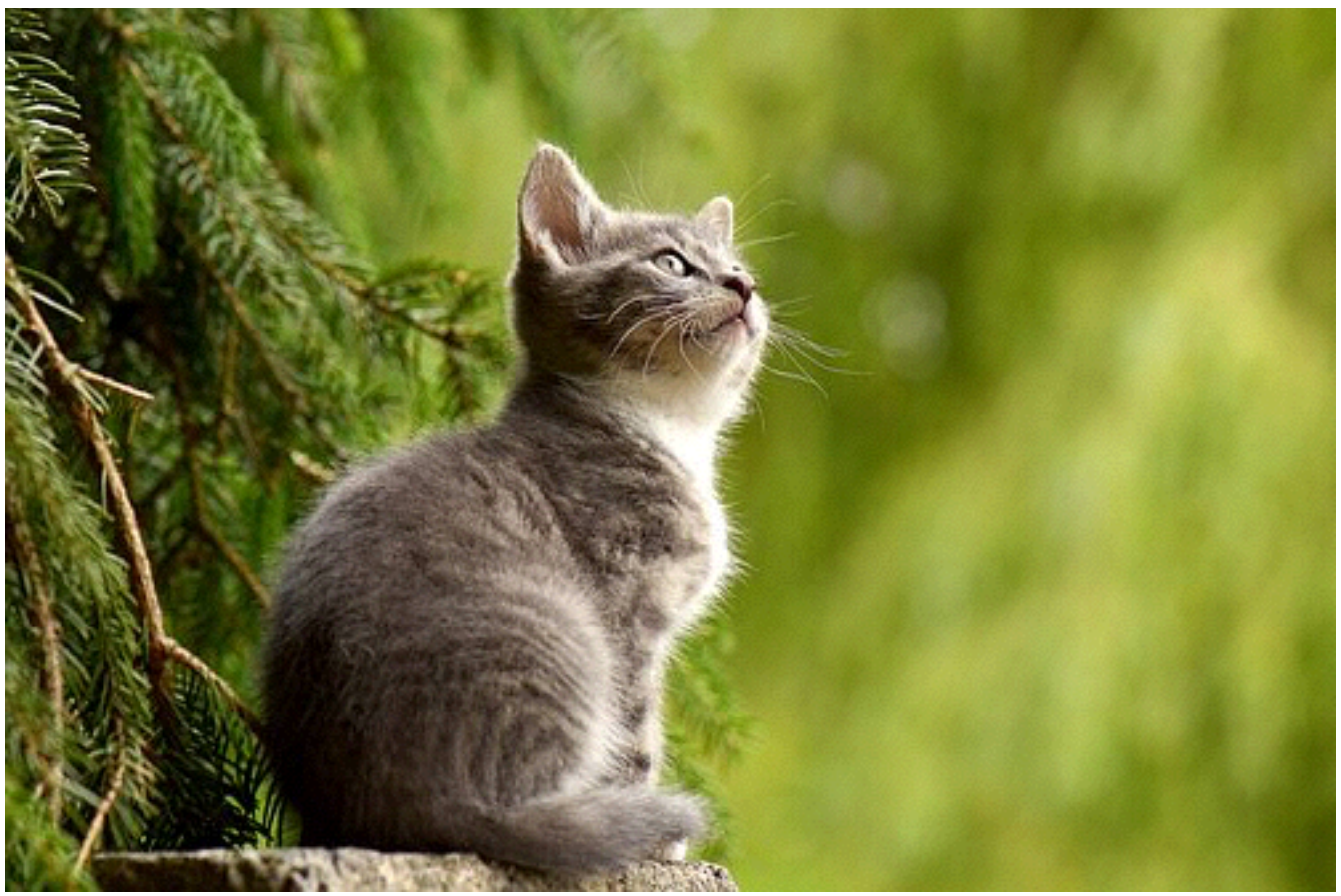
Don't use simulation!
(not always possible!)



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

What is the problem?

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



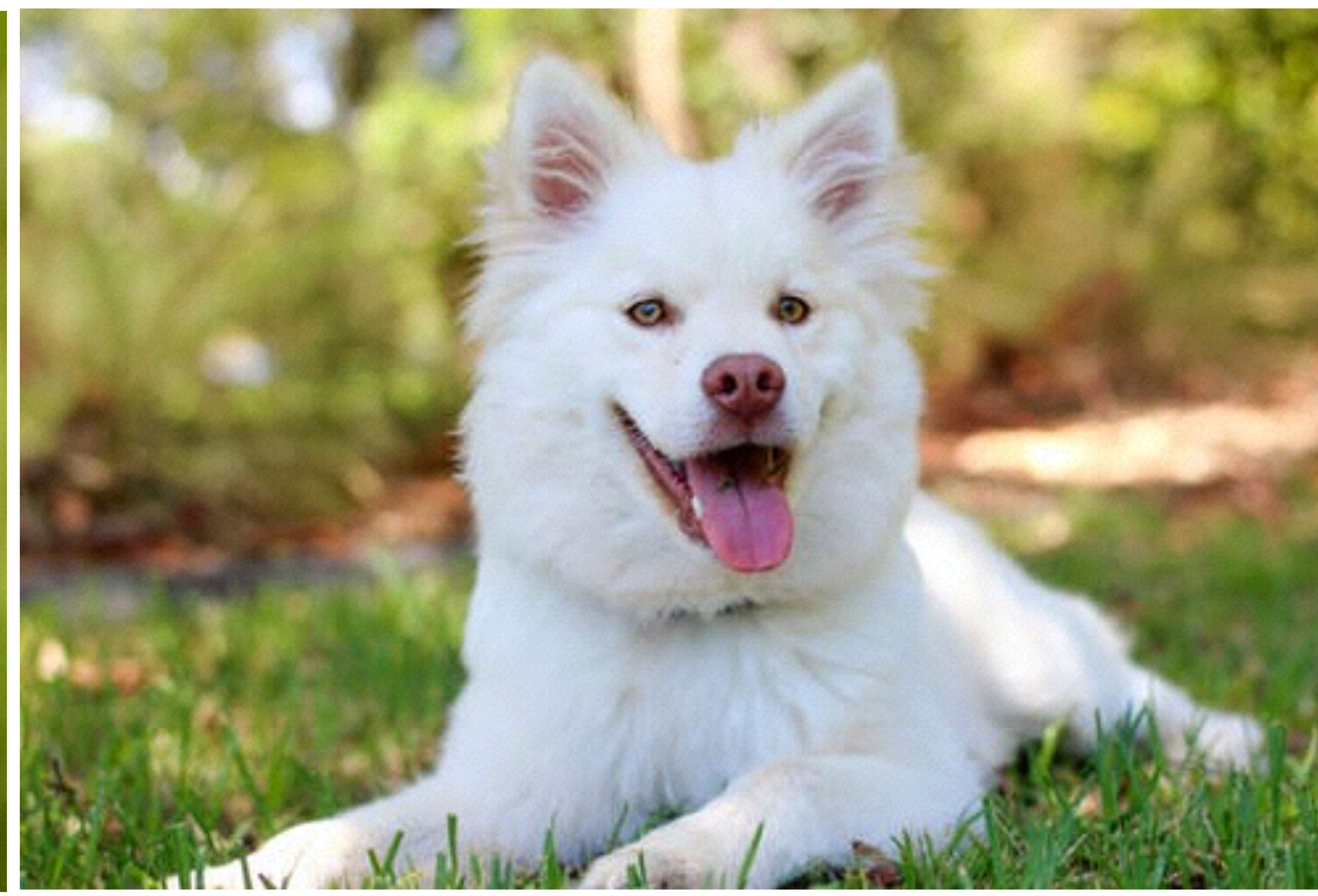
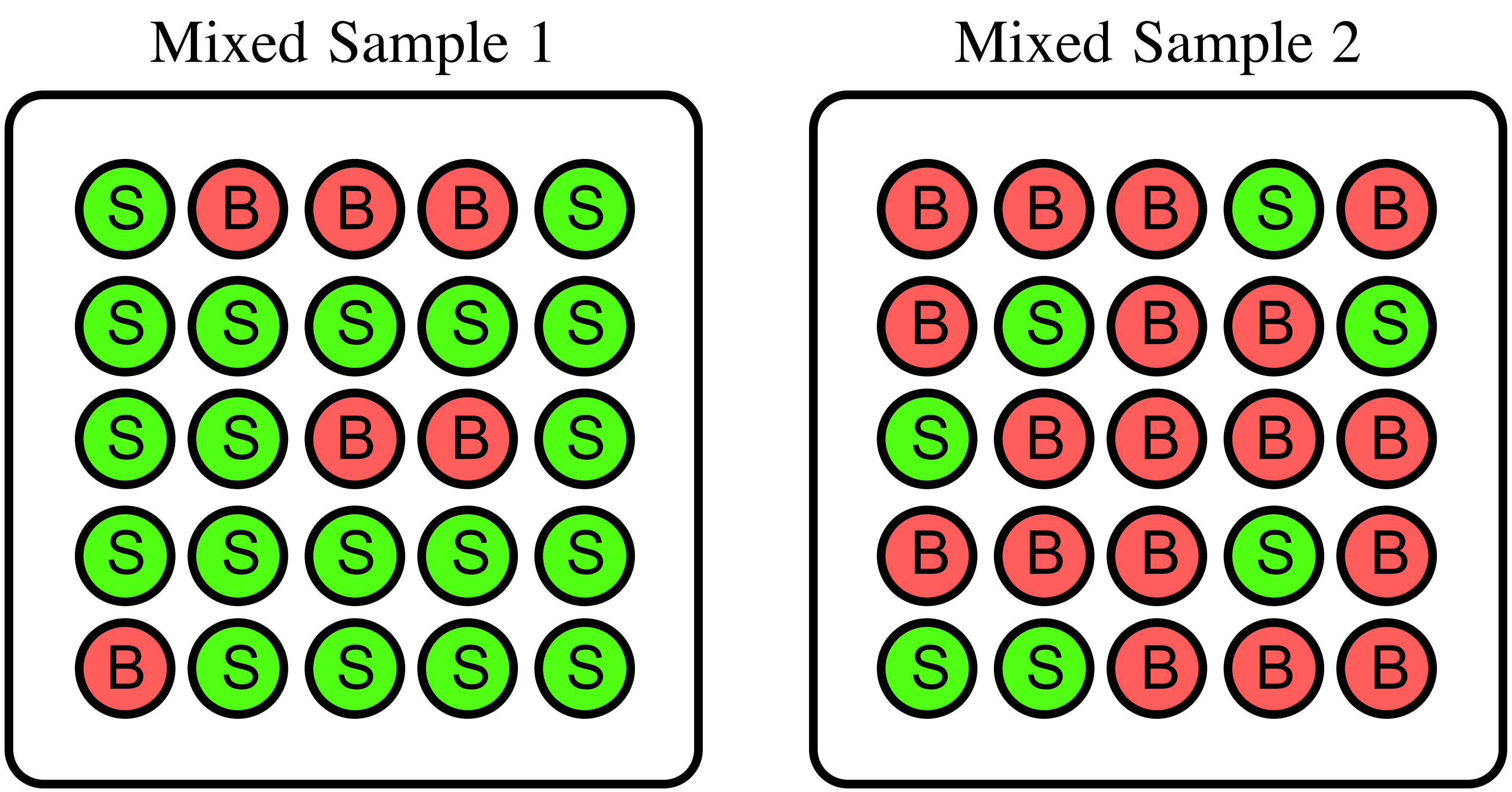


Image credit: pixabay.com

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

What is the problem?

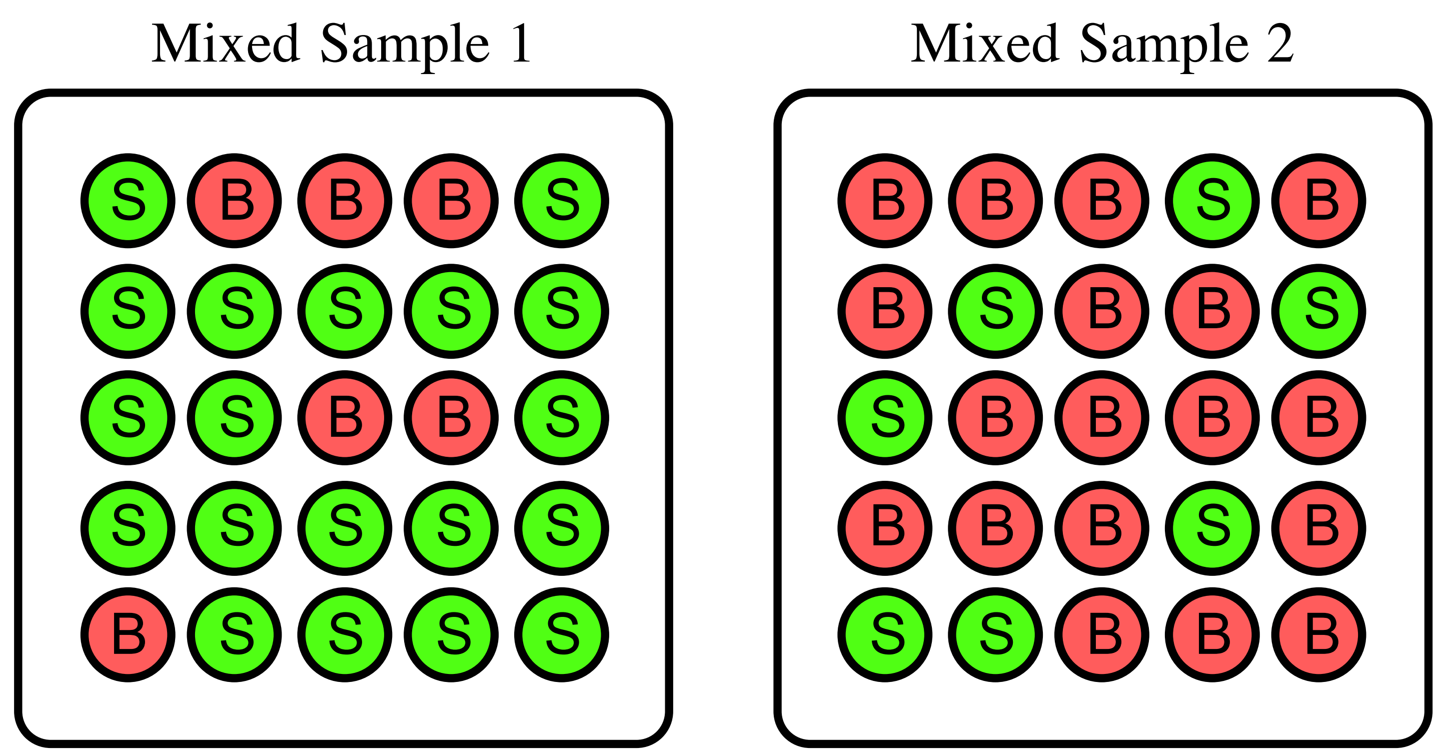
The data are unlabeled and in the best case, come to us as mixtures of two classes ("signal" and "background").



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

Weak supervision: *Classification Without Labels*

Can we learn without any label information?



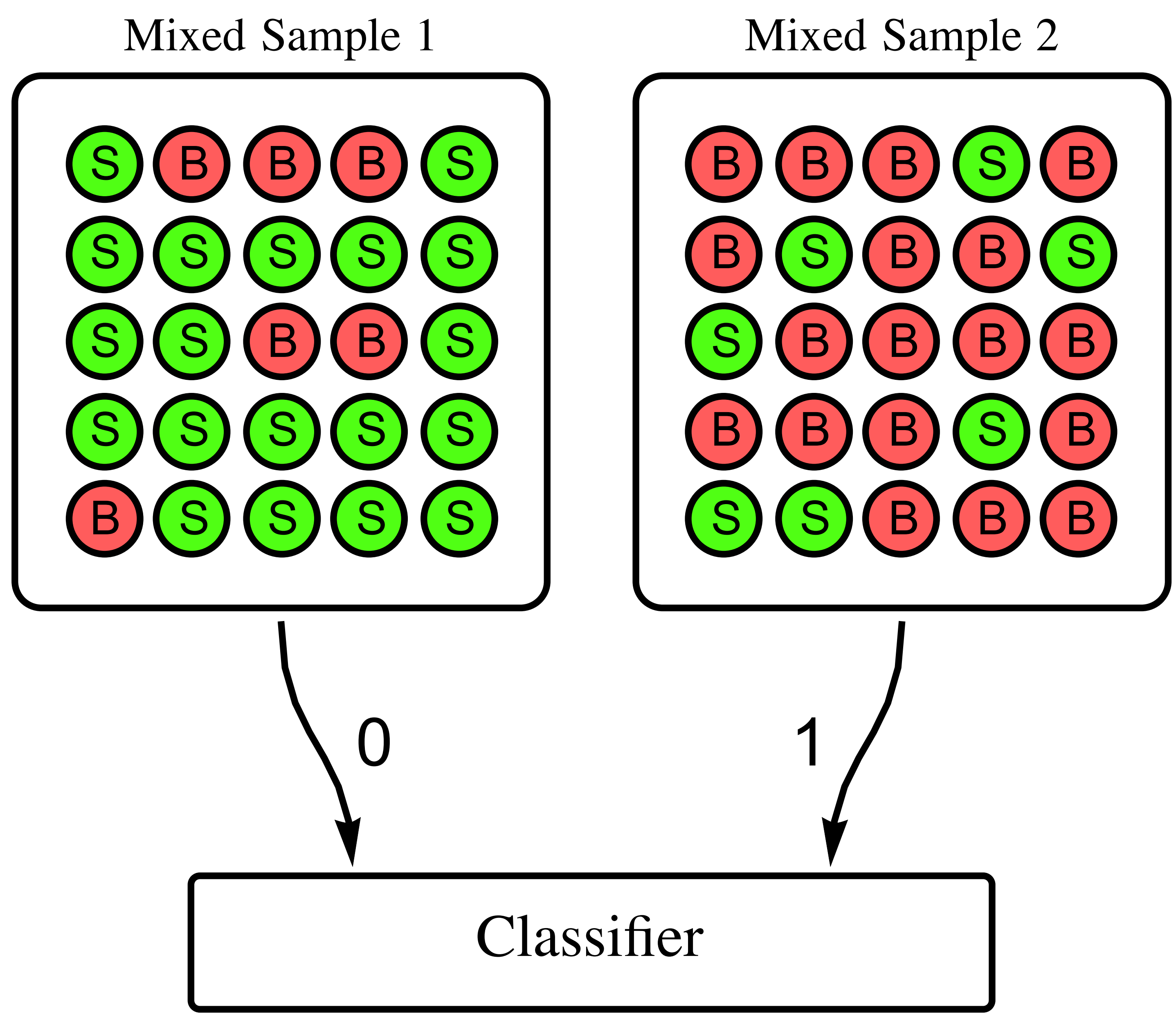
E. Metodiev, **BPN**, J. Thaler, JHEP 10 (2017) 51

Weak supervision: *Classification Without Labels*

Can we learn without any label information?

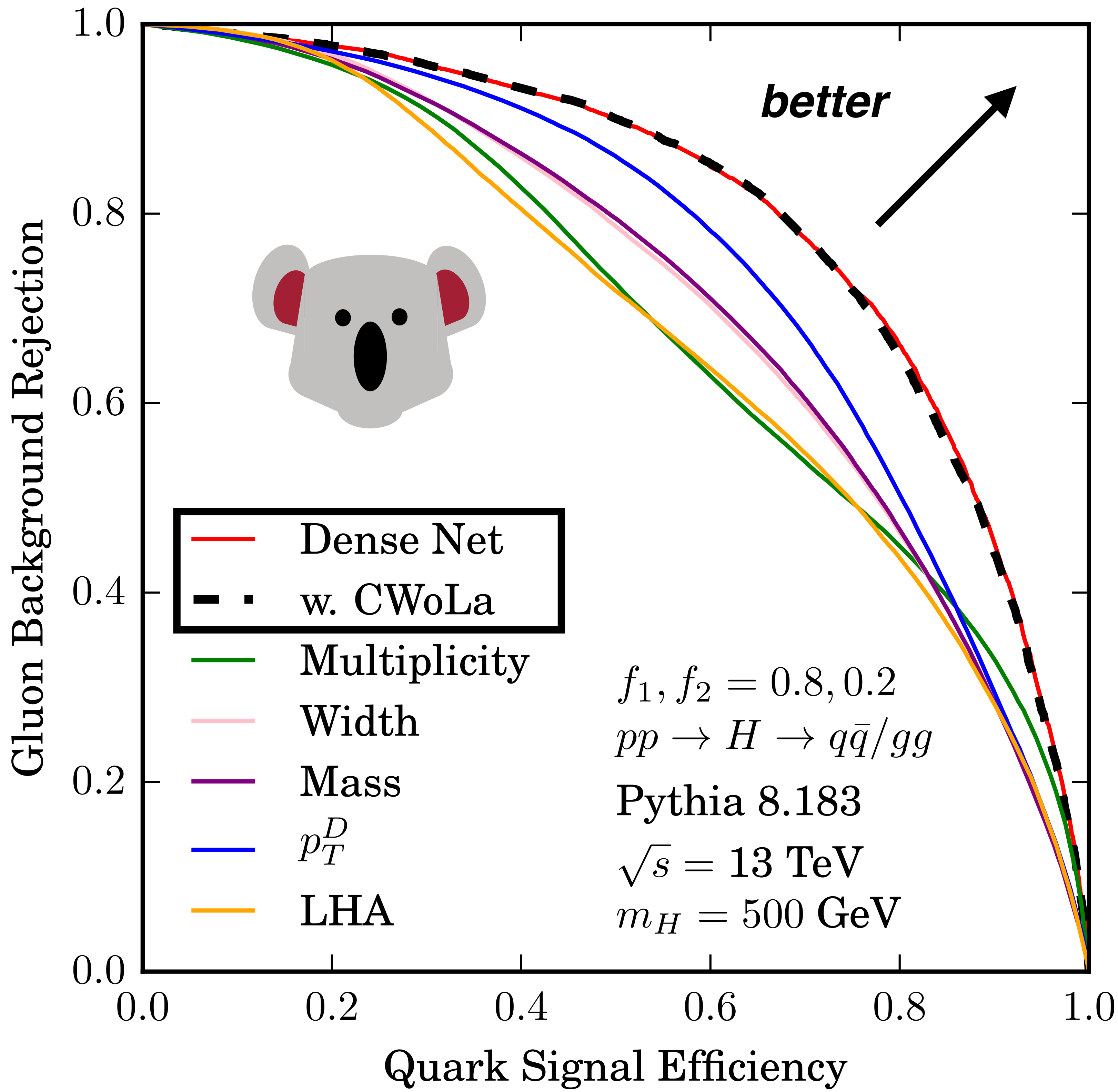
Yes !

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



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

Works!



see also L. Dery, **BPN**, F. Rubbo, A. Schwartzman, JHEP 05 (2017) 145

P. Komiske, E. Metodiev, **BPN**, M. Schwartz, PRD 98 (2018) 011502(R).

CMS Cohen paper

Anomaly detection + weak supervision

How can we use CWoLa to **hunt for new** particles?

*Image from *The Courier Mail*. Koala is actually being freed - I do not condone violence against these animals!

CWoLa Hunting

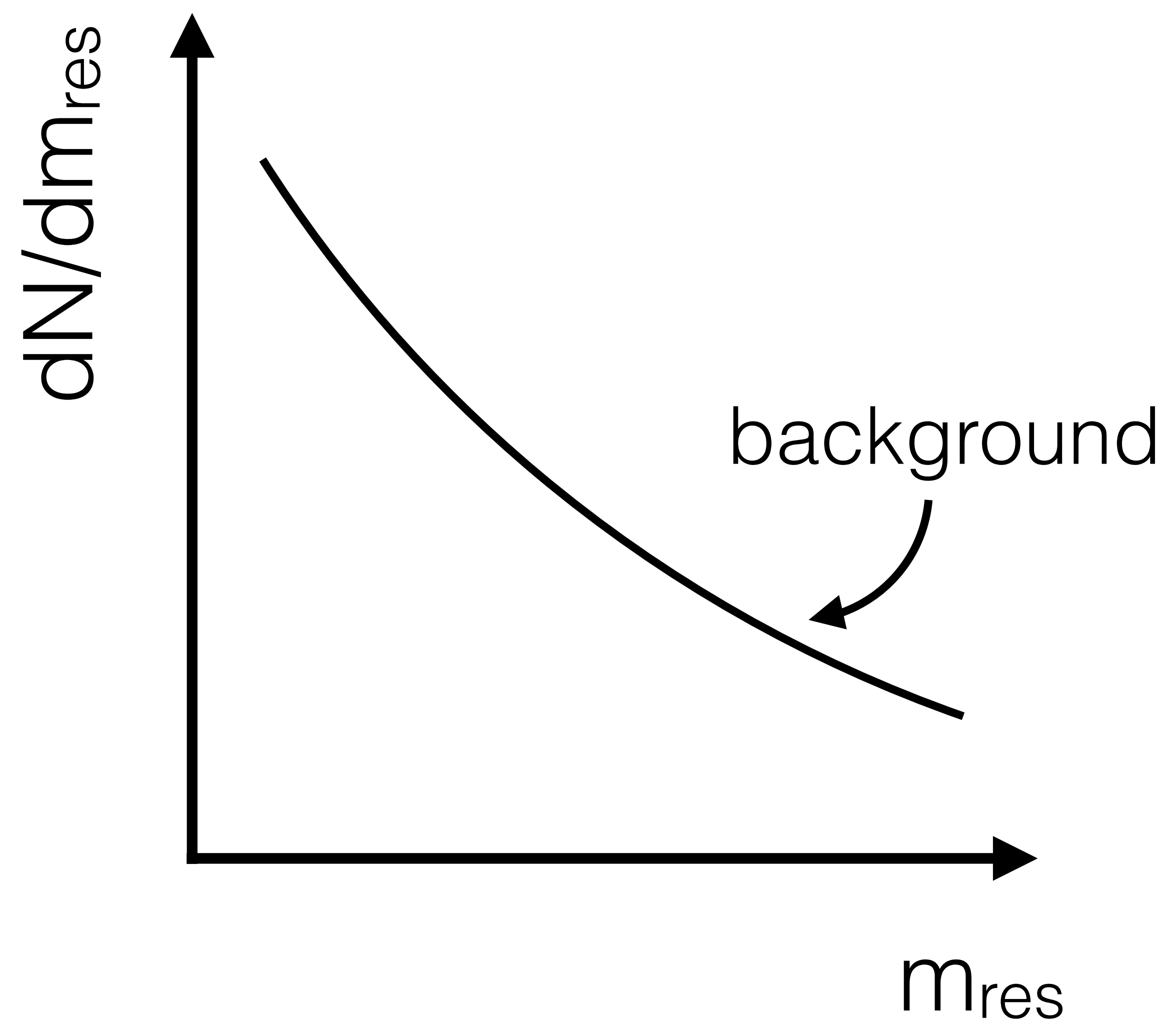
How can we use CWoLa to **hunt for new** particles?

*Image from *The Courier Mail*. Koala is actually being freed - I do not condone violence against these animals!

CWoLa Hunting

Phys. Rev. Lett. 121 (2018) 241803

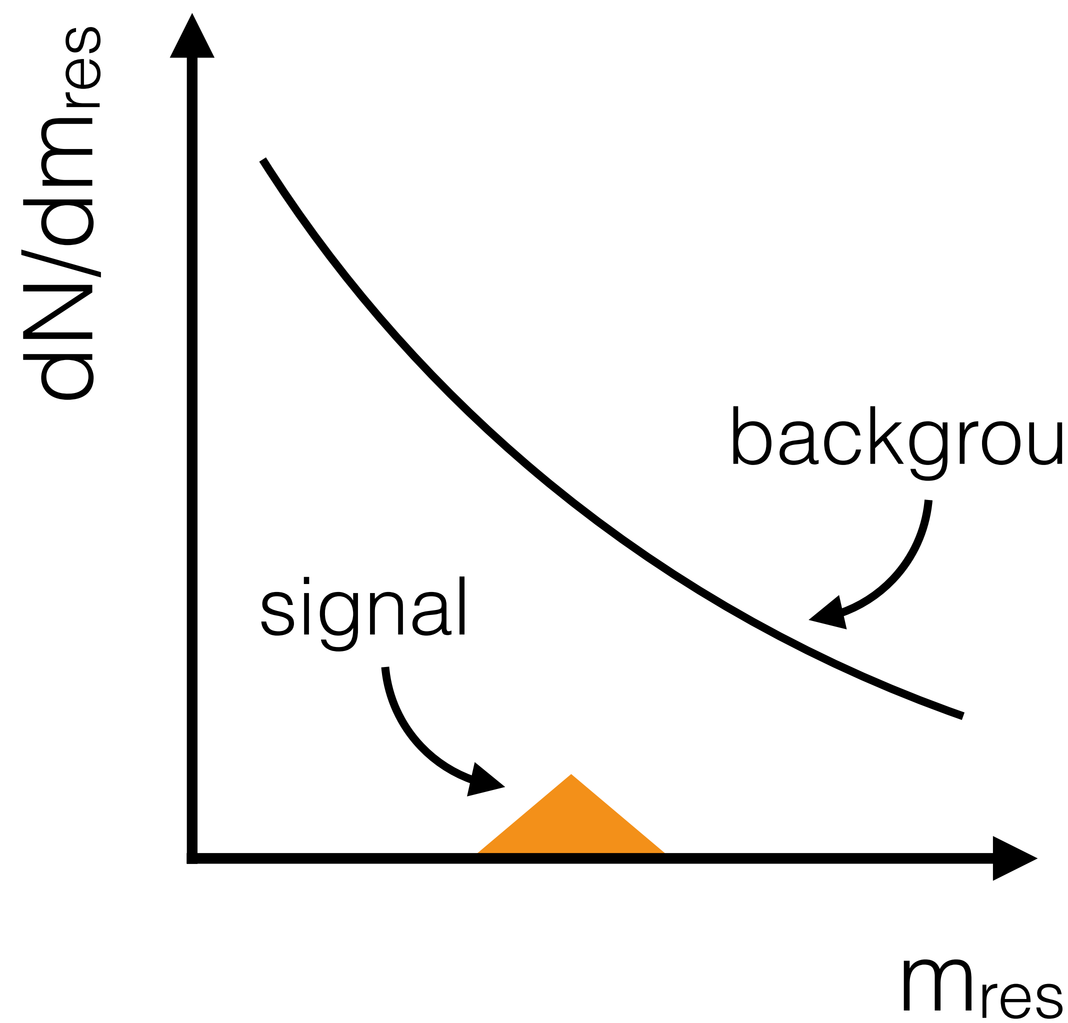
J. Collins, K. Howe, **BPN**



CWoLa Hunting

Phys. Rev. Lett. 121 (2018) 241803

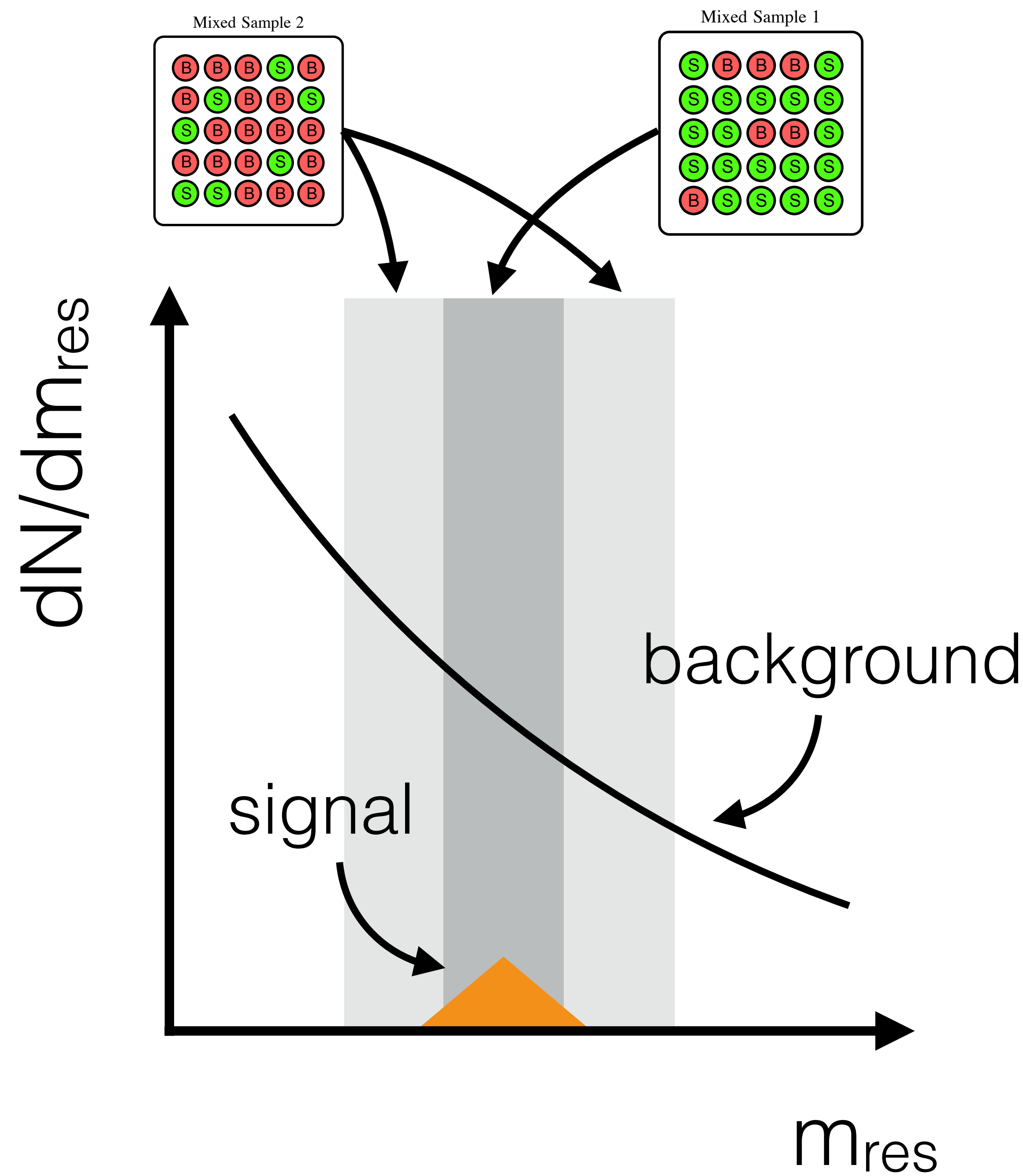
J. Collins, K. Howe, **BPN**



CWoLa Hunting

[Phys. Rev. Lett. 121 (2018) 241803](#)

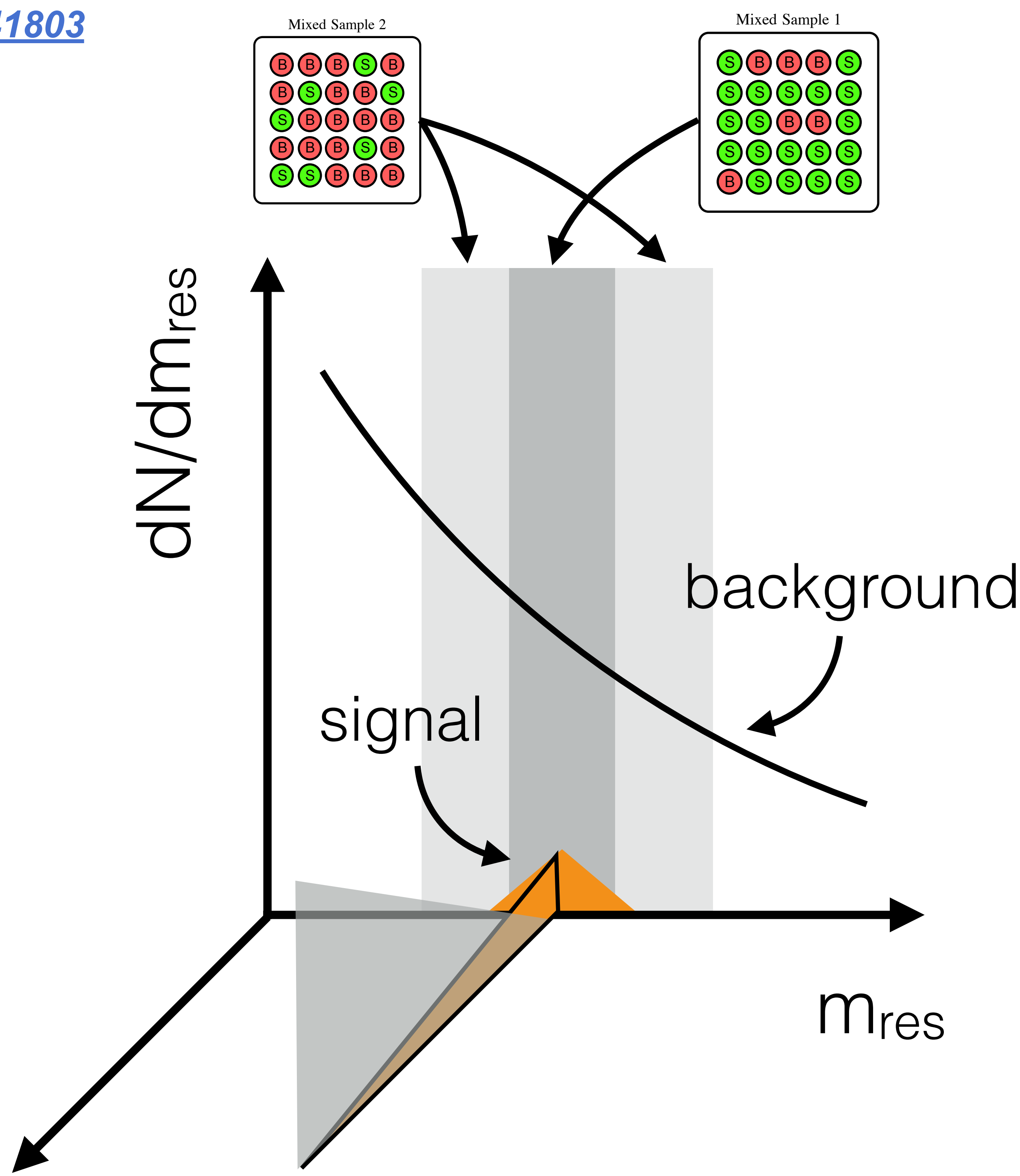
J. Collins, K. Howe, **BPN**



CWoLa Hunting

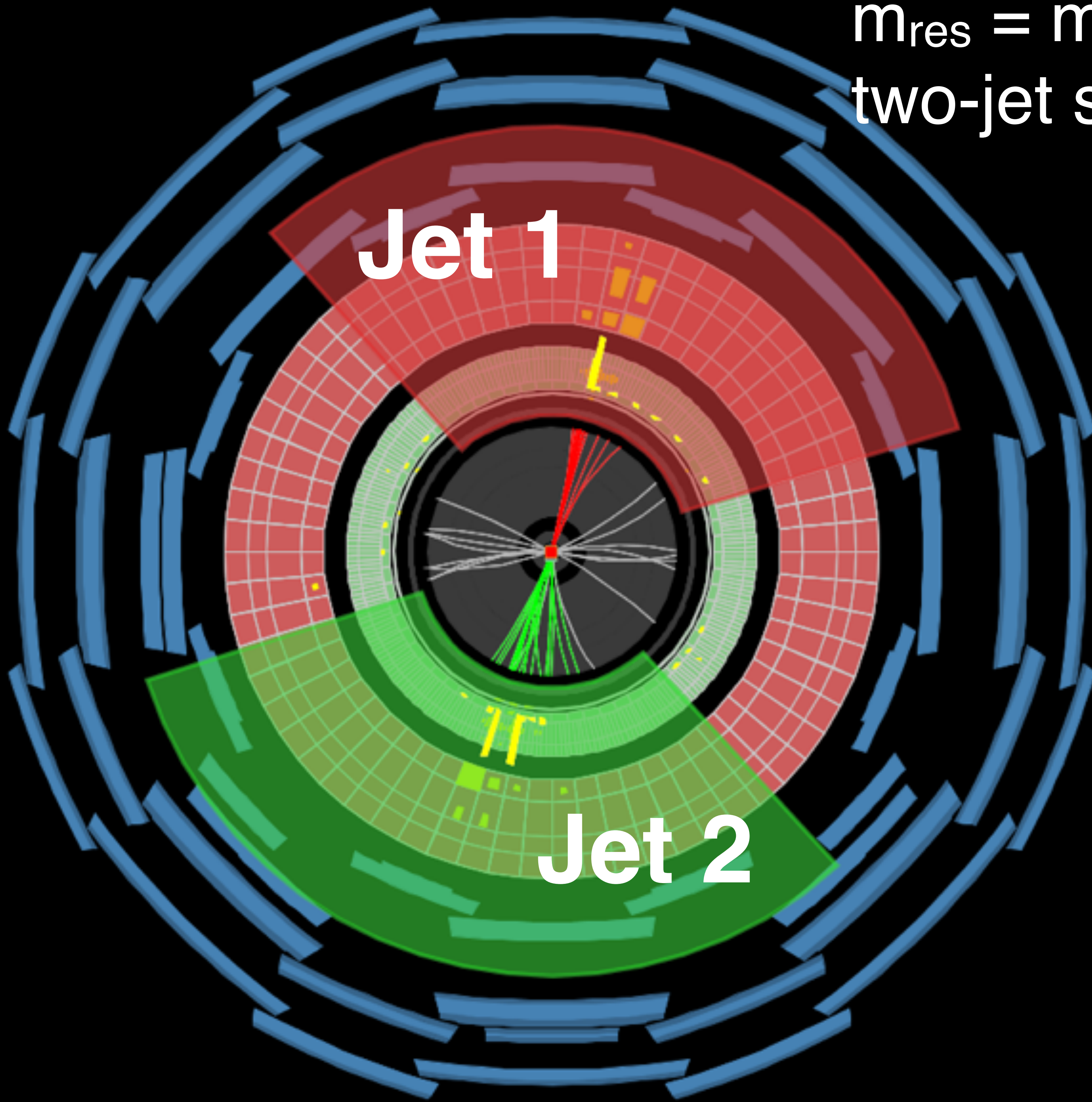
Phys. Rev. Lett. 121 (2018) 241803

J. Collins, K. Howe, **BPN**

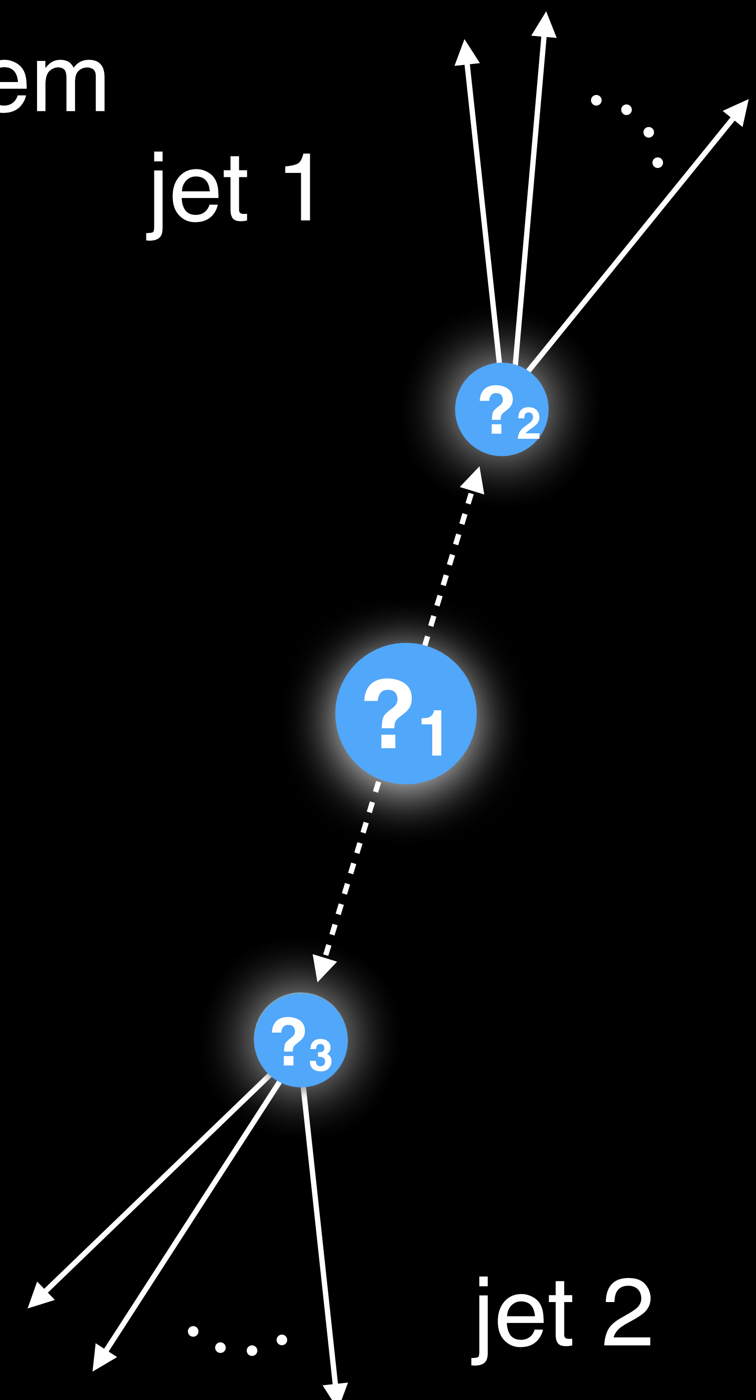


+ be careful to not pay a big trails factor
(ask if interested)

Example: two-jet search



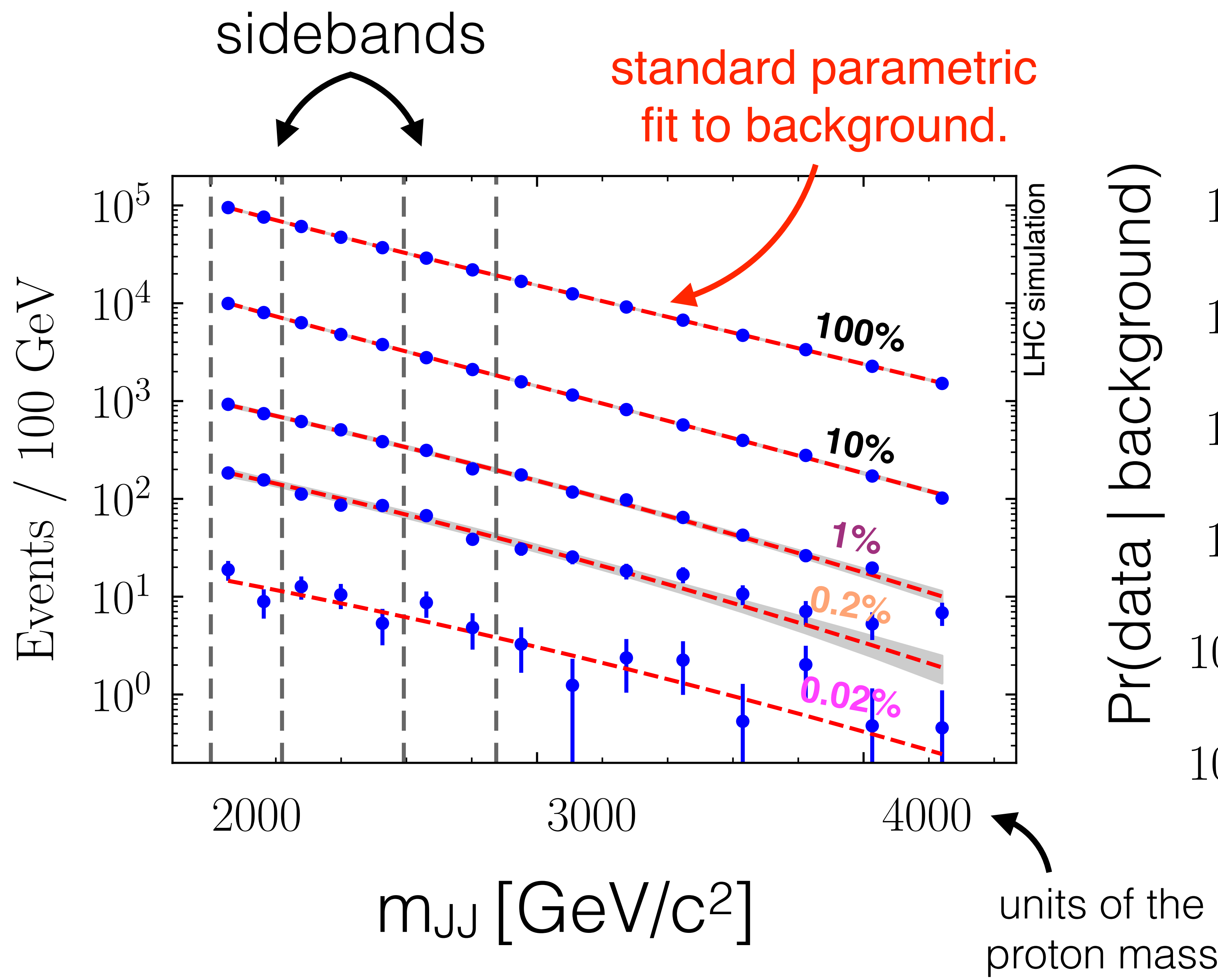
m_{res} = mass of two-jet system

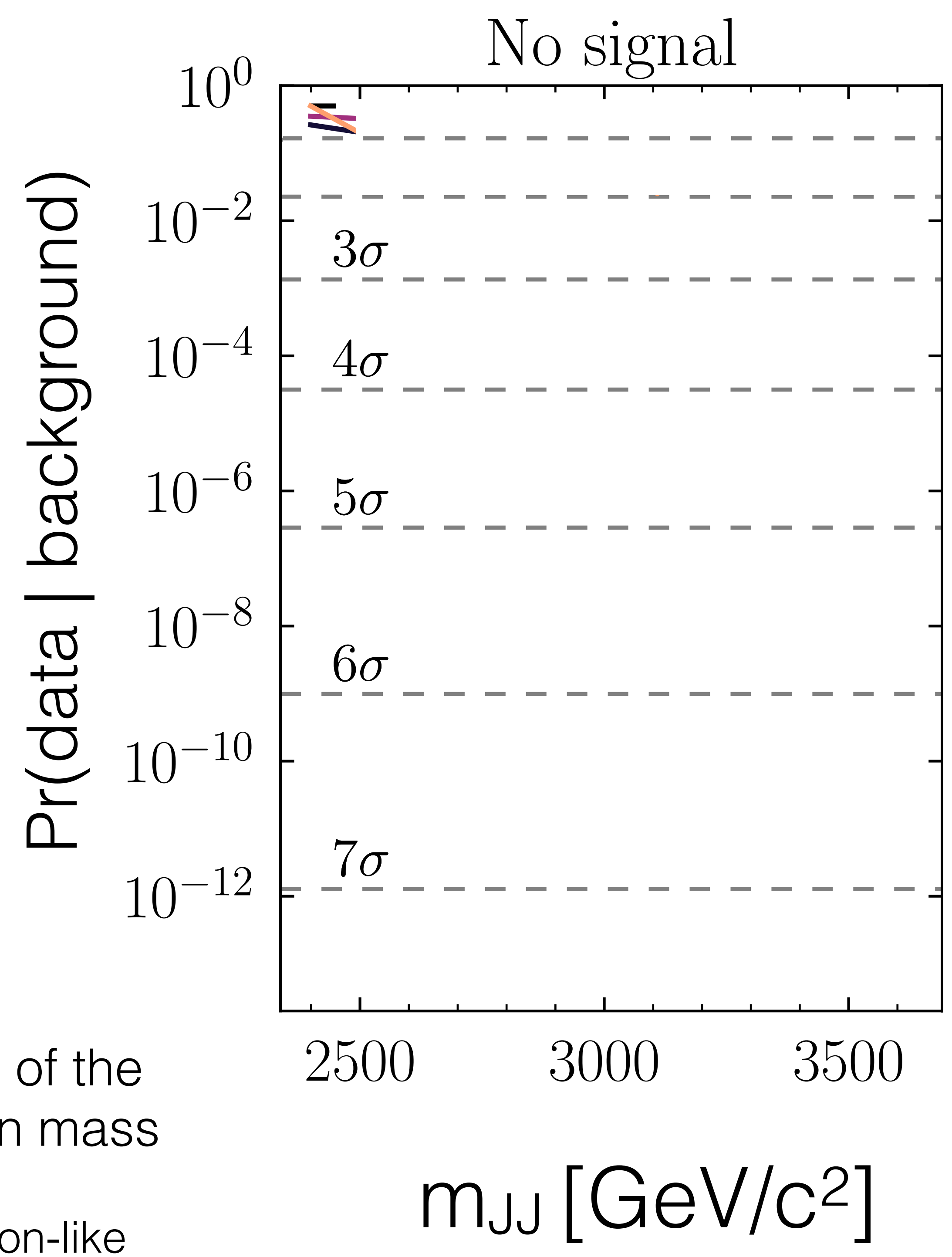


collisions in/out of page

y = many features of the two jets

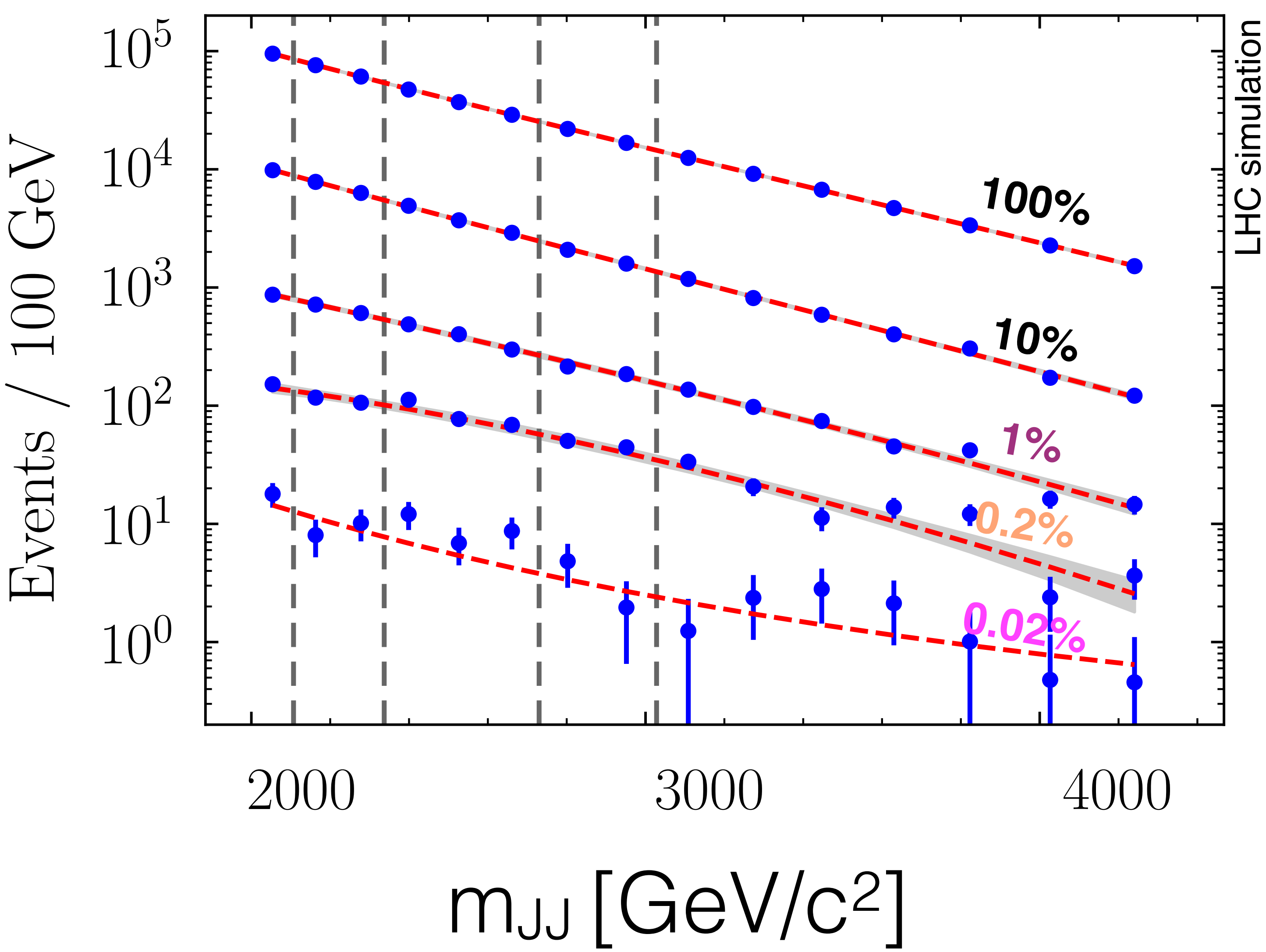
Example: two-jet search

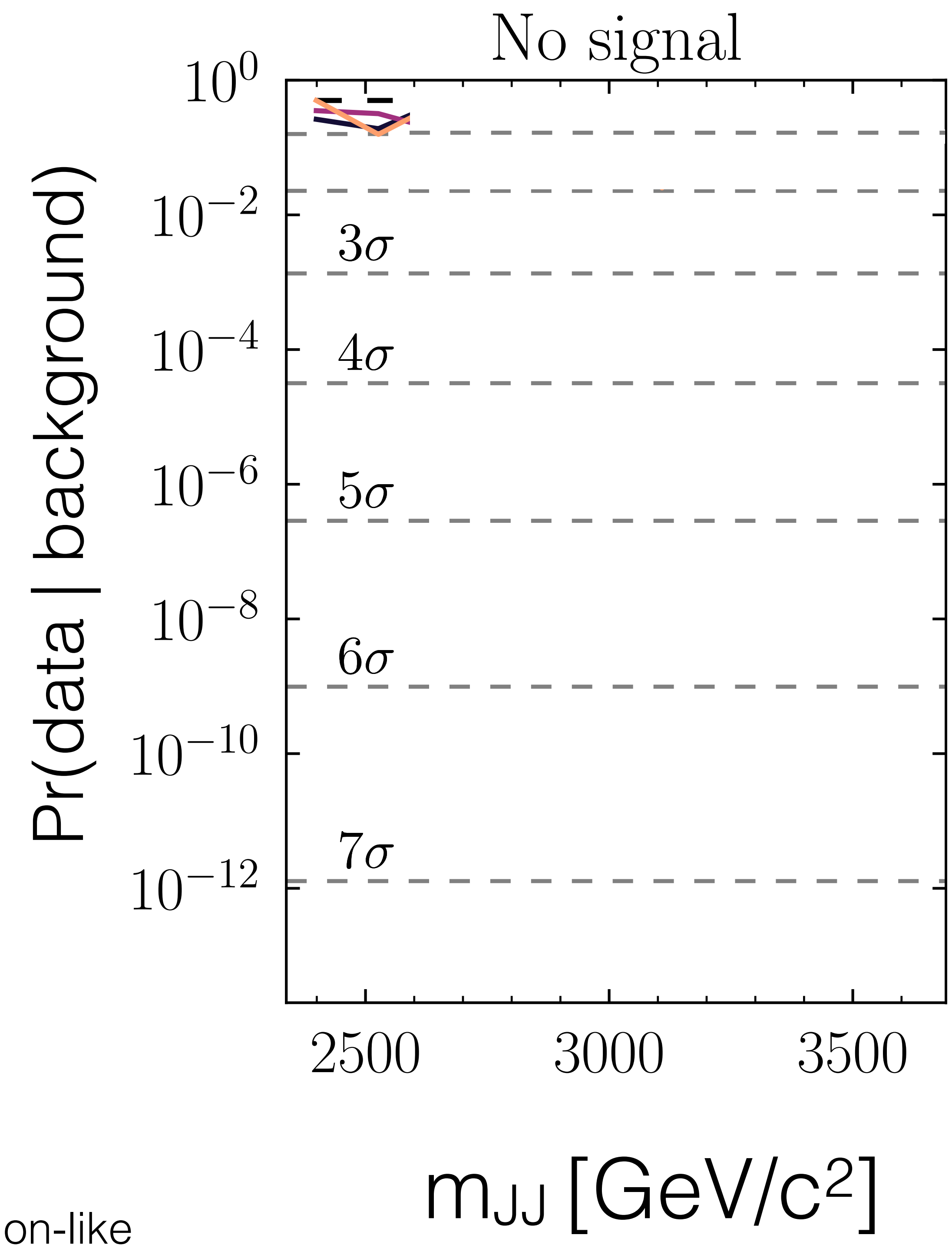




- - - no cut on NN
— most 10% signal-region-like
— most 1% signal-region-like
— most 0.2% signal-region-like

Example: two-jet search





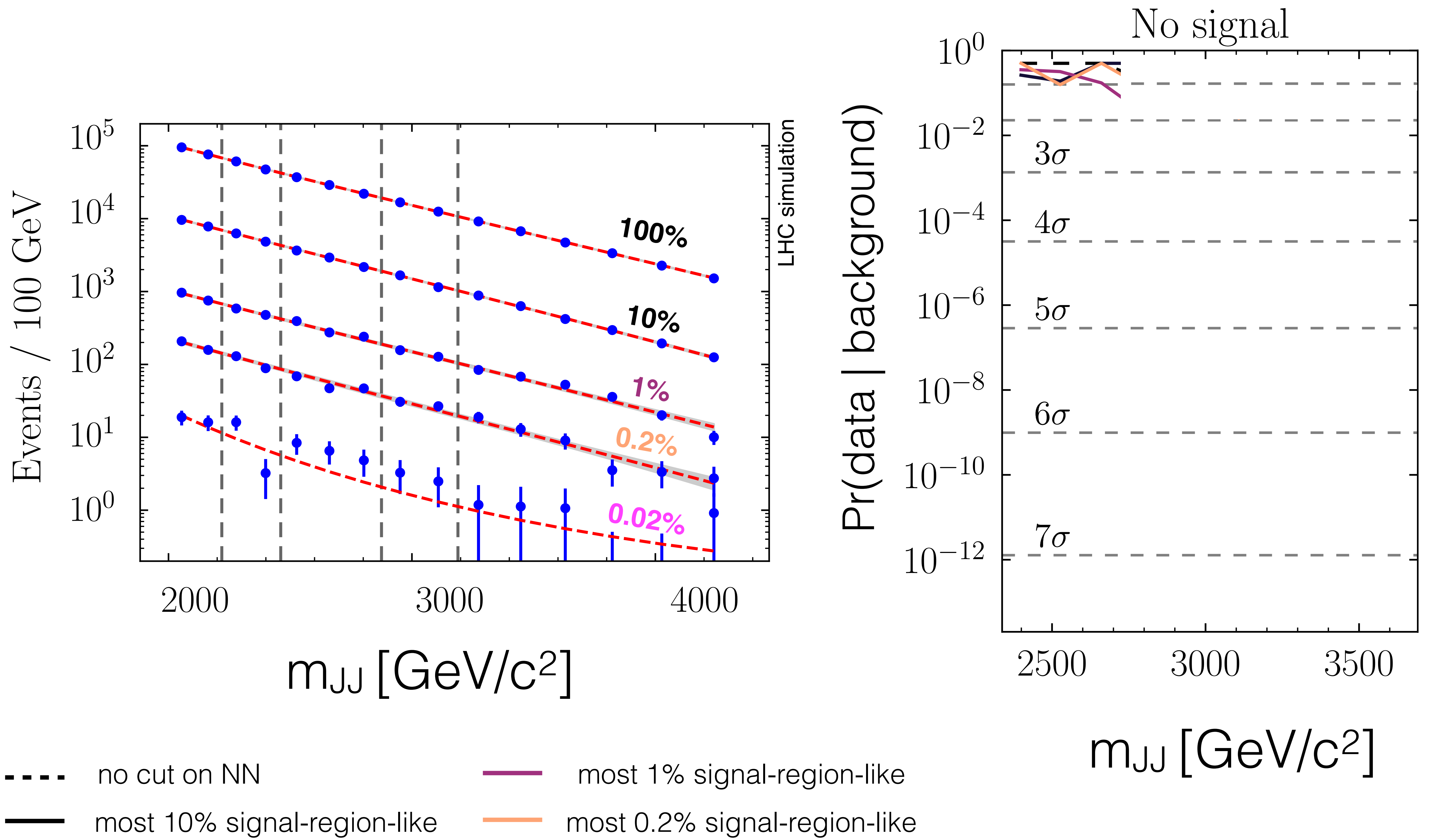
no cut on NN

most 10% signal-region-like

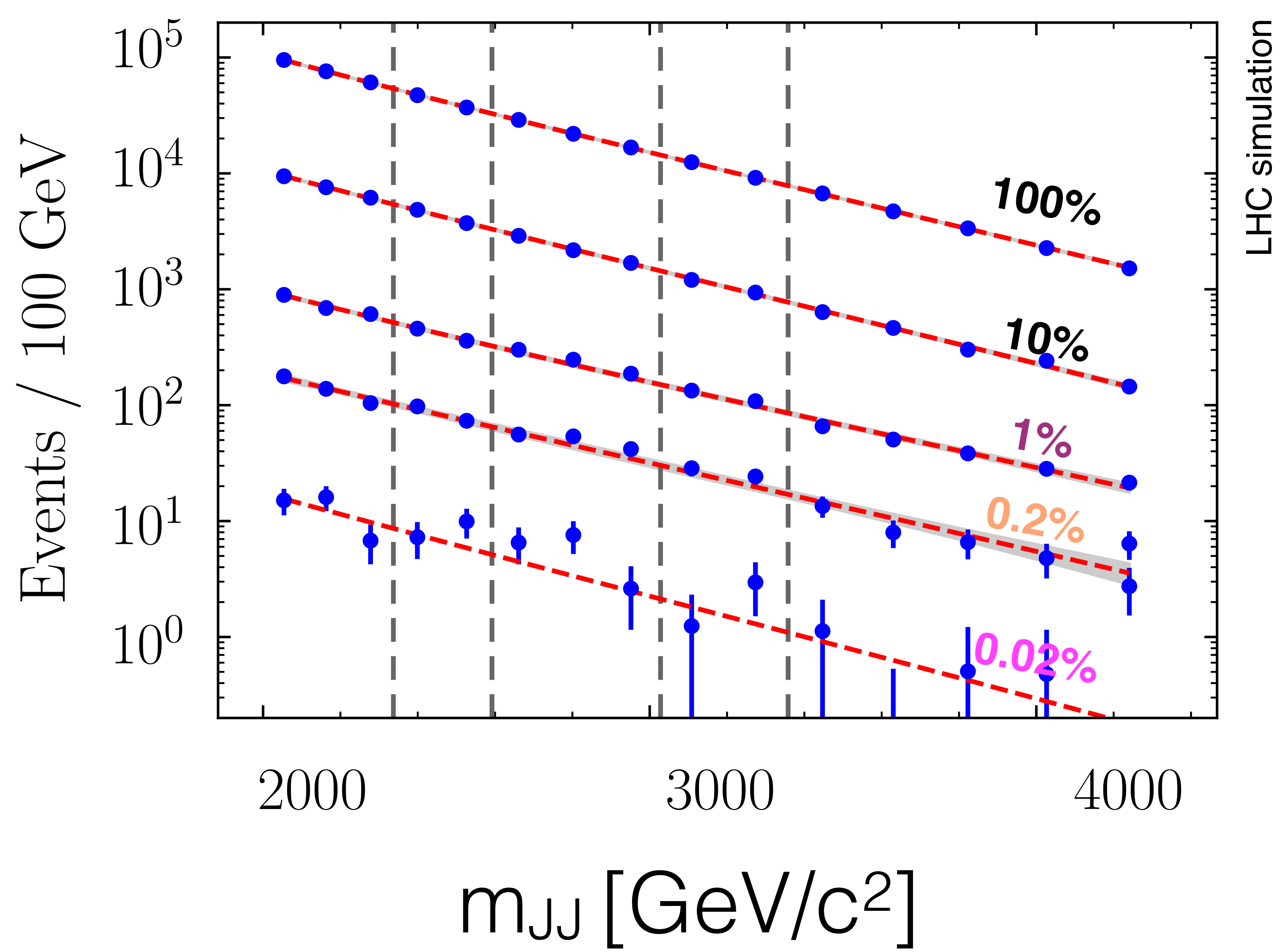
most 1% signal-region-like

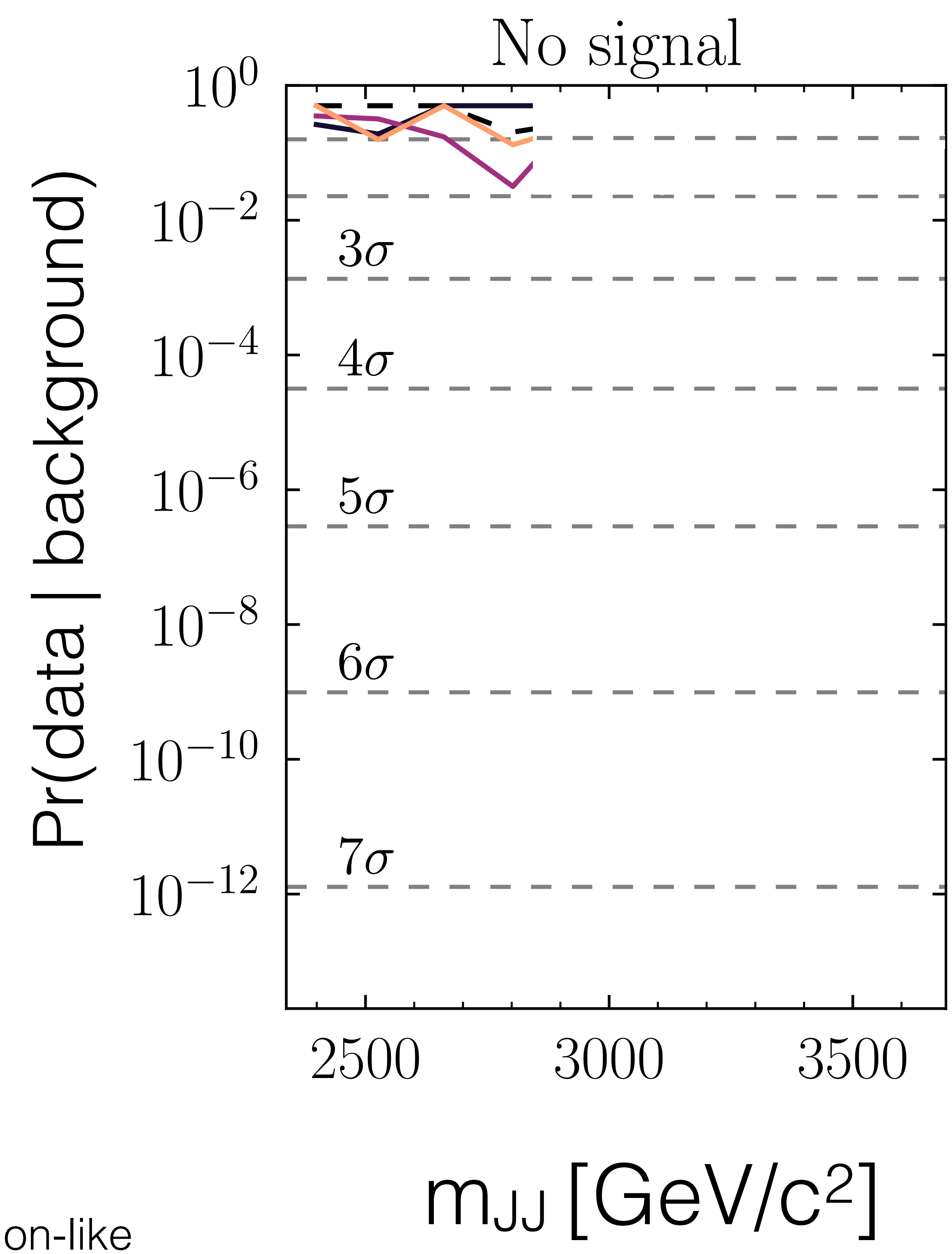
most 0.2% signal-region-like

Example: two-jet search



Example: two-jet search

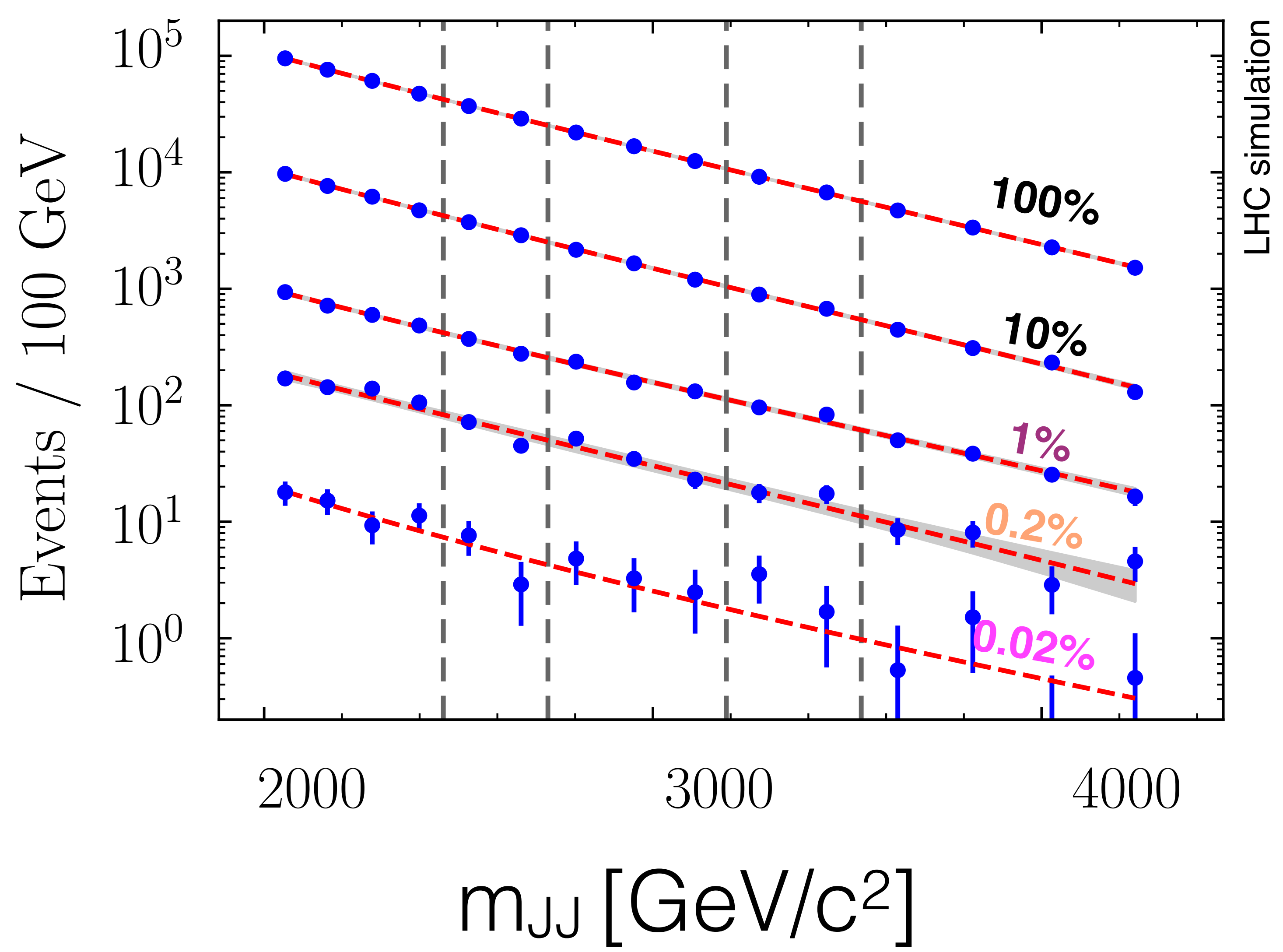


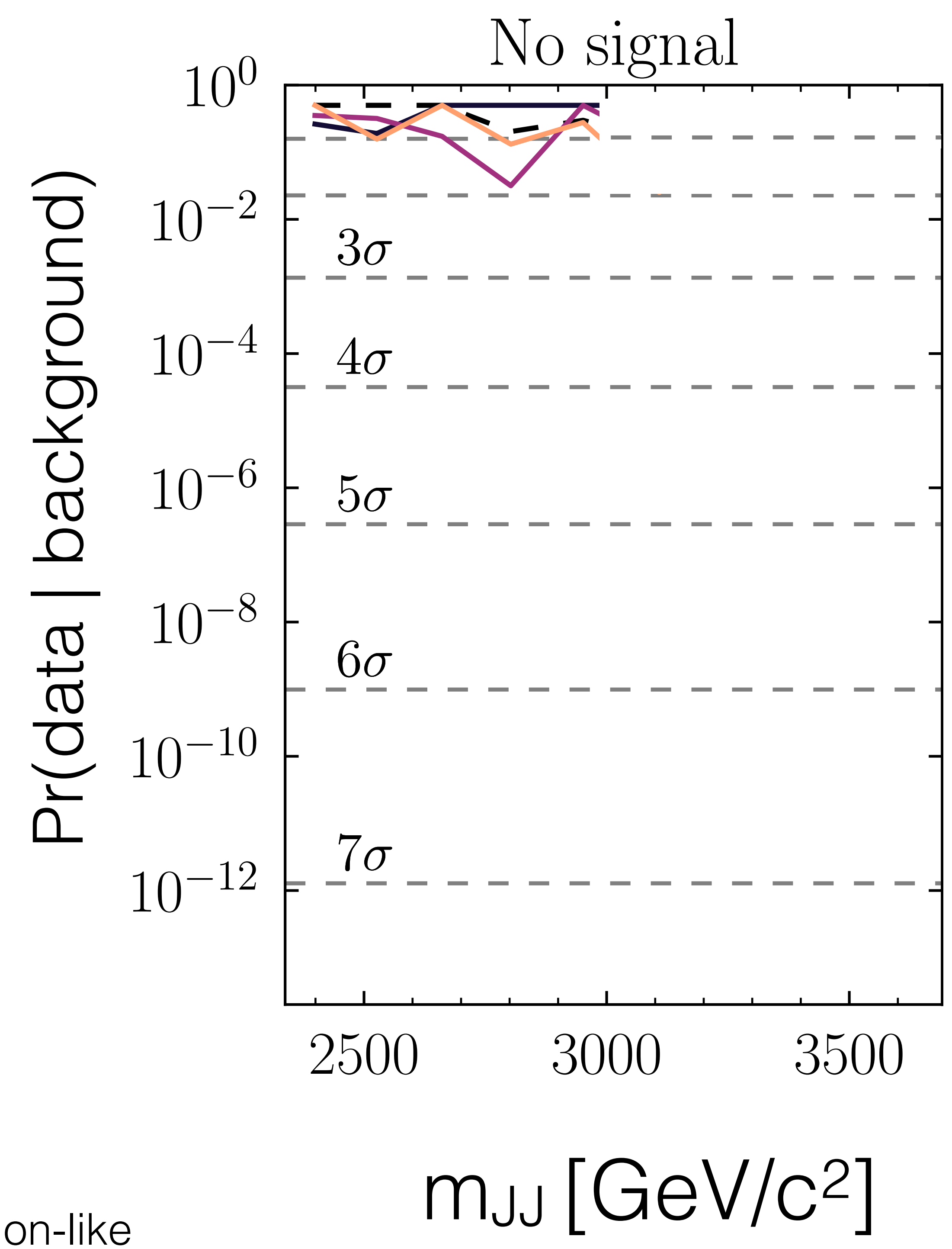


no cut on NN — most 1% signal-region-like

most 10% signal-region-like — most 0.2% signal-region-like

Example: two-jet search





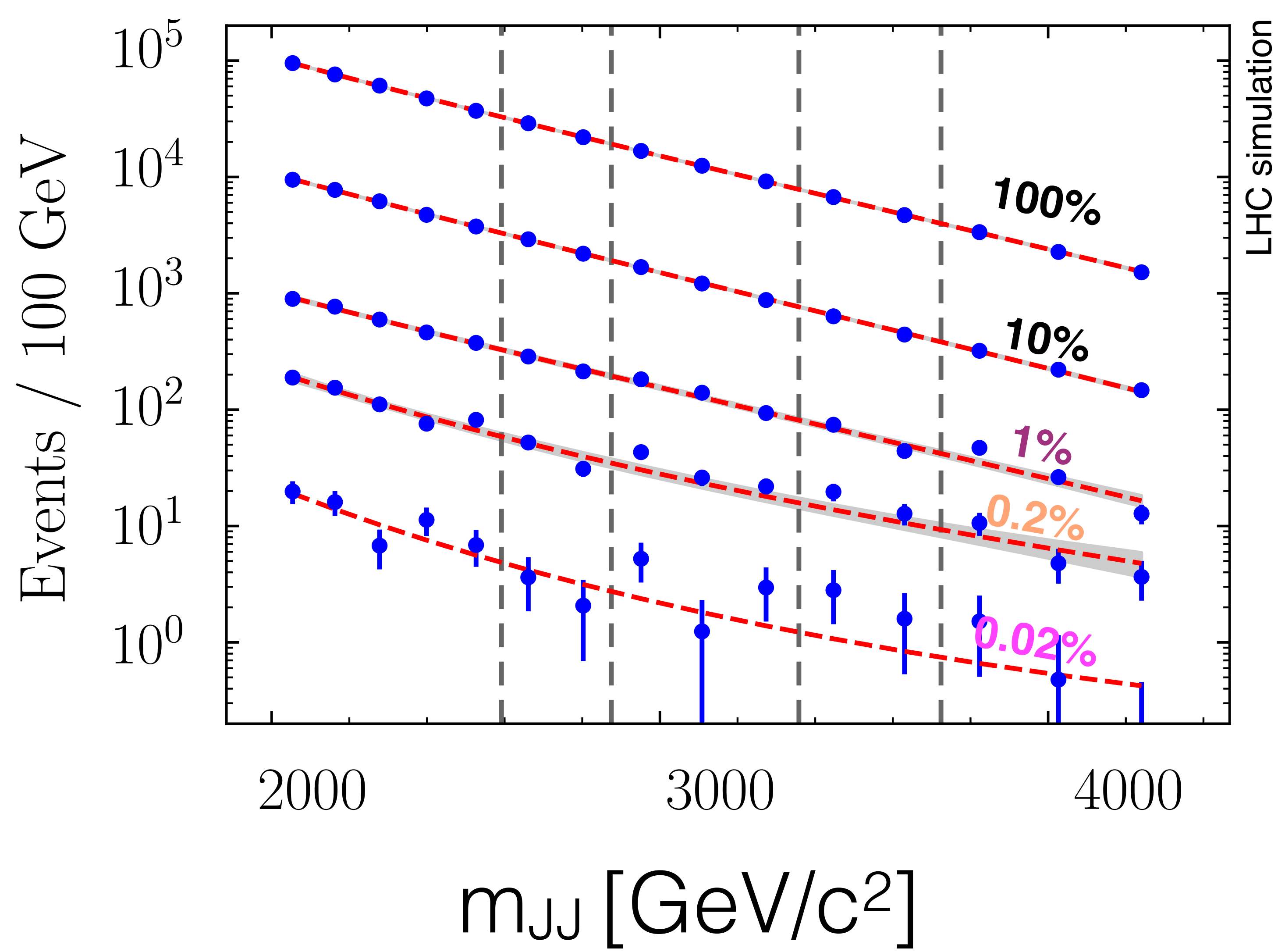
no cut on NN

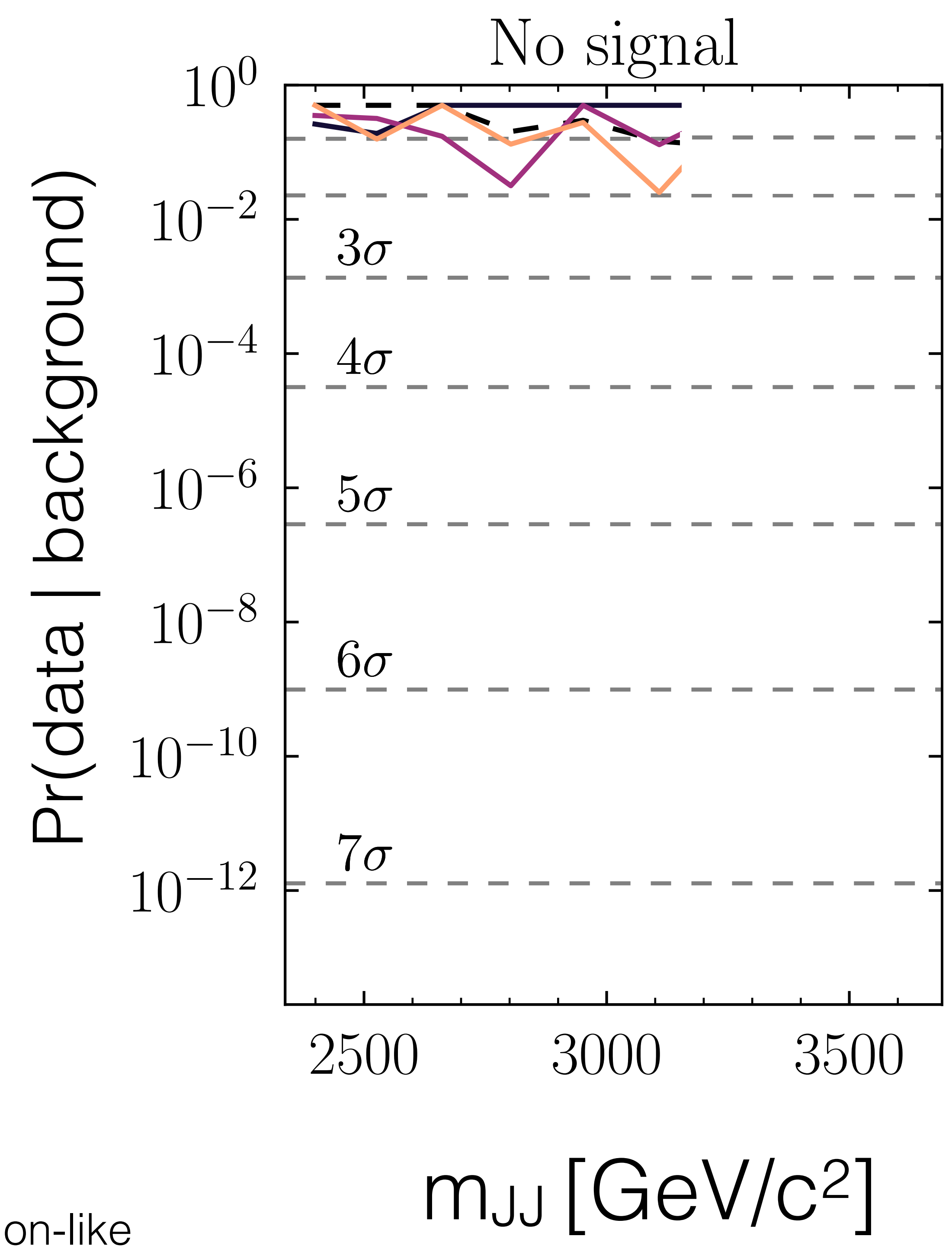
most 10% signal-region-like

most 1% signal-region-like

most 0.2% signal-region-like

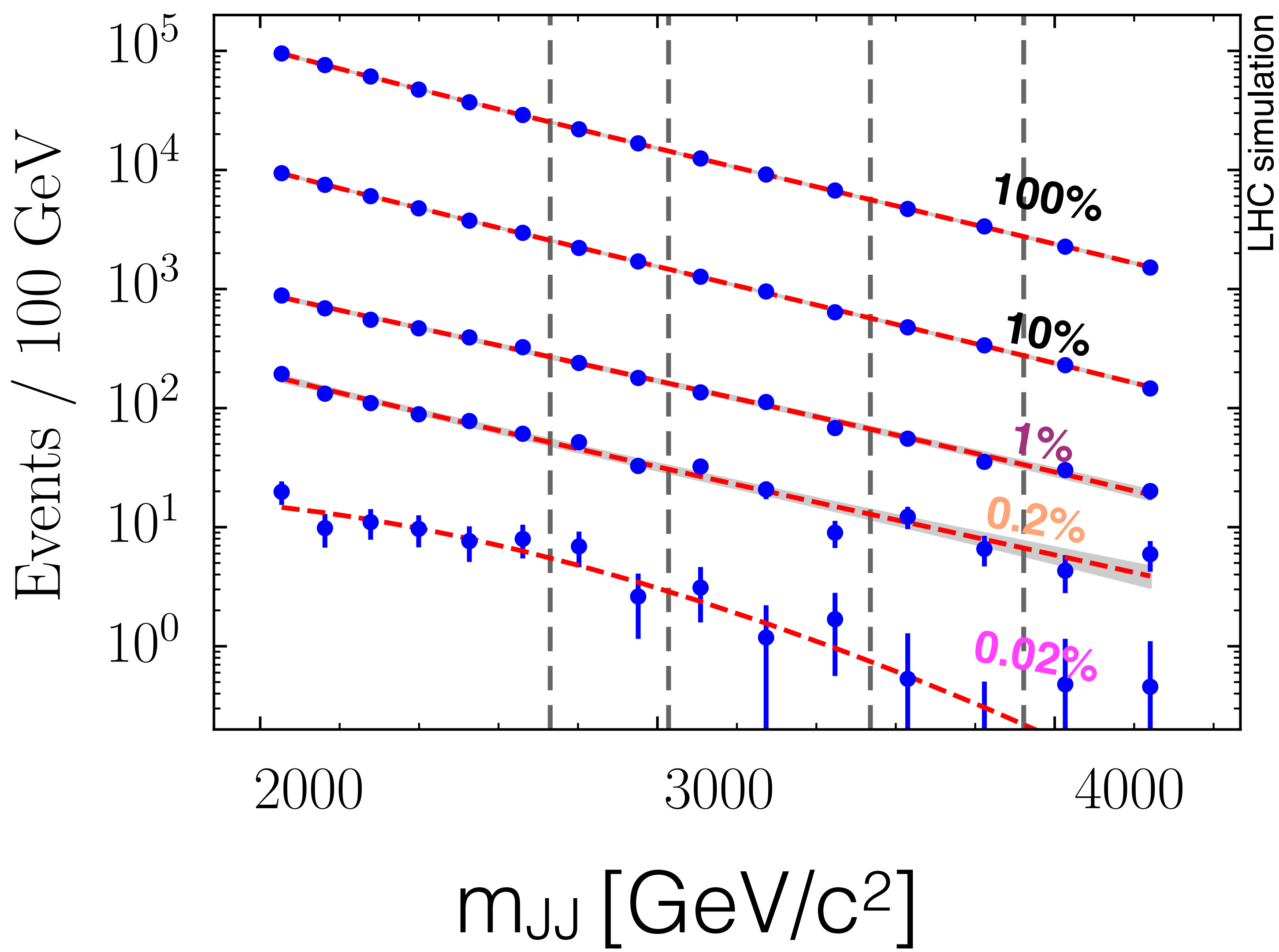
Example: two-jet search

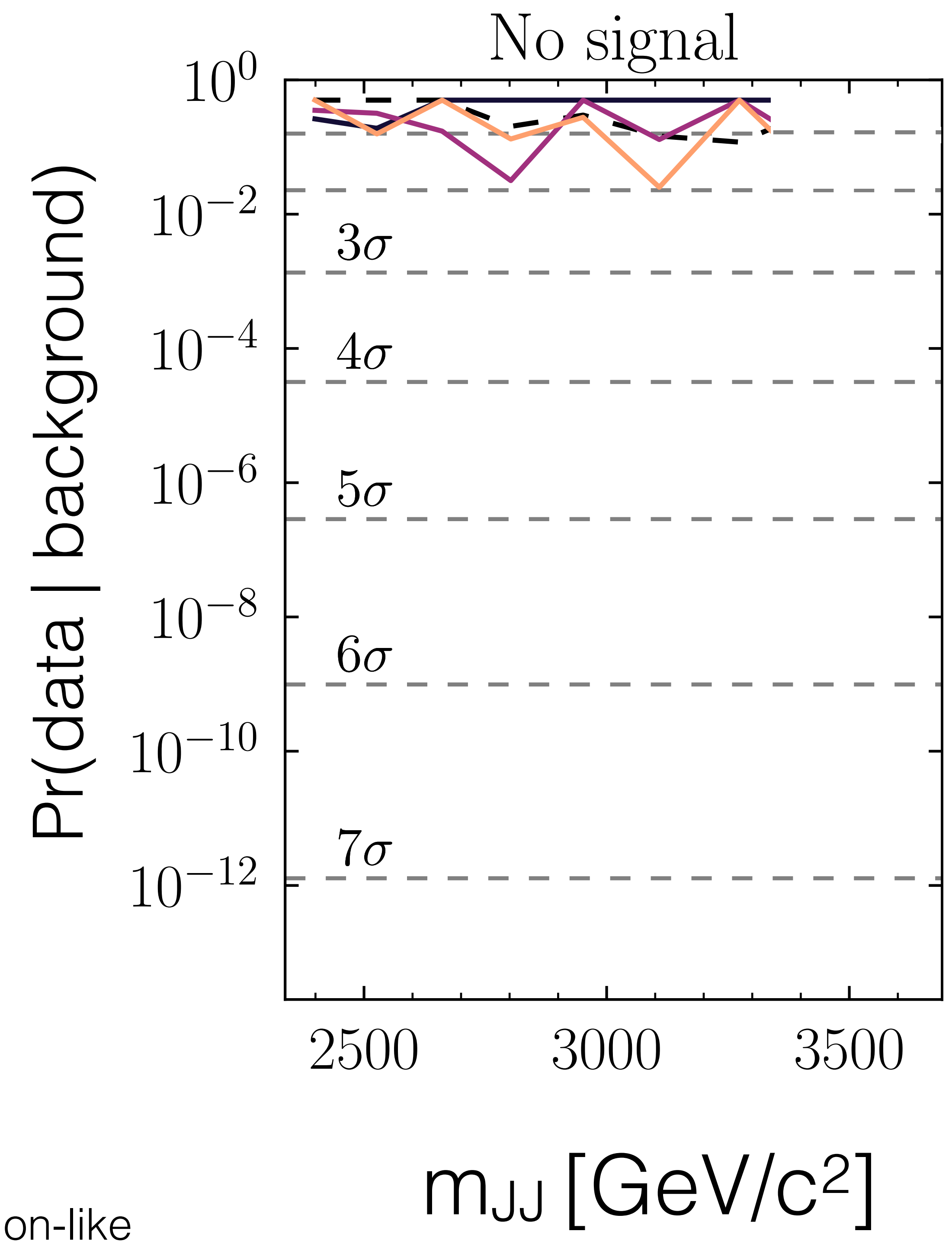




no cut on NN
most 10% signal-region-like
most 1% signal-region-like
most 0.2% signal-region-like

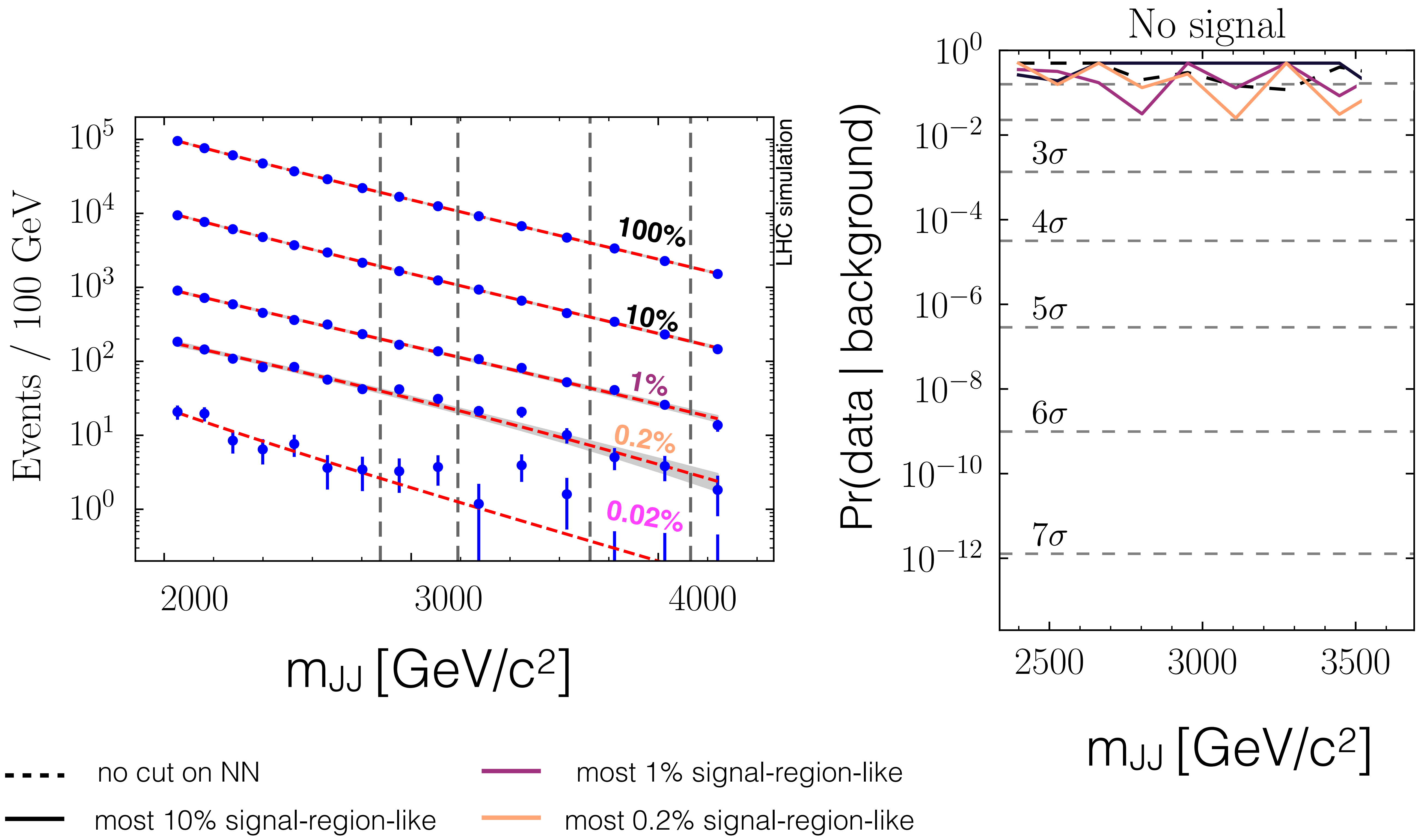
Example: two-jet search



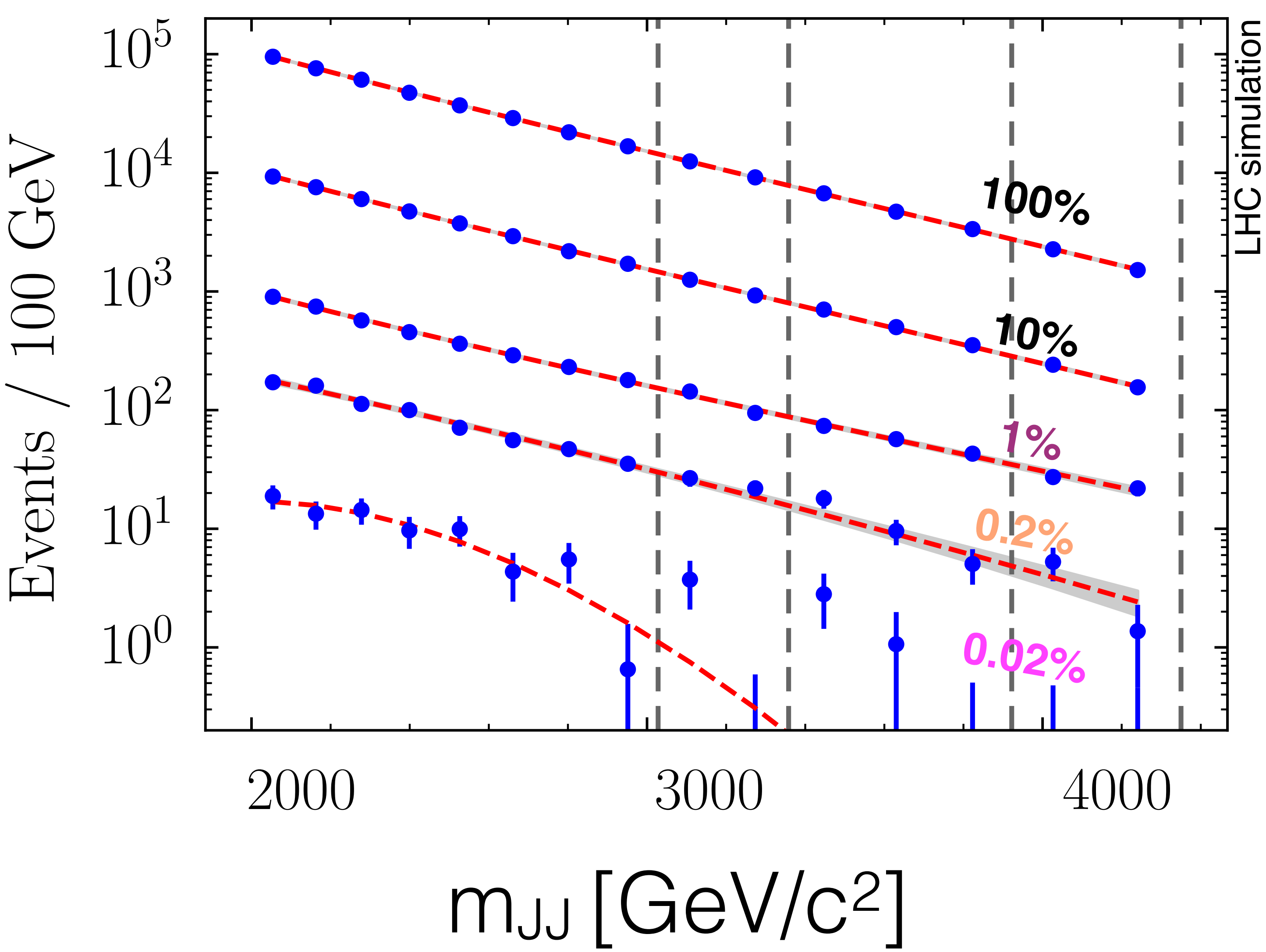


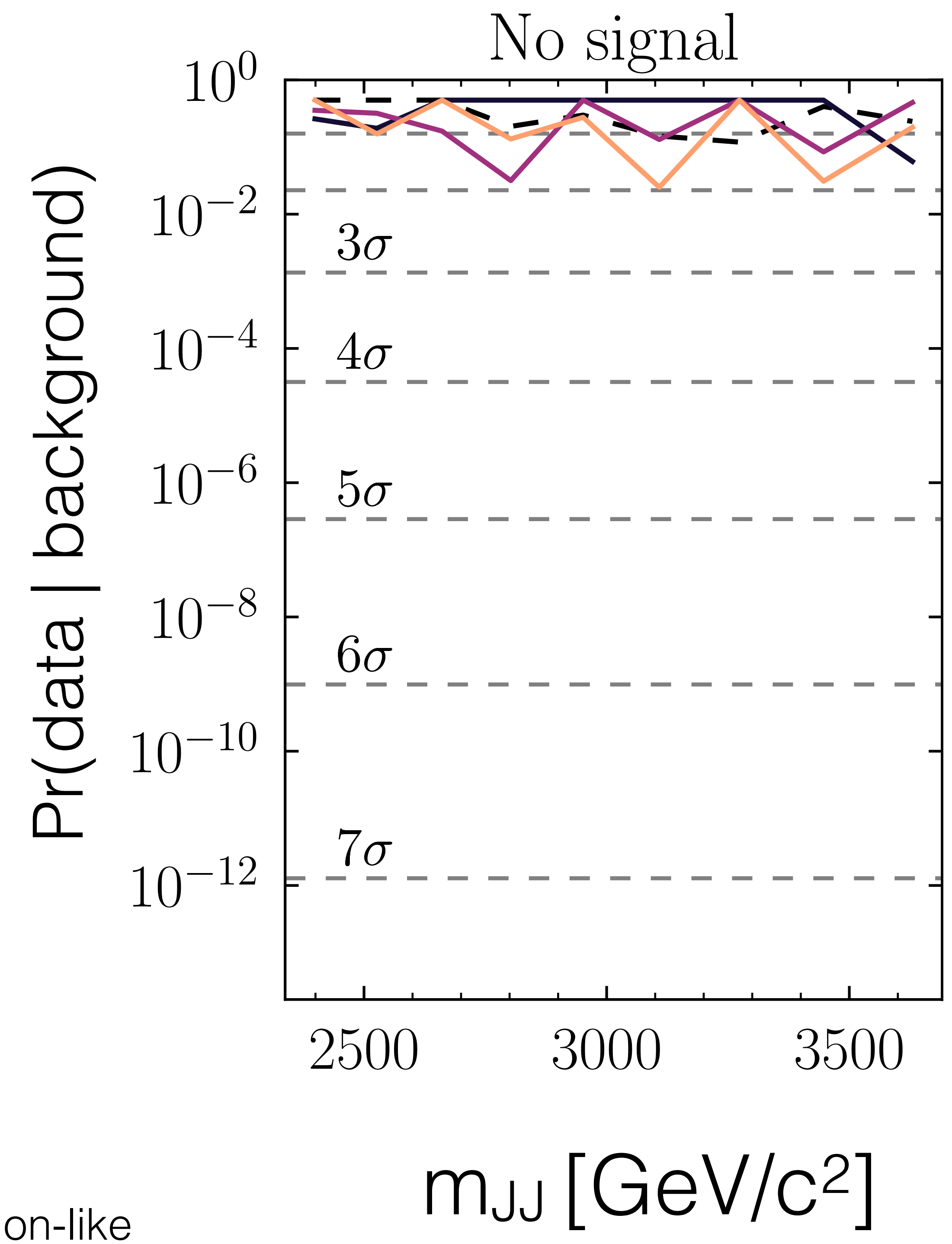
no cut on NN
most 10% signal-region-like
most 1% signal-region-like
most 0.2% signal-region-like

Example: two-jet search



Example: two-jet search

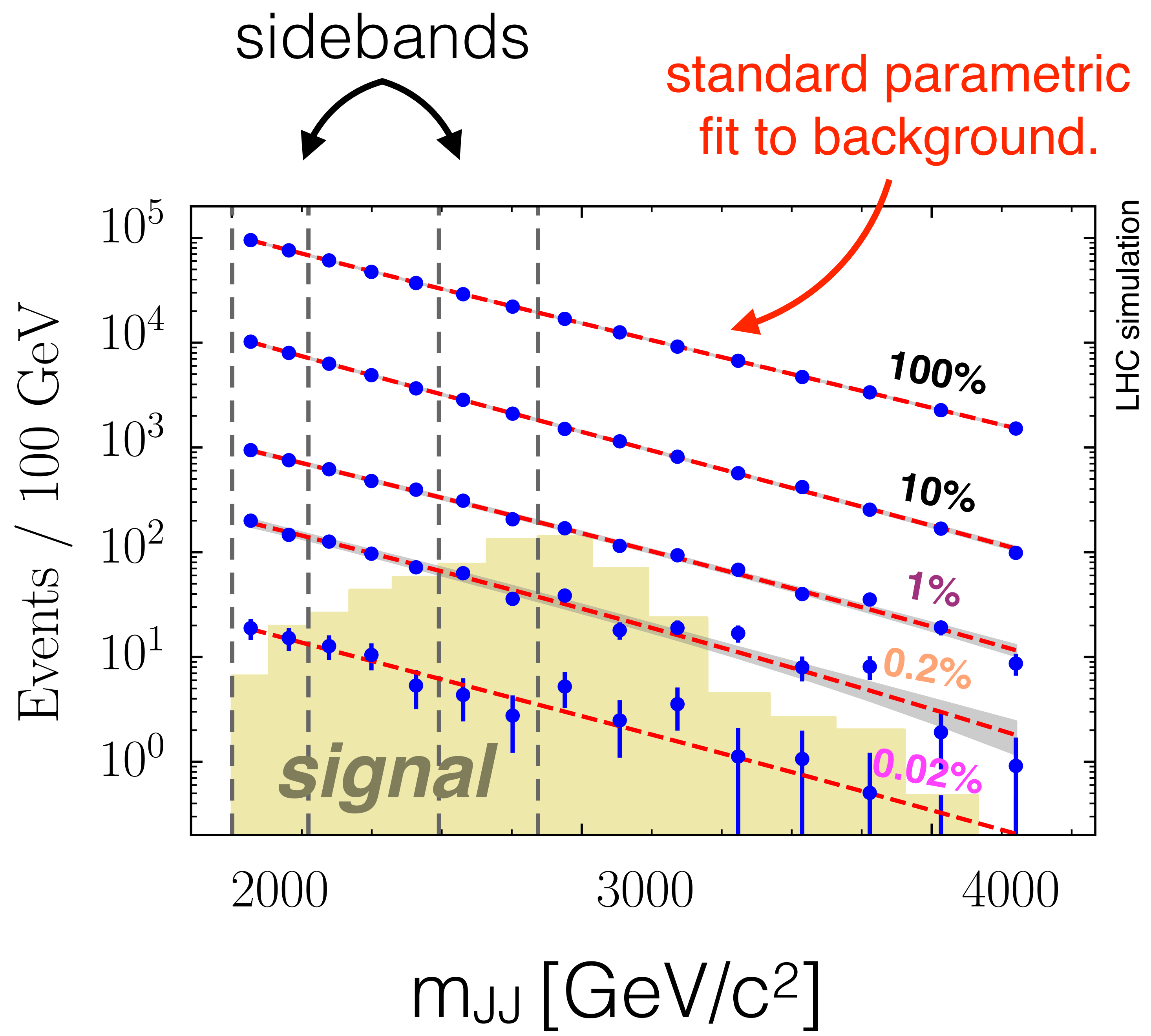


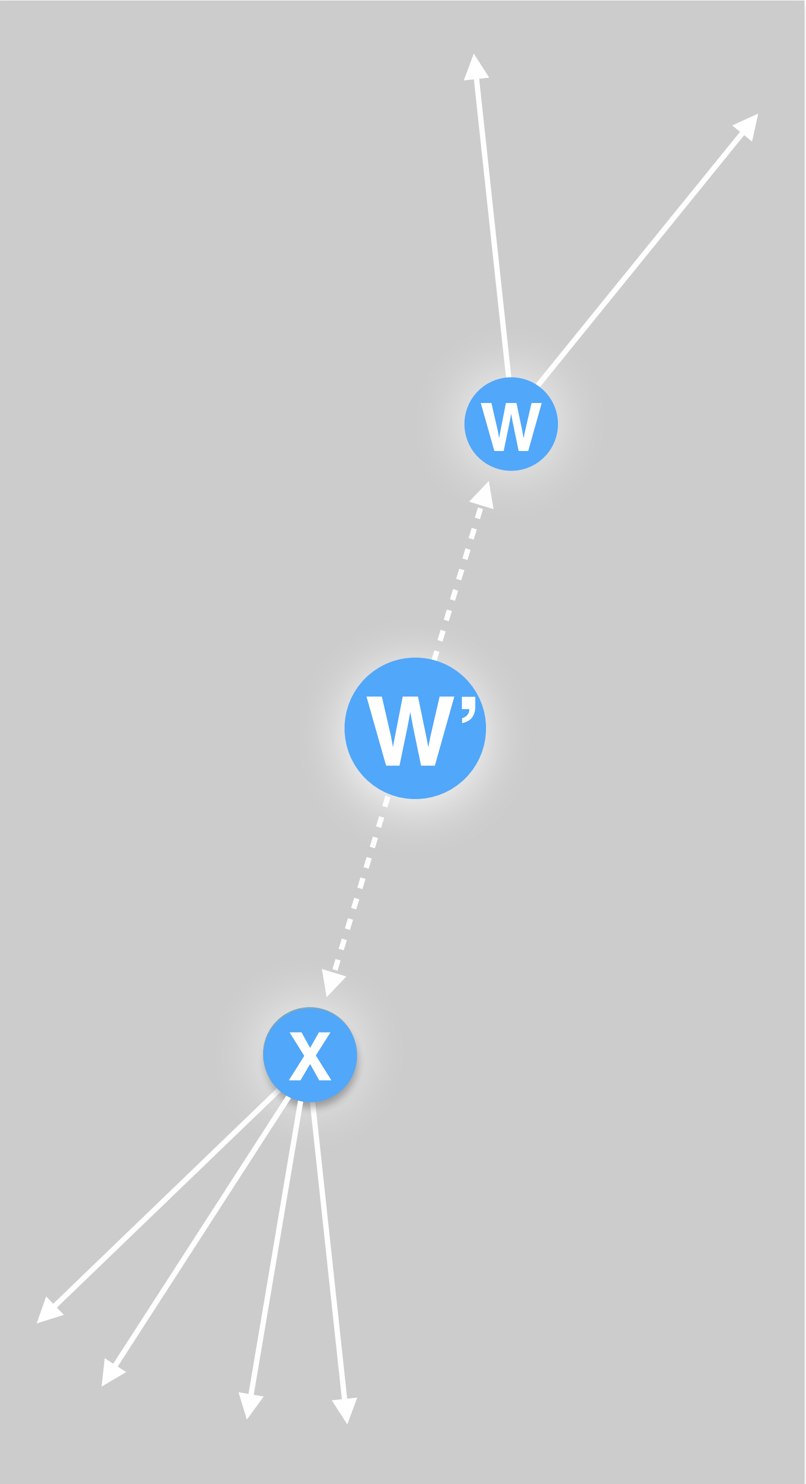


no cut on NN — most 1% signal-region-like

most 10% signal-region-like — most 0.2% signal-region-like

…and when there is a signal?





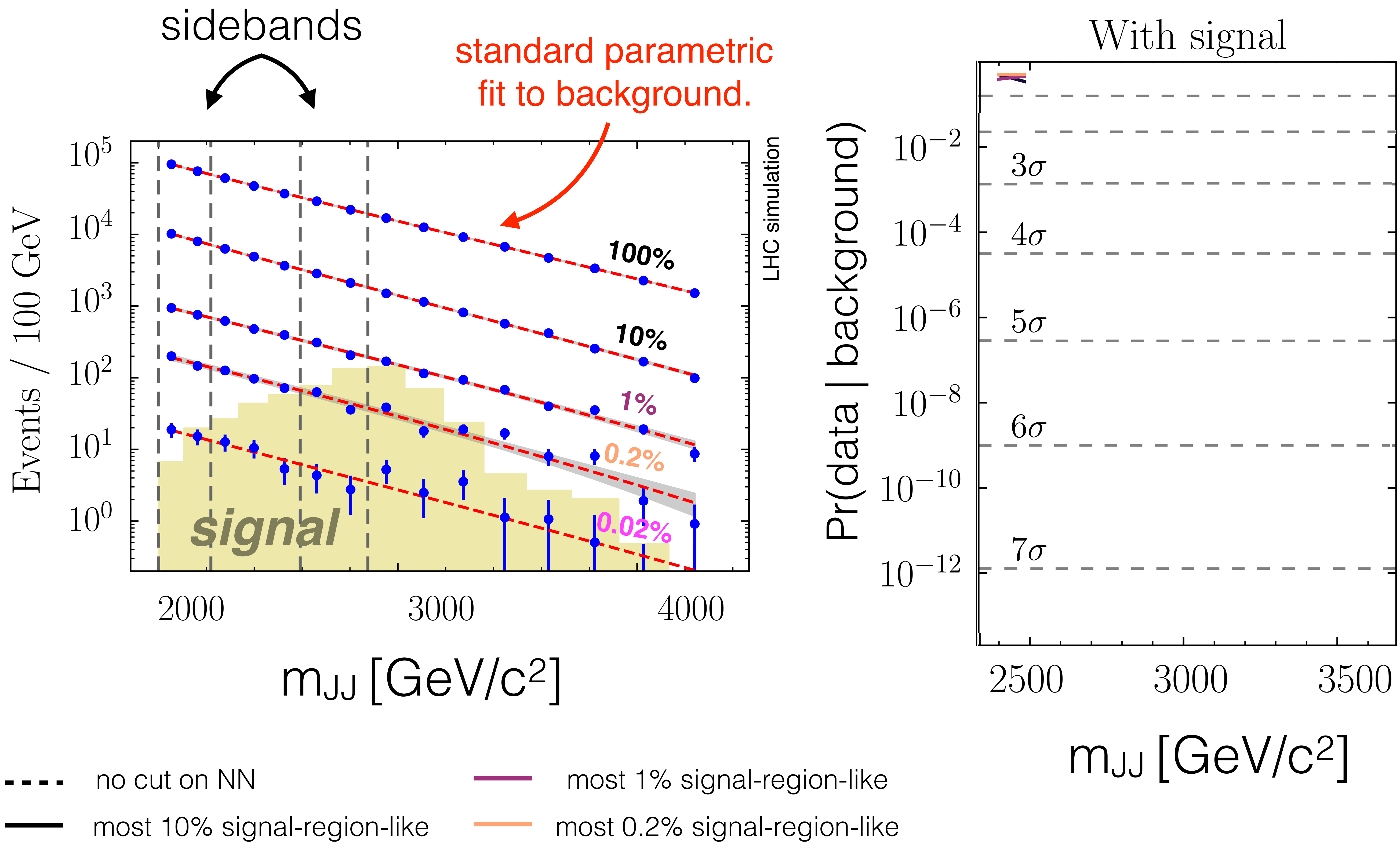
no cut on NN

most 10% signal-region-like

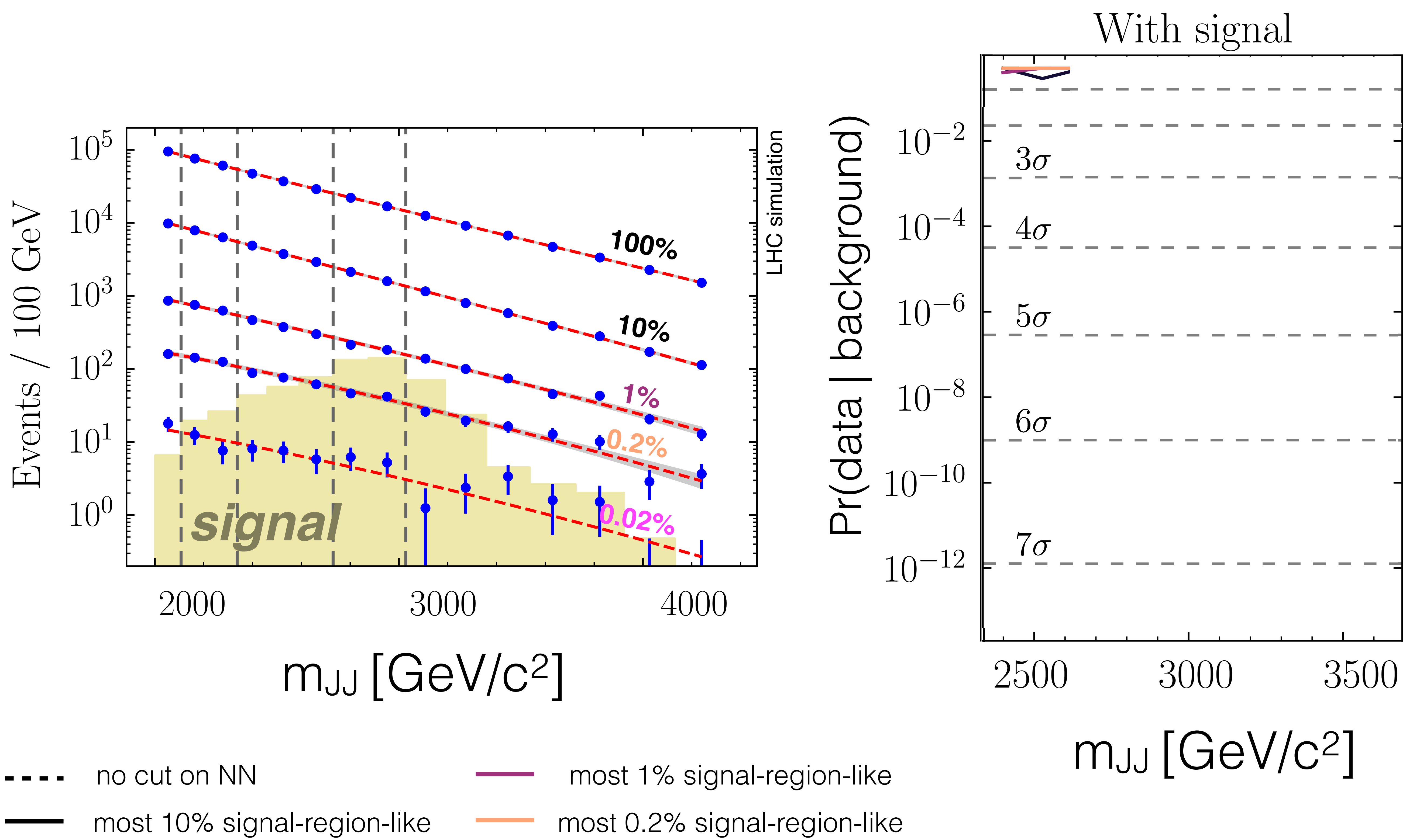
most 1% signal-region-like

most 0.2% signal-region-like

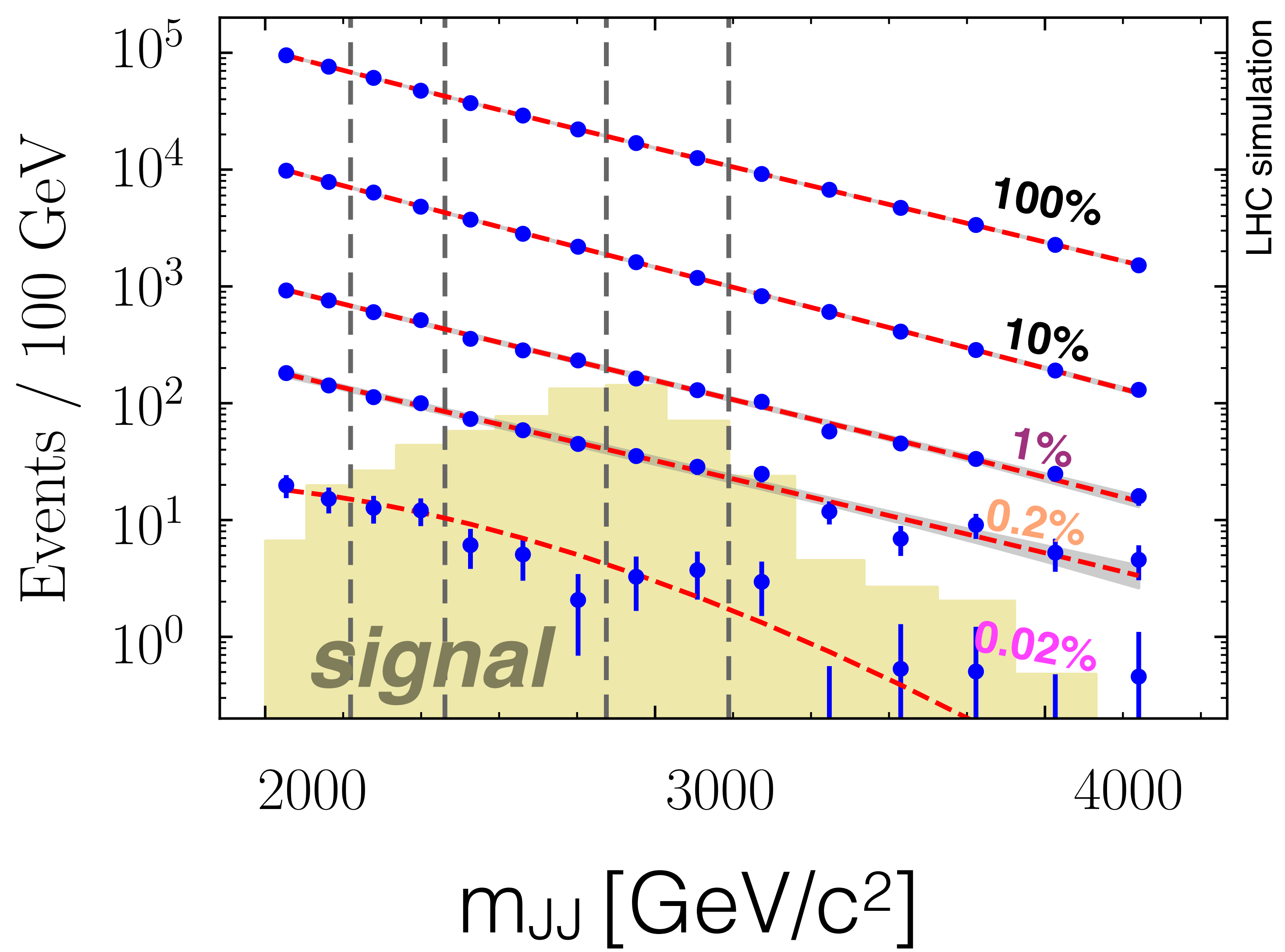
…and when there is a signal?

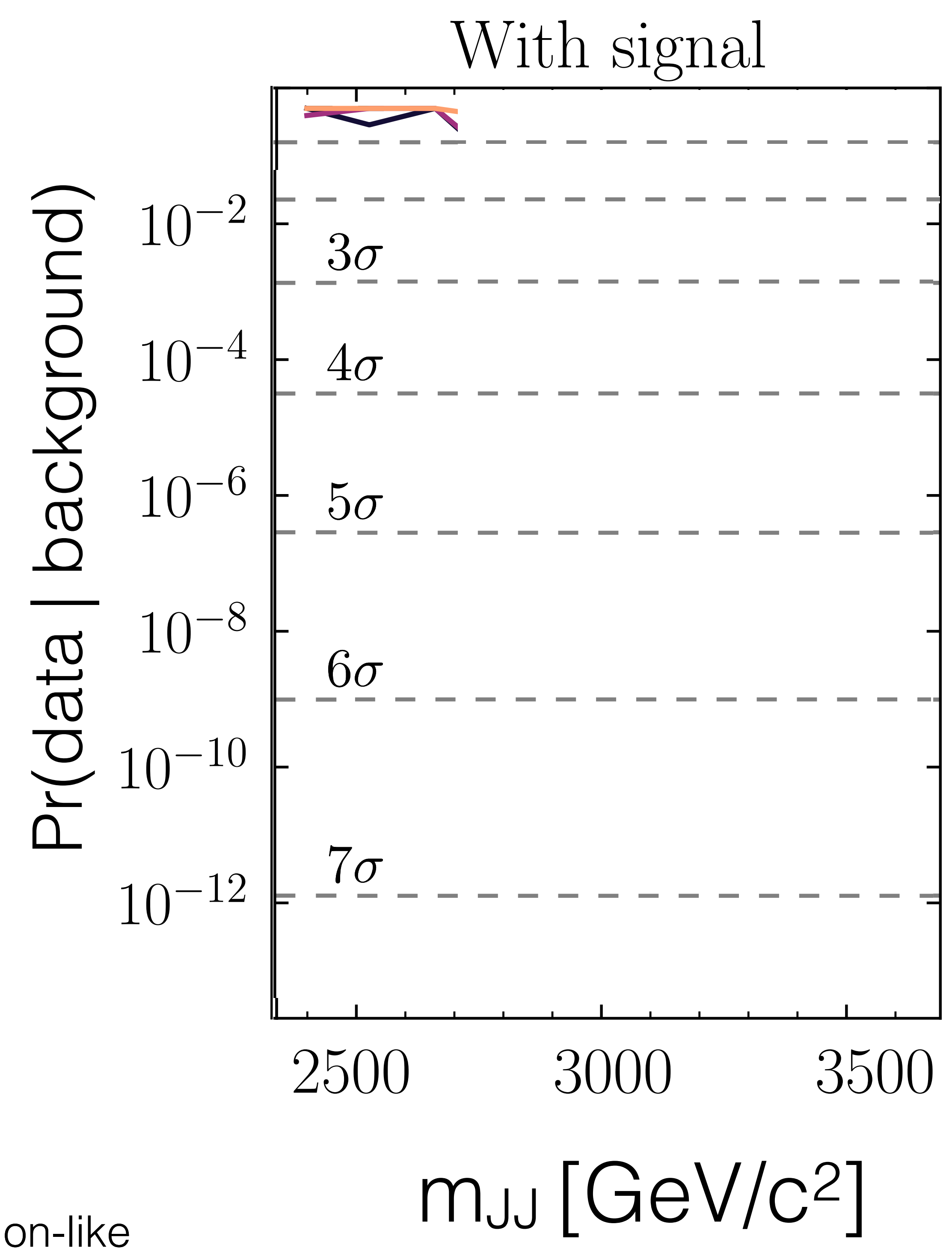


…and when there is a signal?



…and when there is a signal?





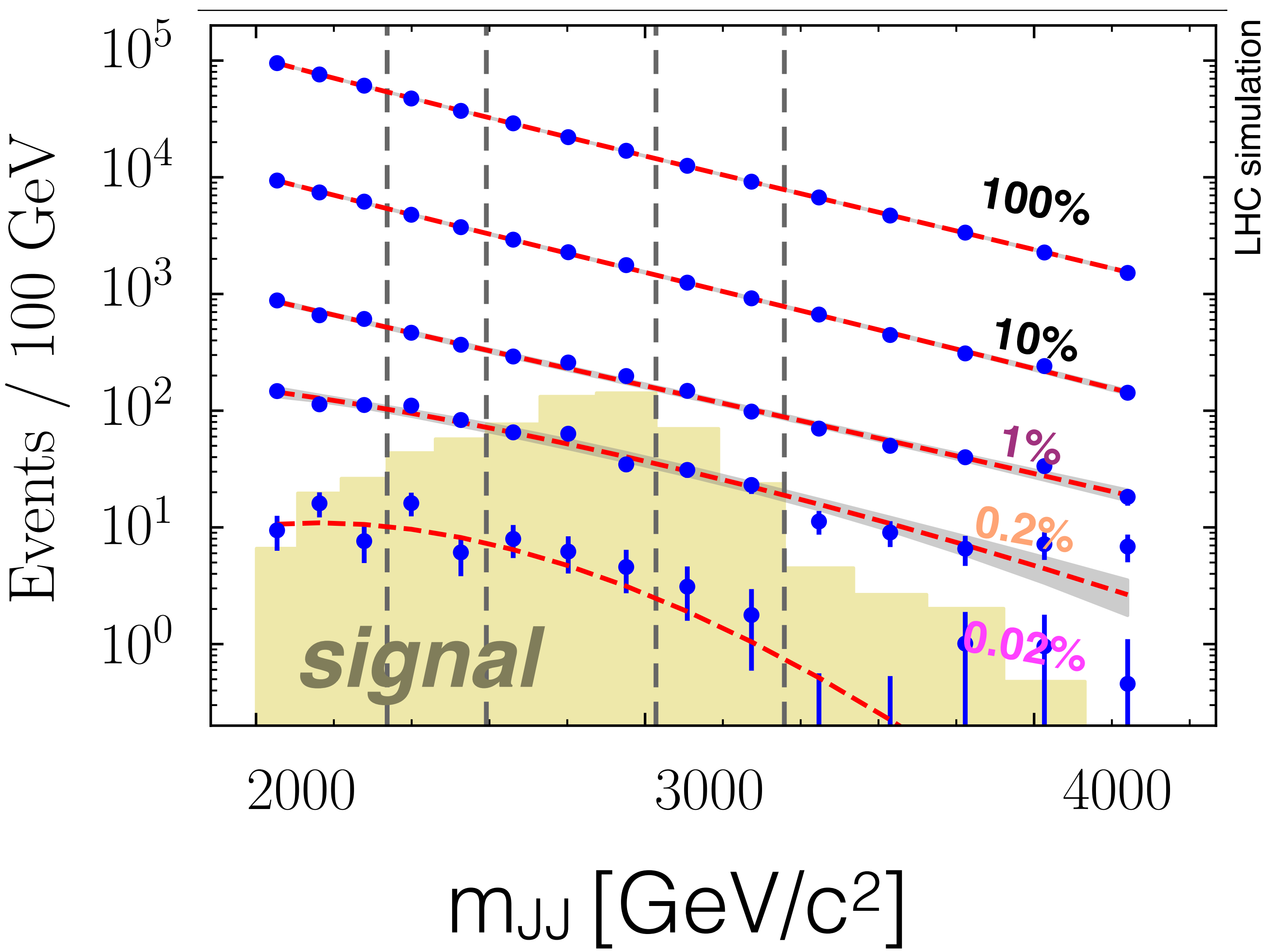
no cut on NN

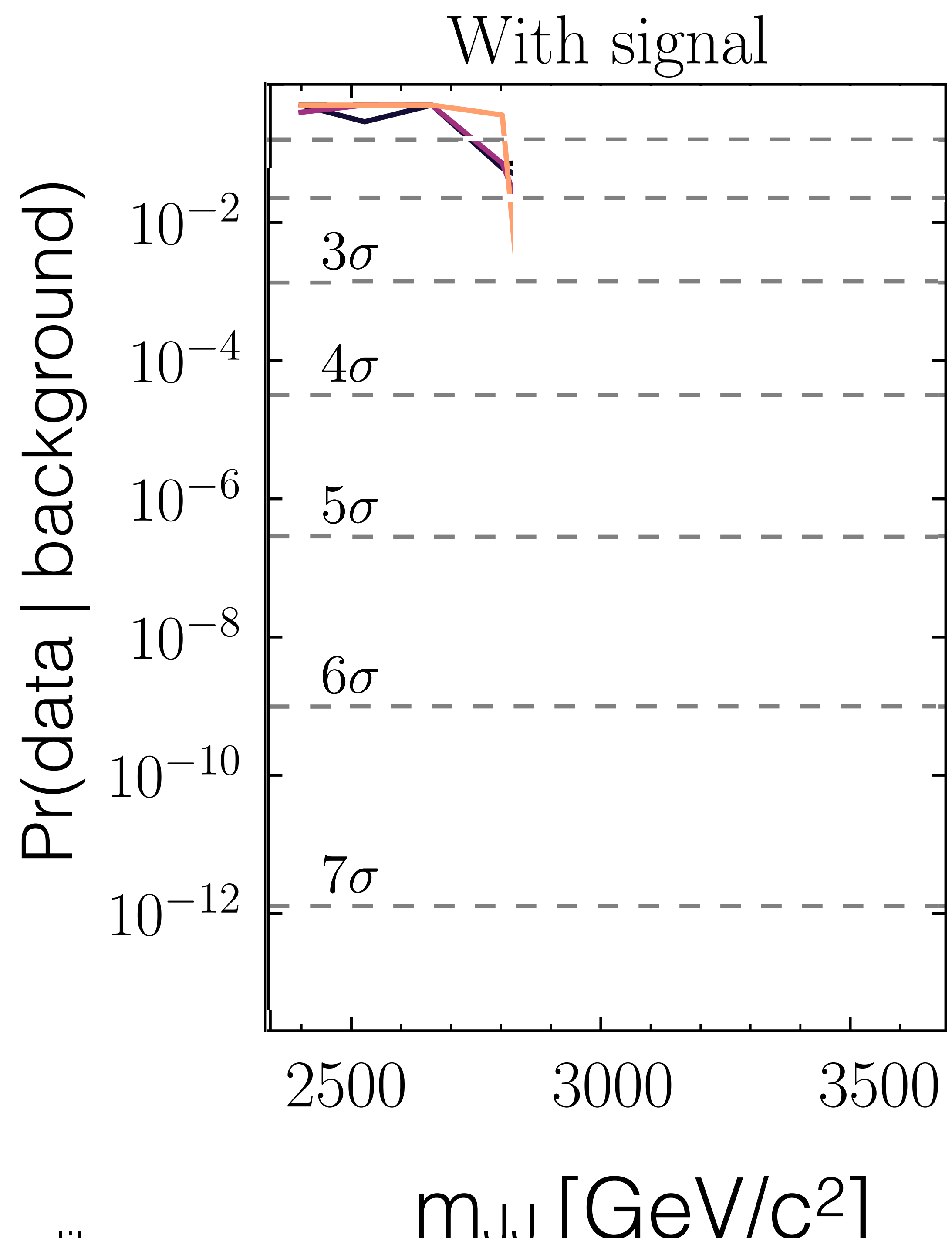
most 10% signal-region-like

most 1% signal-region-like

most 0.2% signal-region-like

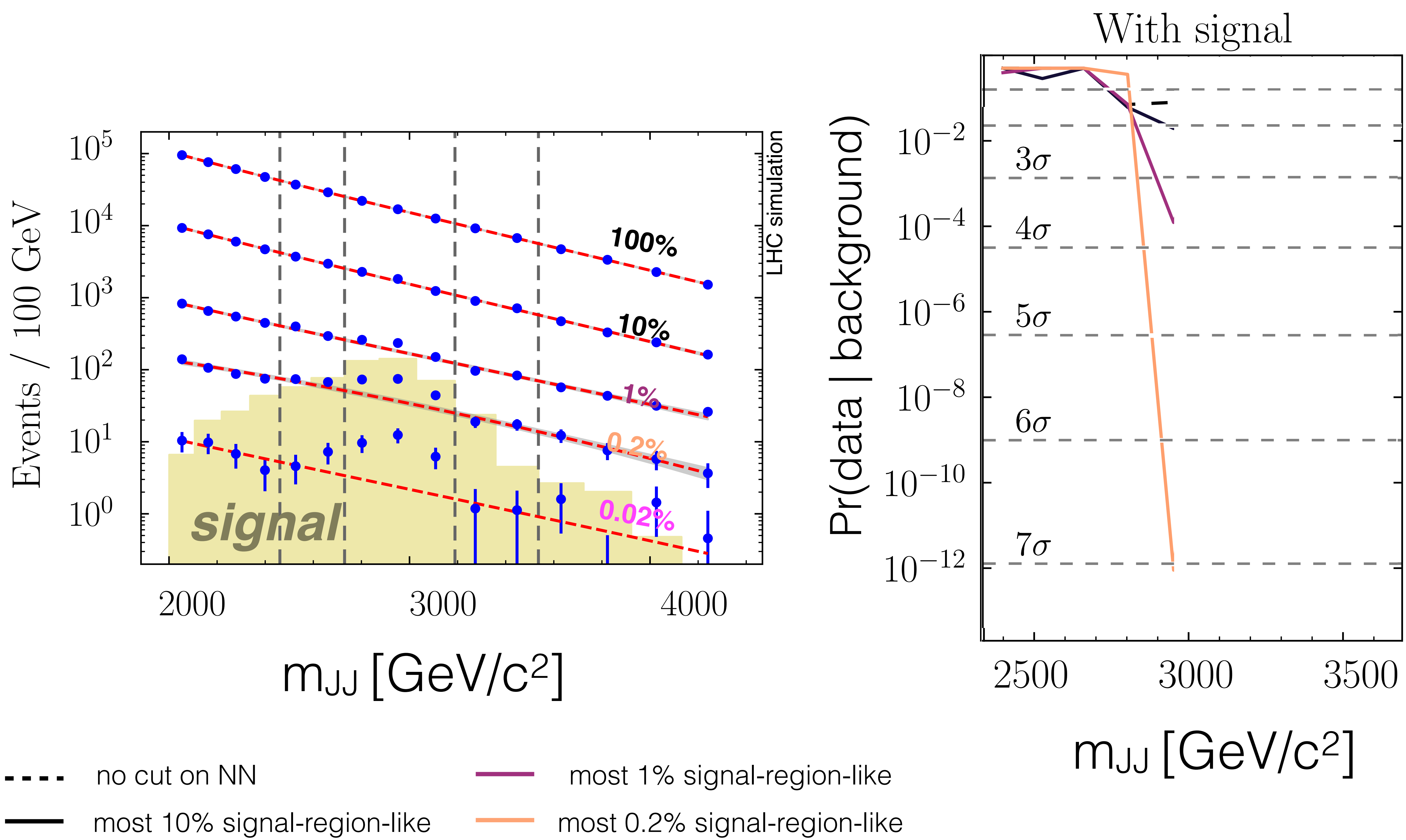
…and when there is a signal?



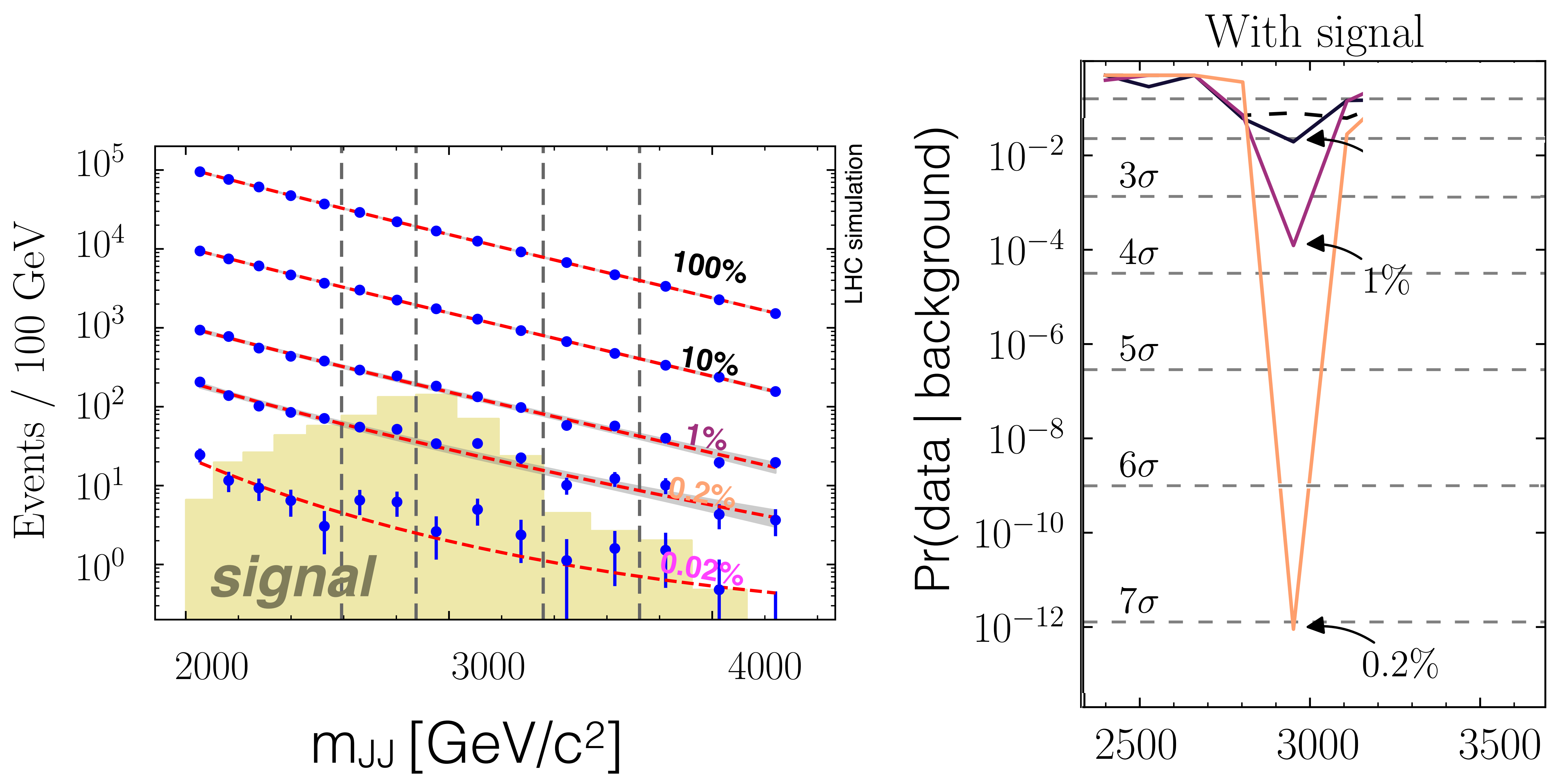


- - - no cut on NN
— most 10% signal-region-like
— most 1% signal-region-like
— most 0.2% signal-region-like

…and when there is a signal?

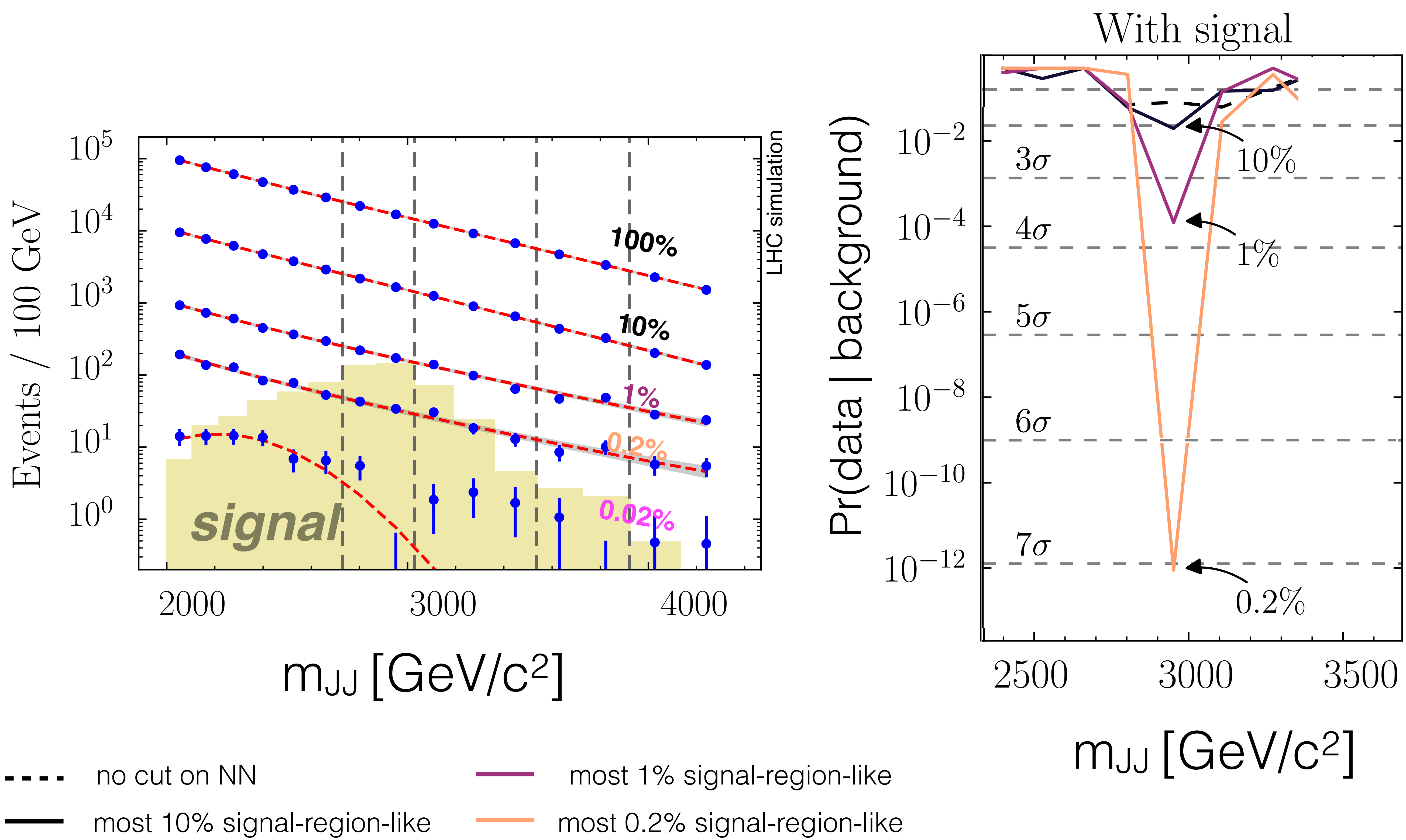


…and when there is a signal?

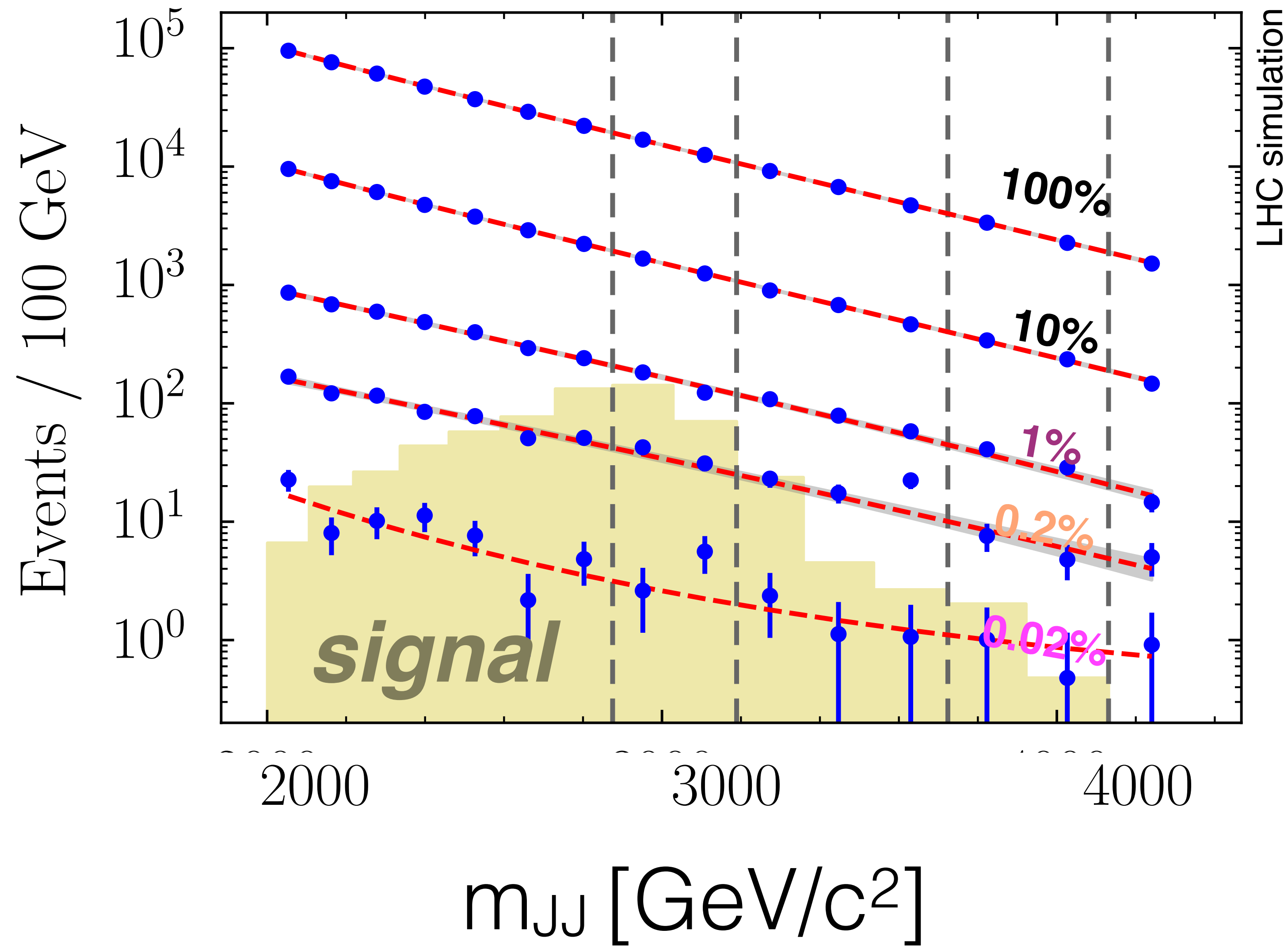


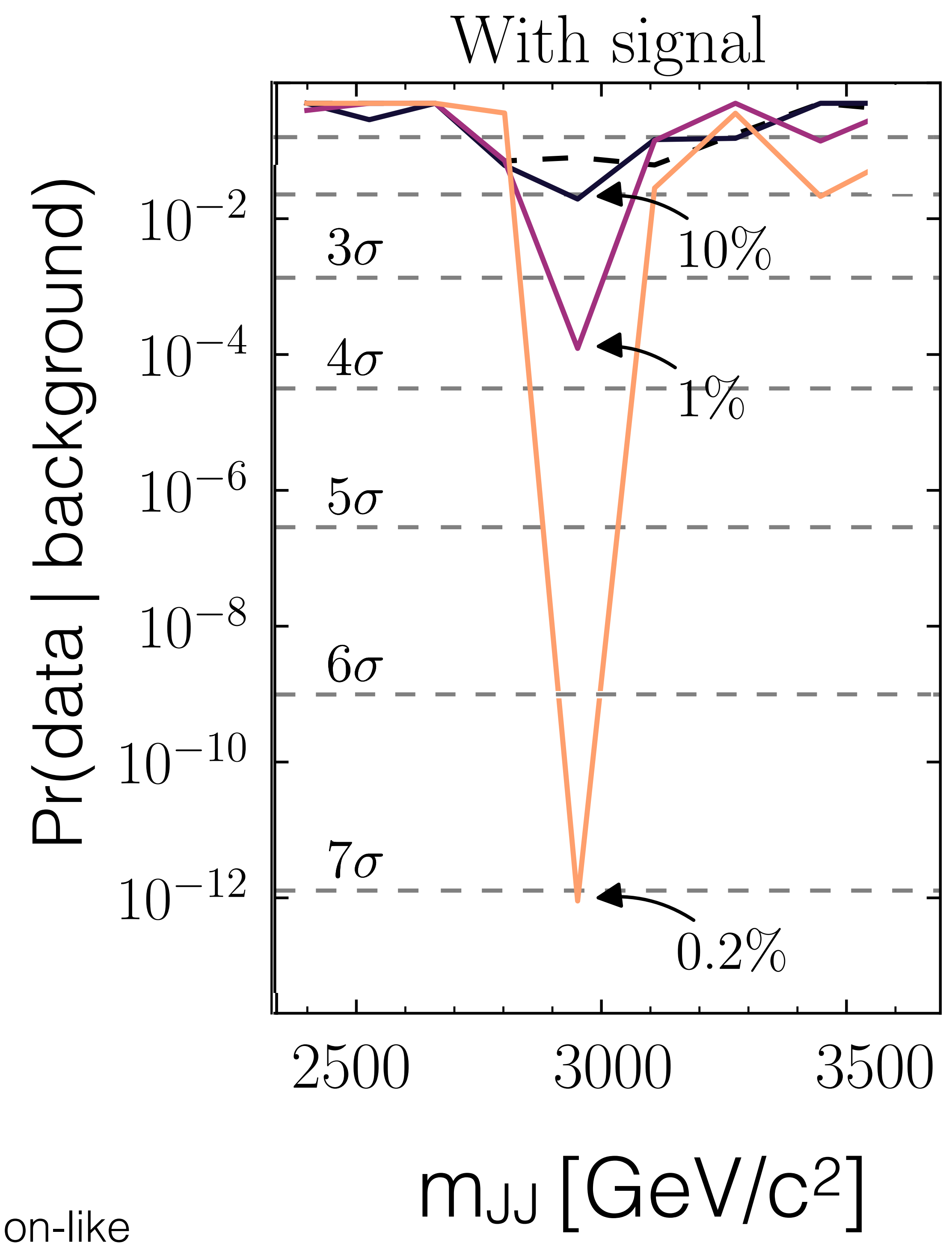
- - - no cut on NN
— most 10% signal-region-like
— most 1% signal-region-like
— most 0.2% signal-region-like

…and when there is a signal?



…and when there is a signal?





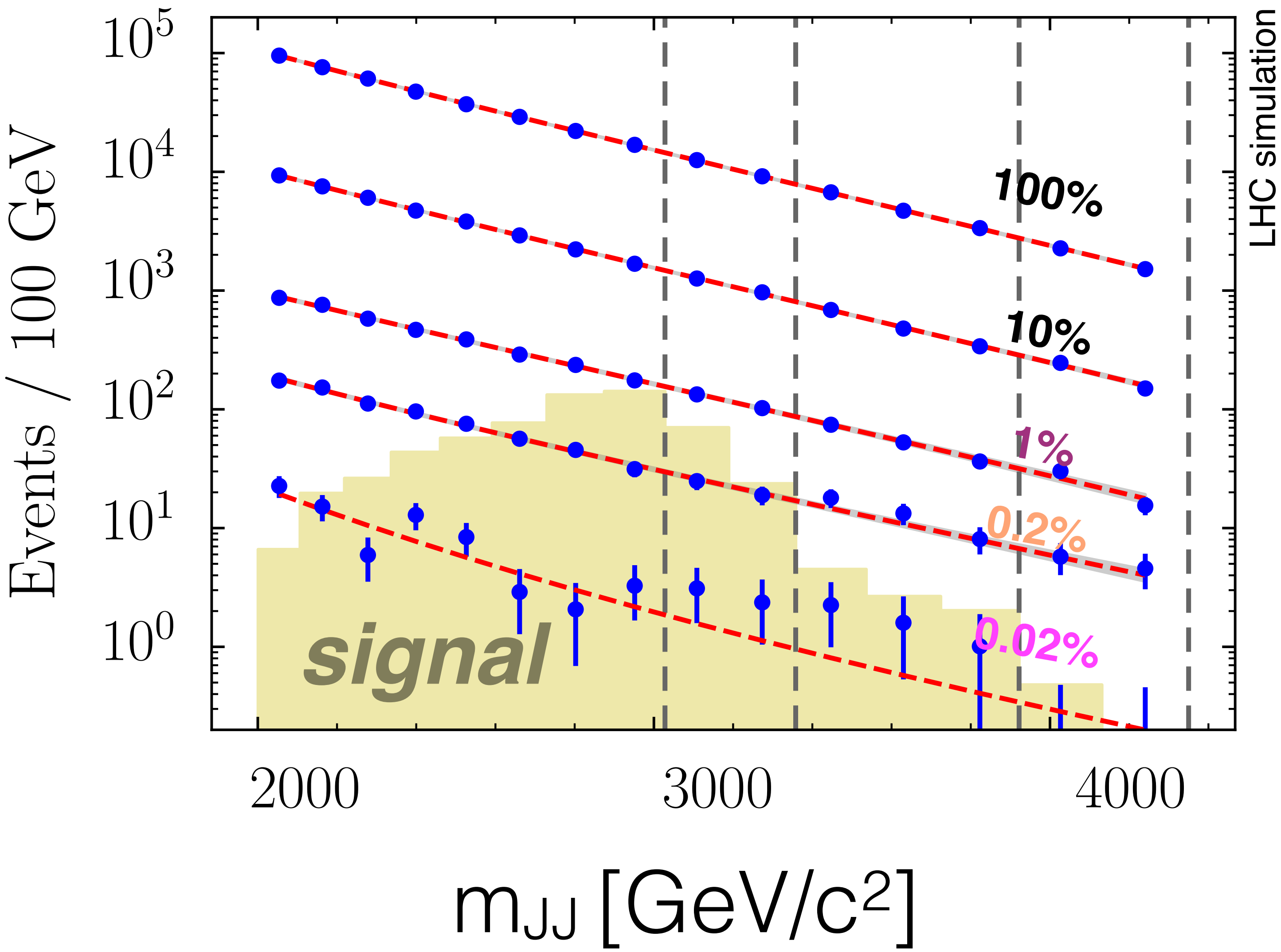
no cut on NN

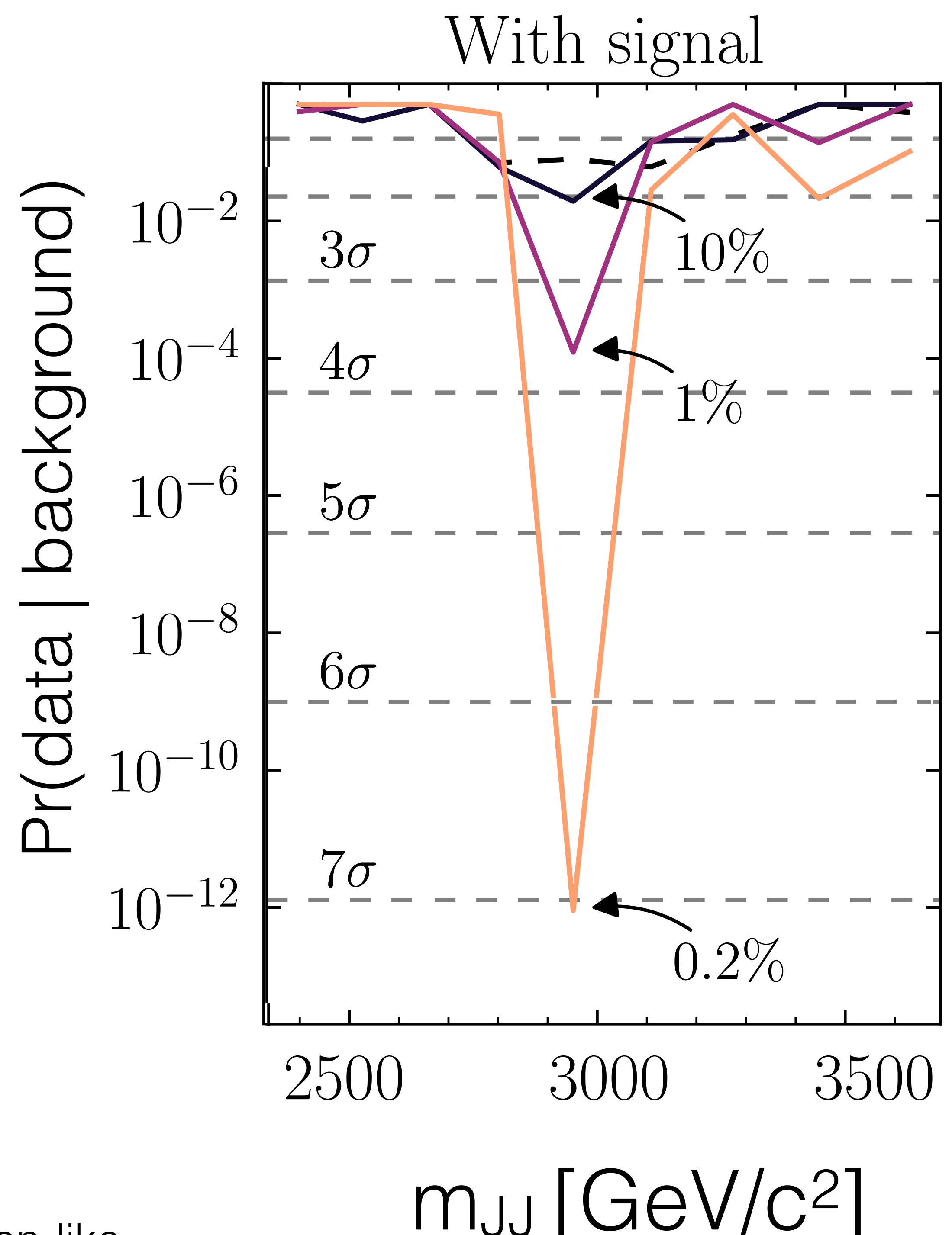
most 1% signal-region-like

most 10% signal-region-like

most 0.2% signal-region-like

…and when there is a signal?



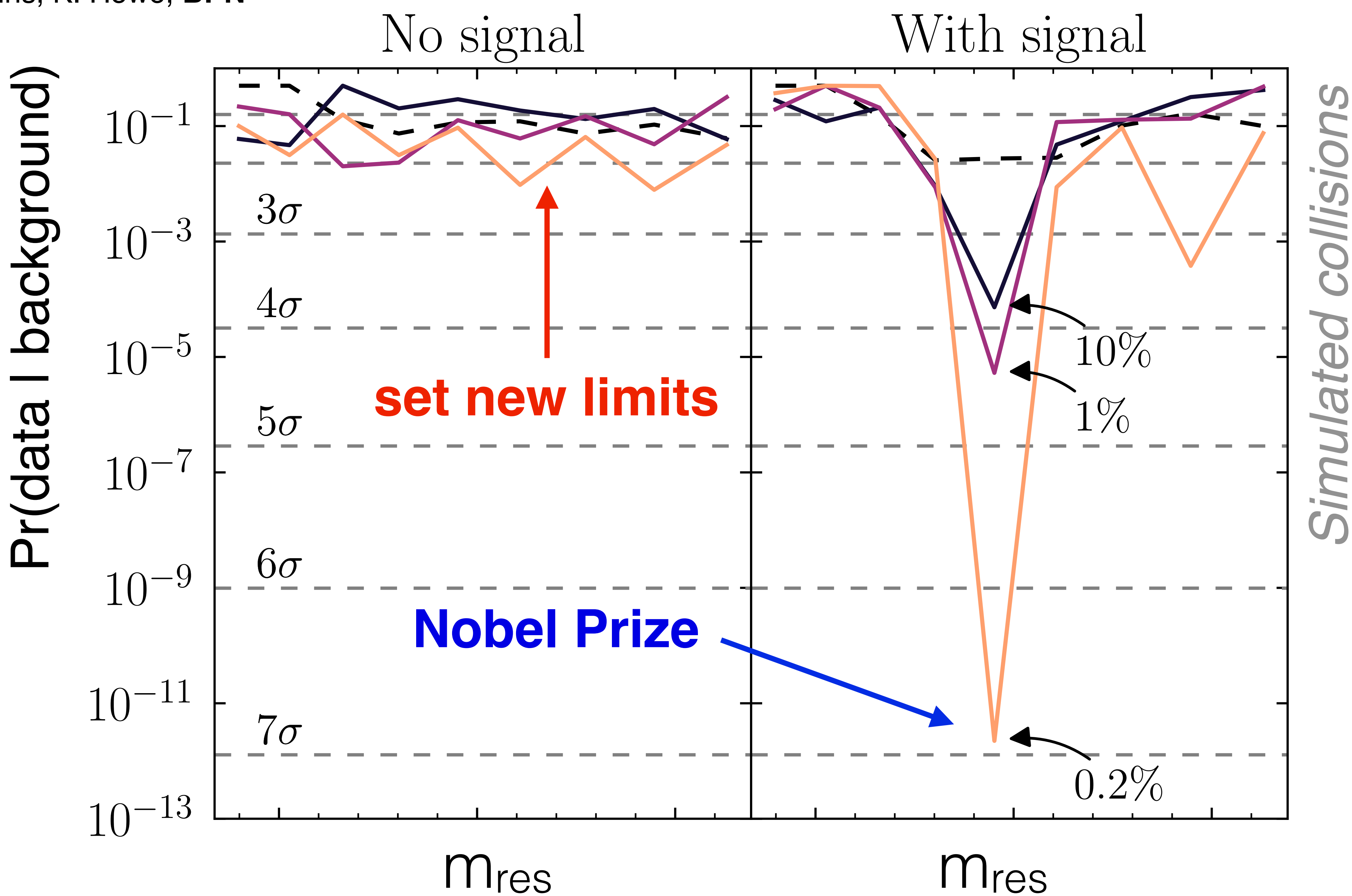


no cut on NN
most 10% signal-region-like
most 1% signal-region-like
most 0.2% signal-region-like

CWoLa hunting: overview

Phys. Rev. Lett. 121 (2018) 241803

J. Collins, K. Howe, **BPN**



*Aviv Cukierman + **BPN** + ATLAS Collaboration

Our first data result from **ATLAS** will come out this spring!*

Tying it all together: DCTR + anomalies

New

We want to reduce our dependence on simulation, but we also don't want to throw away our physics priors!

Tying it all together: DCTR + anomalies

We want to reduce our dependence on simulation, but we also don't want to throw away our physics priors!

Can we use simulations in a way that is (nearly) simulation-independent?

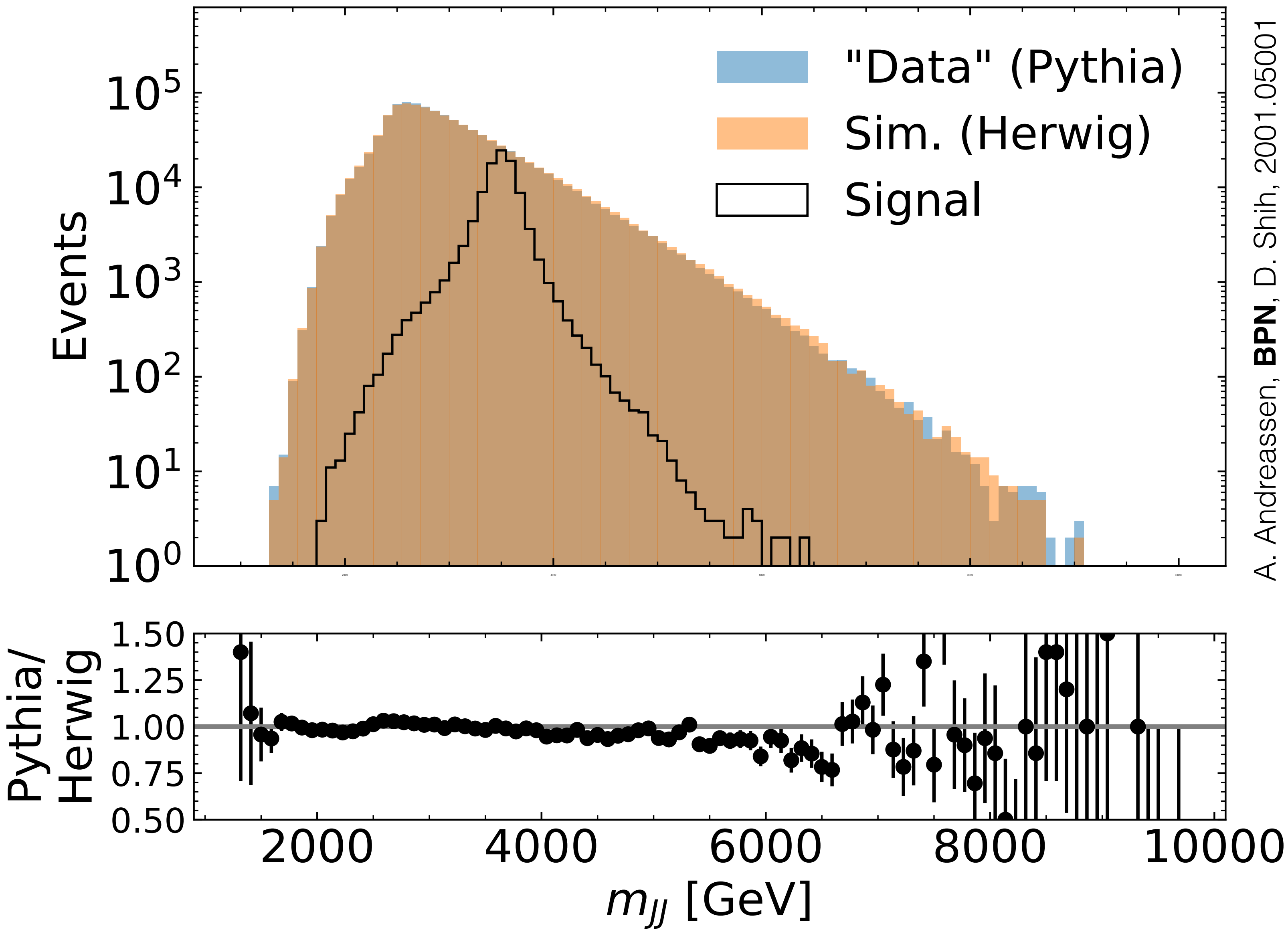
Tying it all together: DCTR + anomalies

We want to reduce our dependence on simulation, but we also don't want to throw away our physics priors!

Can we use simulations in a way that is (nearly) simulation-independent?

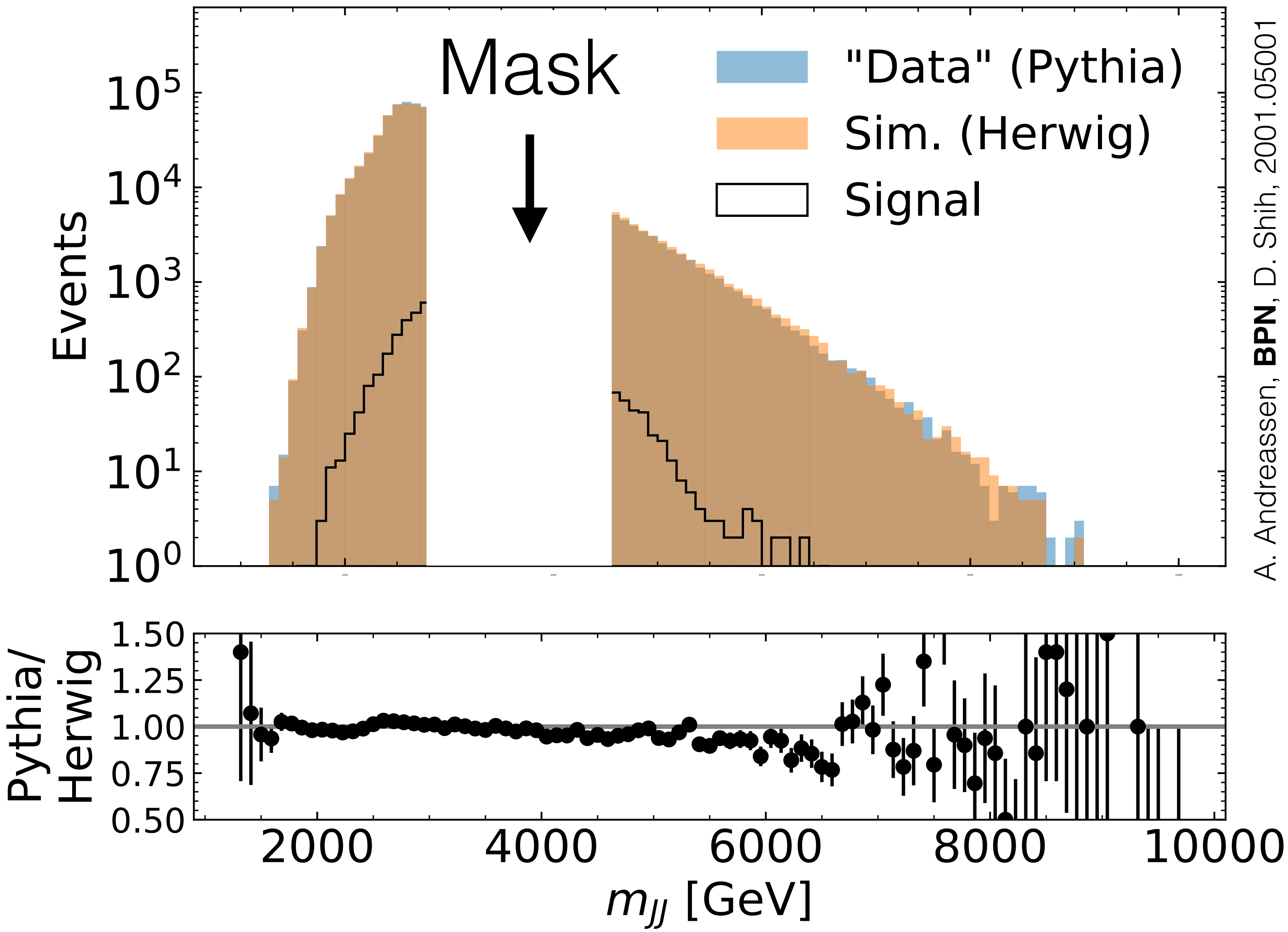
Answer:
Simulation Assisted Likelihood-free Anomaly Detection
(aka SALAD)

SALAD

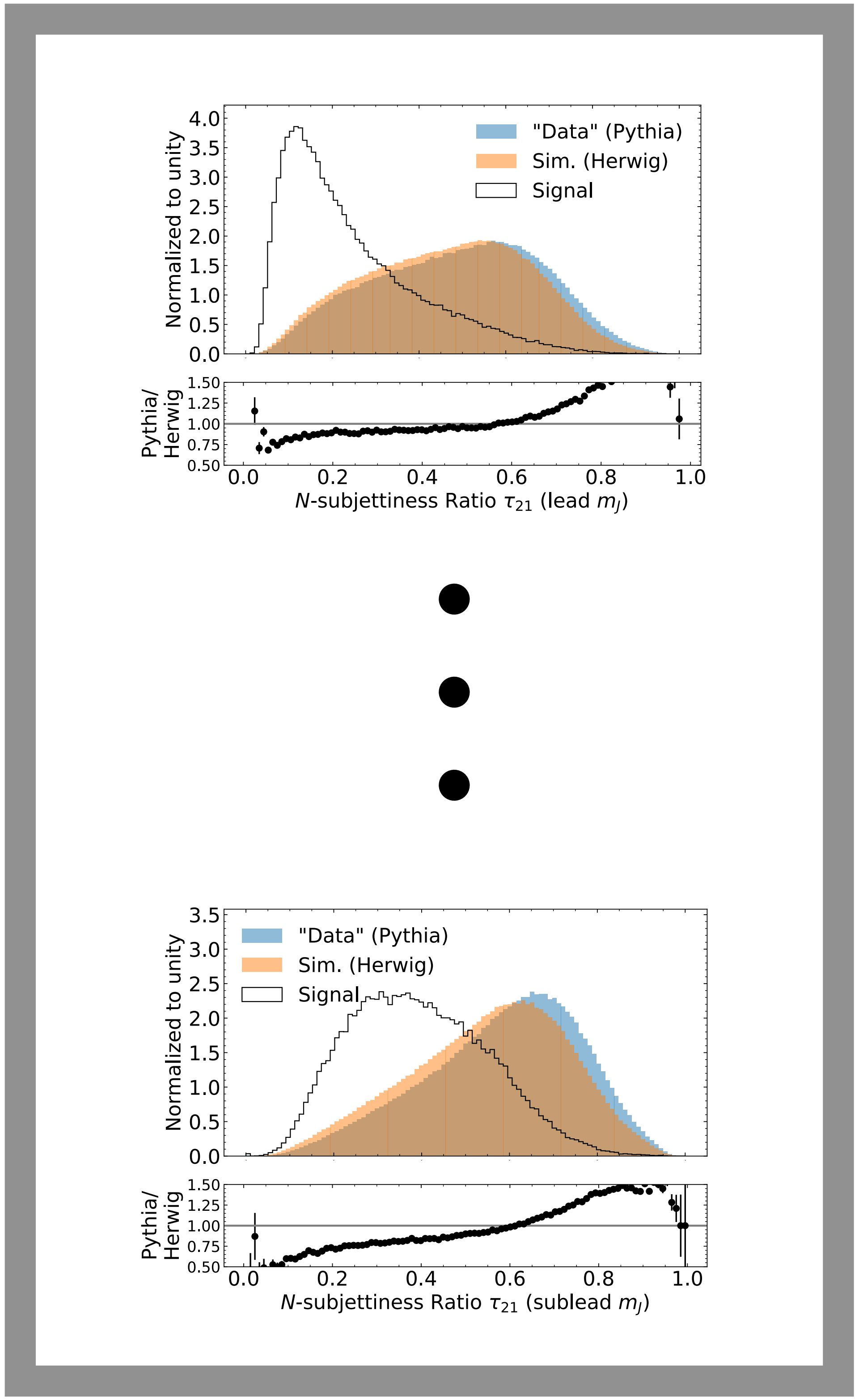


A. Andreassen, **BPN**, D. Shih, 2001.05001

SALAD

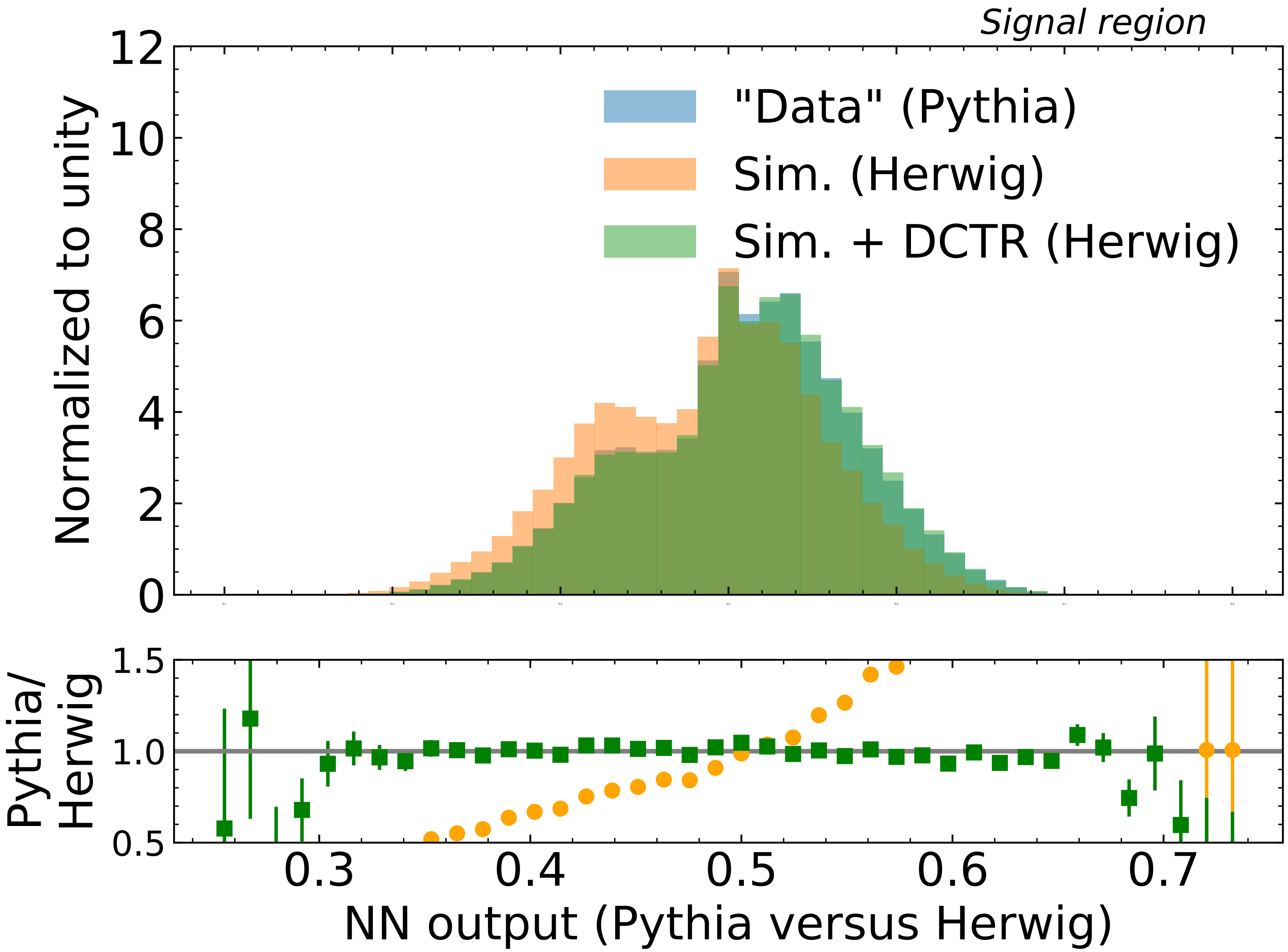


Features



(1) Train DCTR in the sidebands to reweight MC to data

SALAD

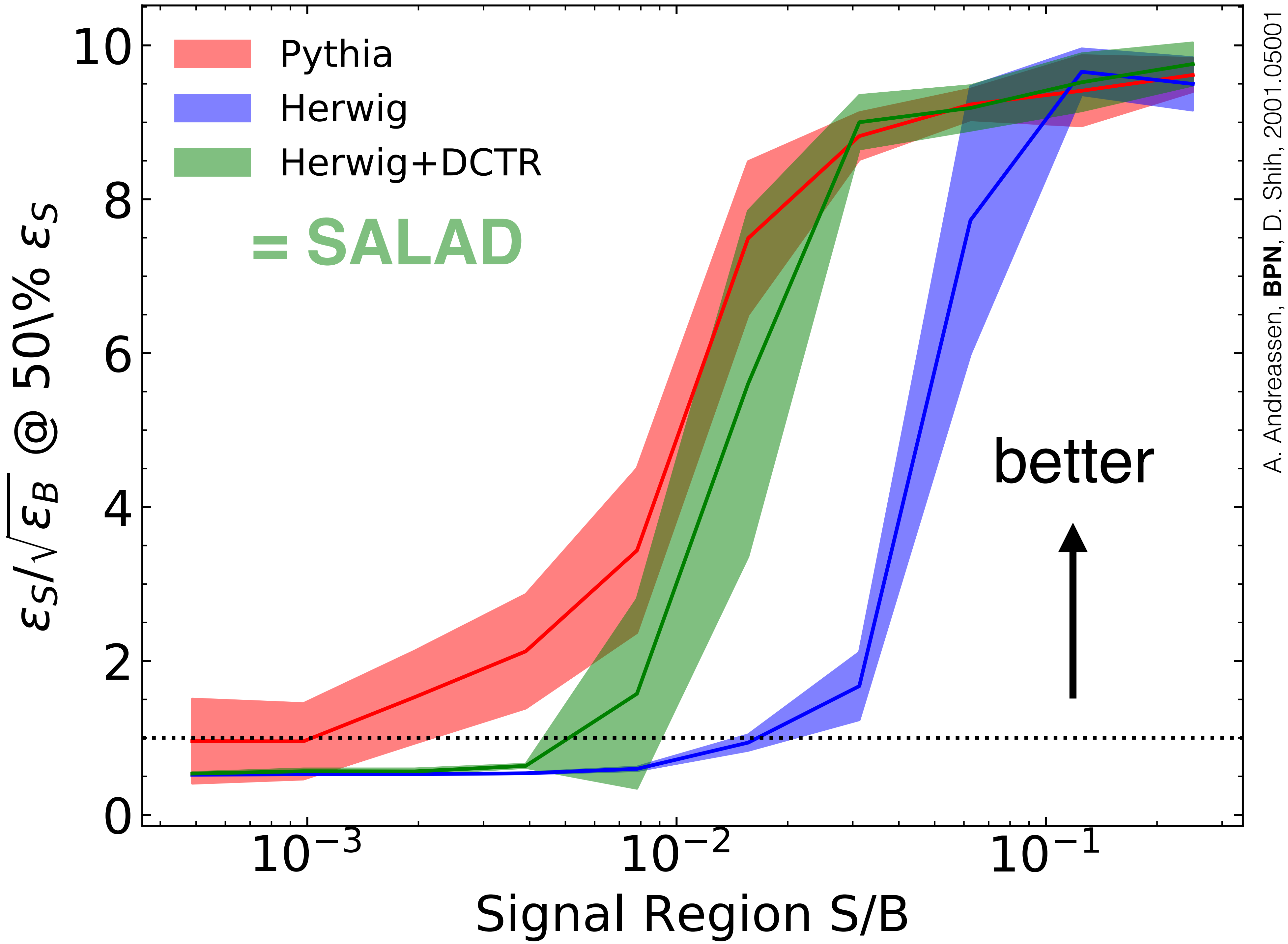


A. Andreassen, **BPN**, D. Shih, 2001.05001

(2) Interpolate DCTR to signal region

(3) Train classifier to distinguish reweighted MC from data

SALAD



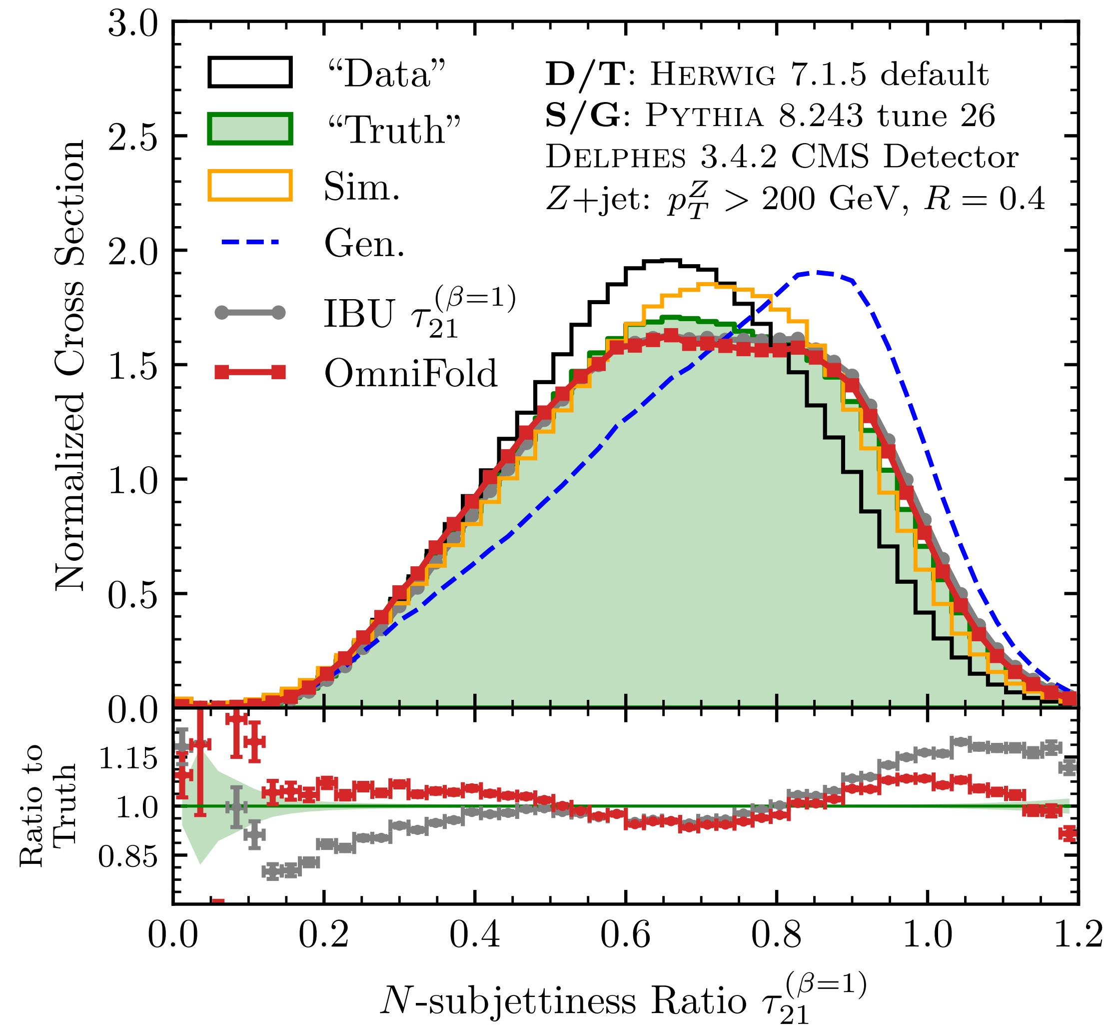
Can also use SALAD for background estimation - see backup

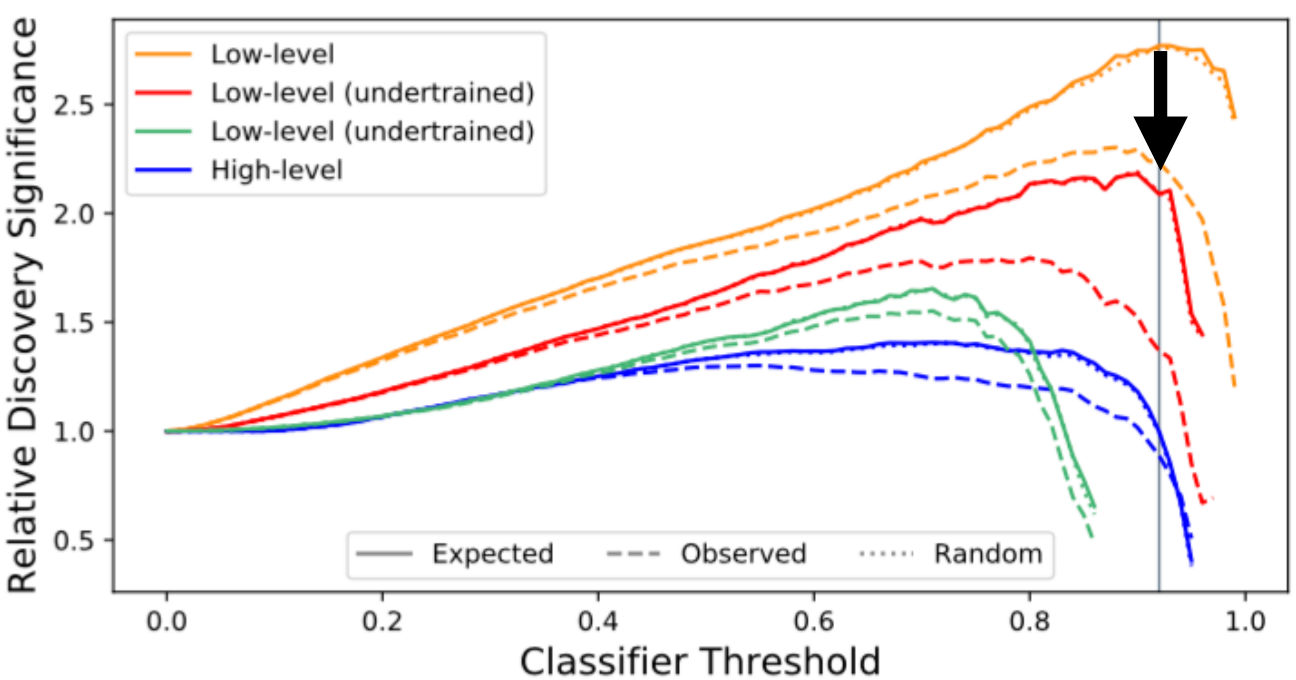
Outline for today

- (Optimally) using NNs for analysis
    - Improving search sensitivity
    - Enhancing SM measurements **DCTR, OmniFold**
- Uncertainties with NNs
    - What are they? **Optimality/Precision uncertainties**
    - How to improve (or avoid)? **Adversarial upper bounds**
- Anomaly detection **Weak supervision** **CWoLa (hunting), SALAD**

Conclusions and outlook

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



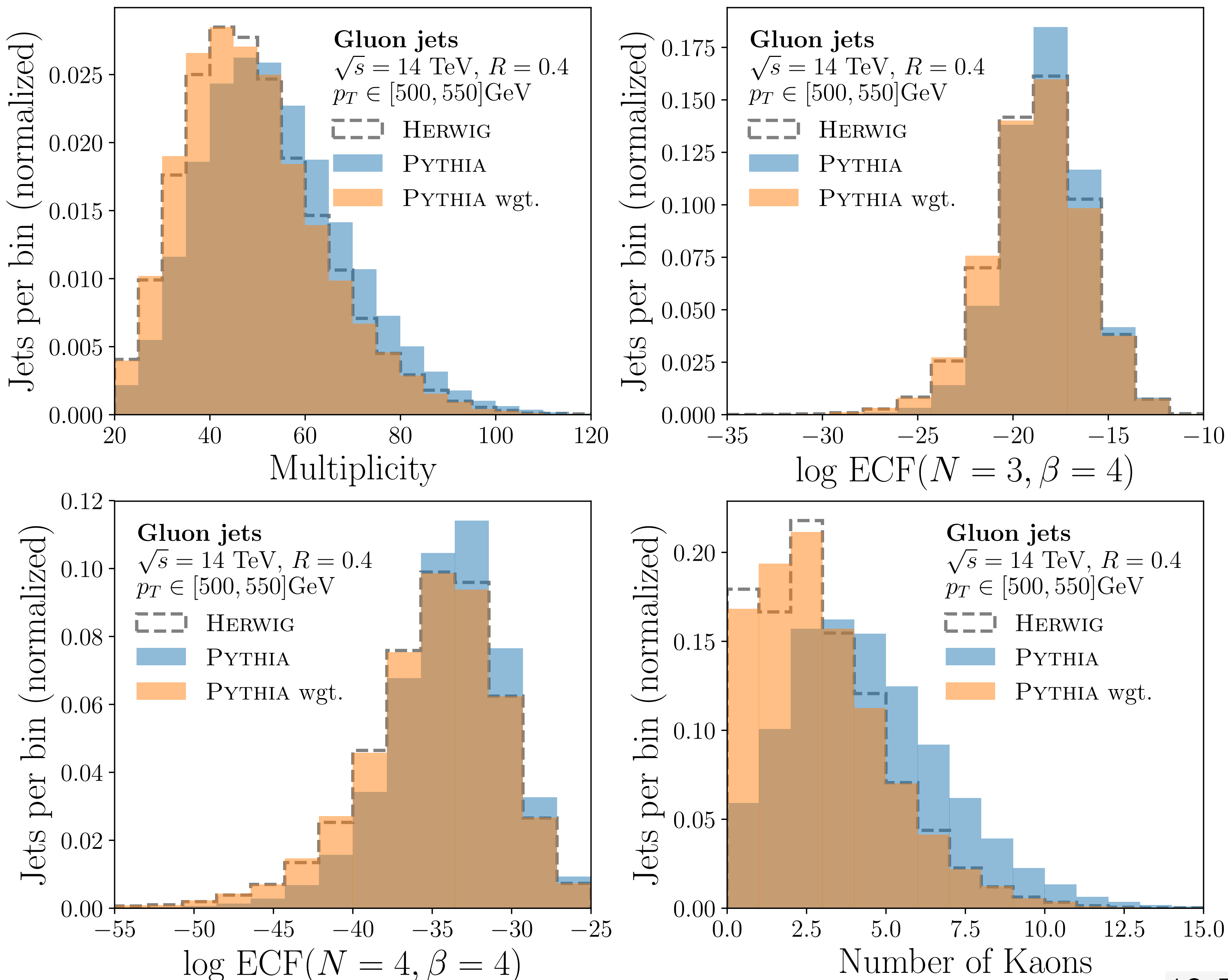


However, we need to be careful about uncertainties - in some cases were are **estimating the wrong ones** and in others, we are **ignoring**!

Backup

Pythia versus Herwig

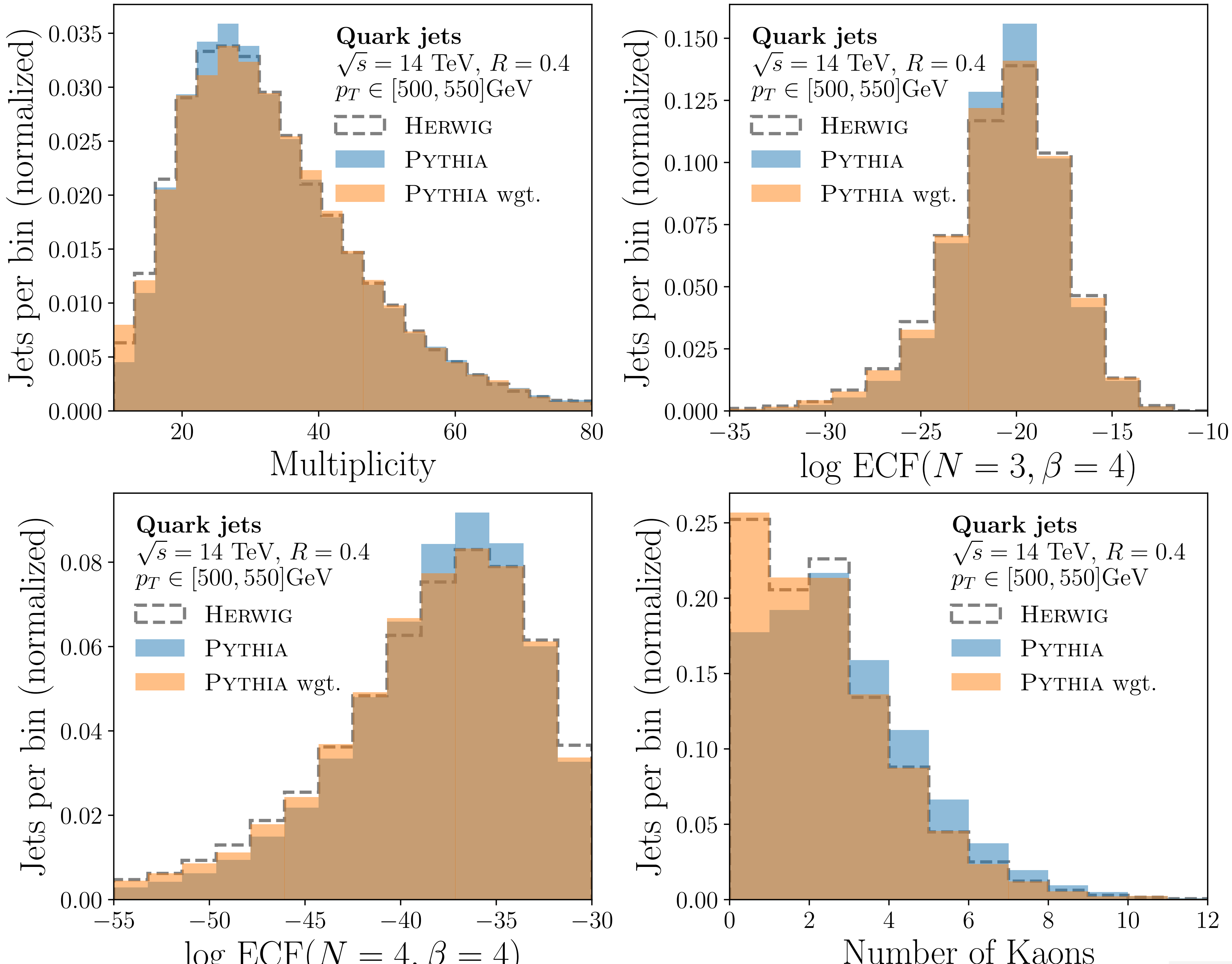
No hyper-parameter tuning - out of the box!



Samples from
10.5281/zenodo.2658764
10.5281/zenodo.3164691

Pythia versus Herwig

No hyper-parameter tuning - out of the box!



Samples from
10.5281/zenodo.2658764
10.5281/zenodo.3164691

Weak supervision take 1: *Learn with proportions*

This is essentially a generalization of the template method.

Standard supervised learning

# of examples for training

loss fcn.

labels

$$f_{\text{full}} = \text{argmin}_{f' : \mathbb{R}^n \to [0,1]} \sum_{i=1}^{N} \ell\left(f'(x_i) - t_i\right)$$

neural network

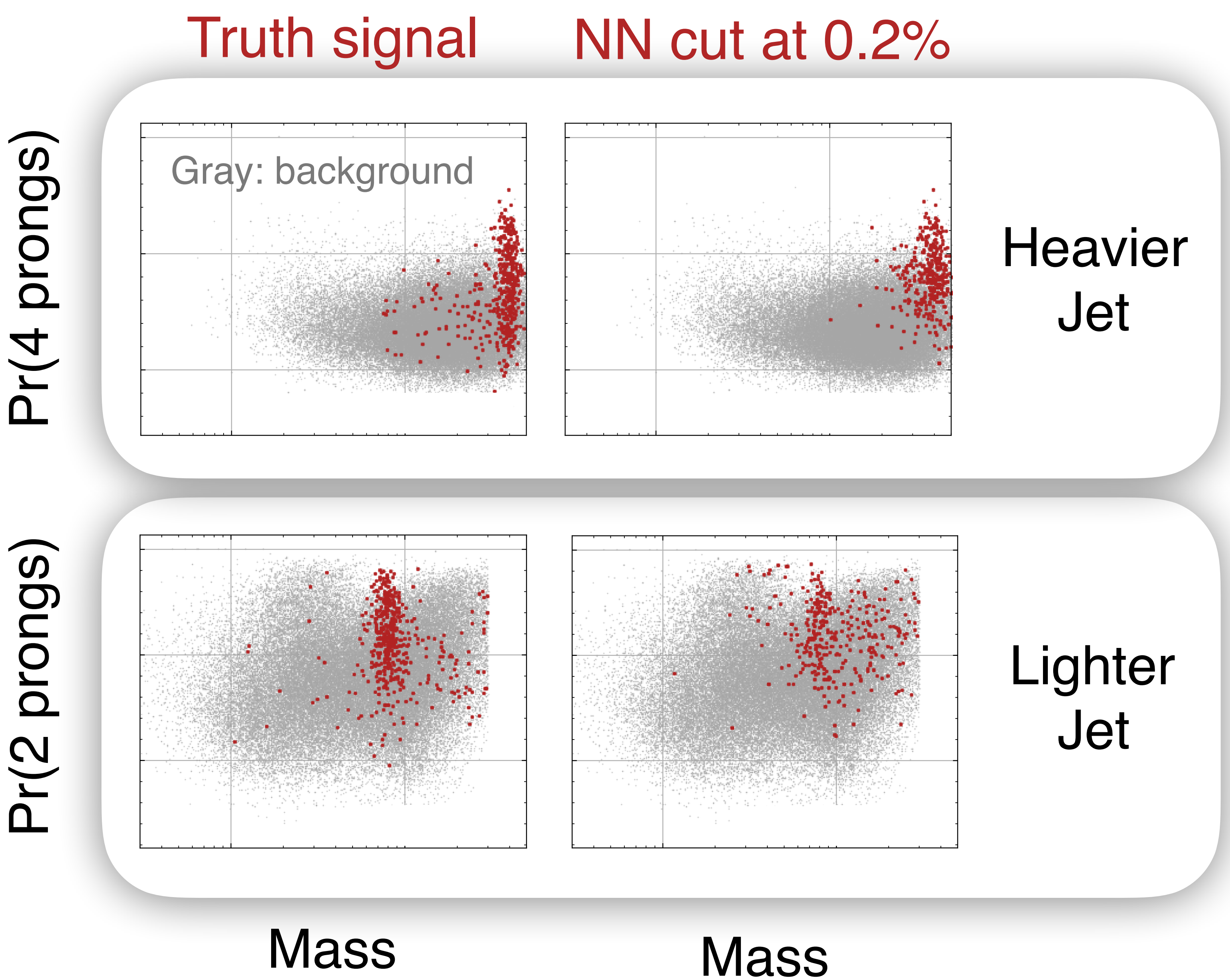
Weakly supervised learning

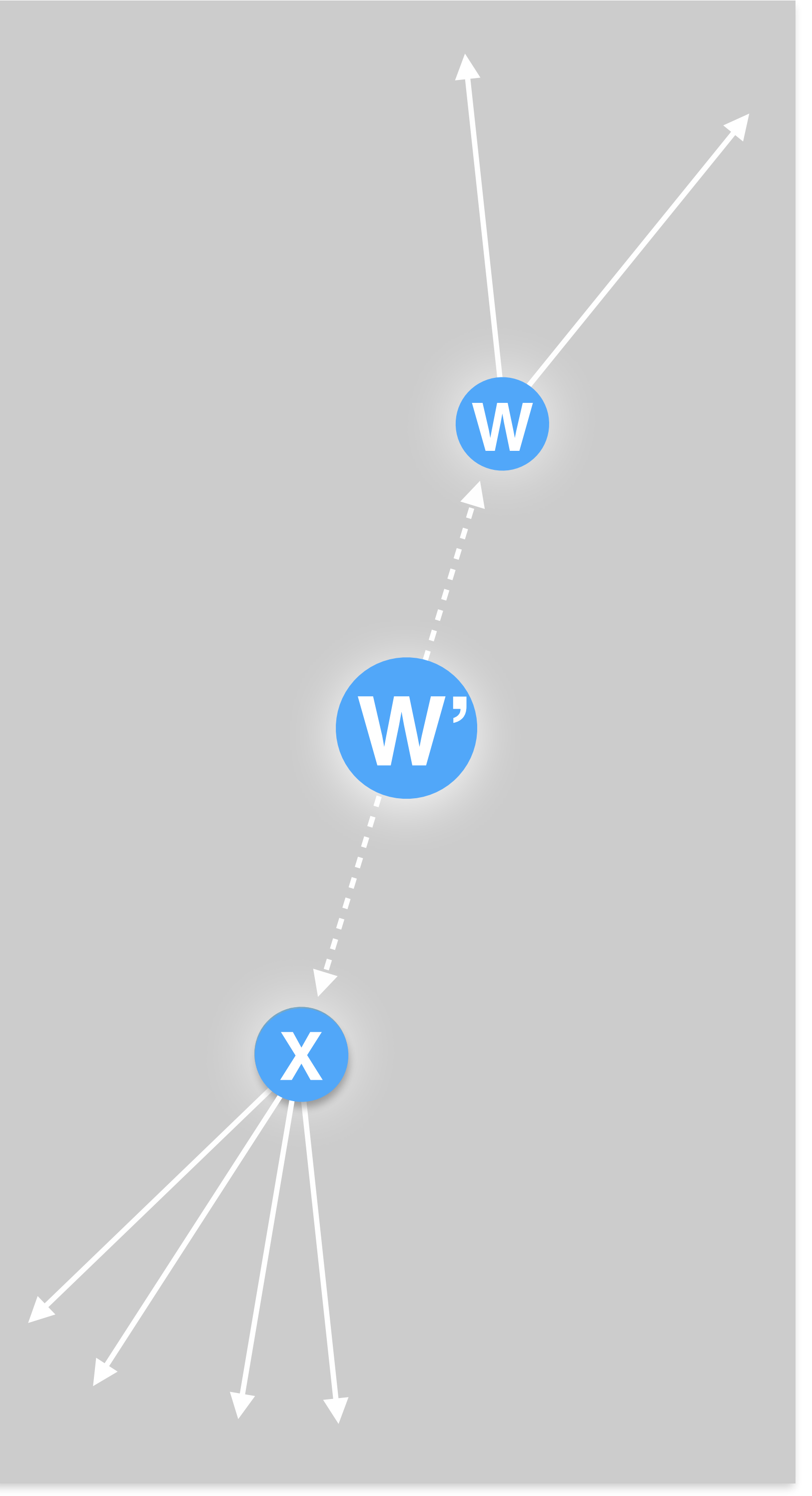
$$f_{\text{weak}} = \text{argmin}_{f' : \mathbb{R}^n \to [0,1]} \ell\left(\sum_{i=1}^{N} \frac{f'(x_i)}{N} - y\right)$$

proportions

L. Dery, **BPN**, F. Rubbo, A. Schwartzman, JHEP 05 (2017) 145

What is the network learning?

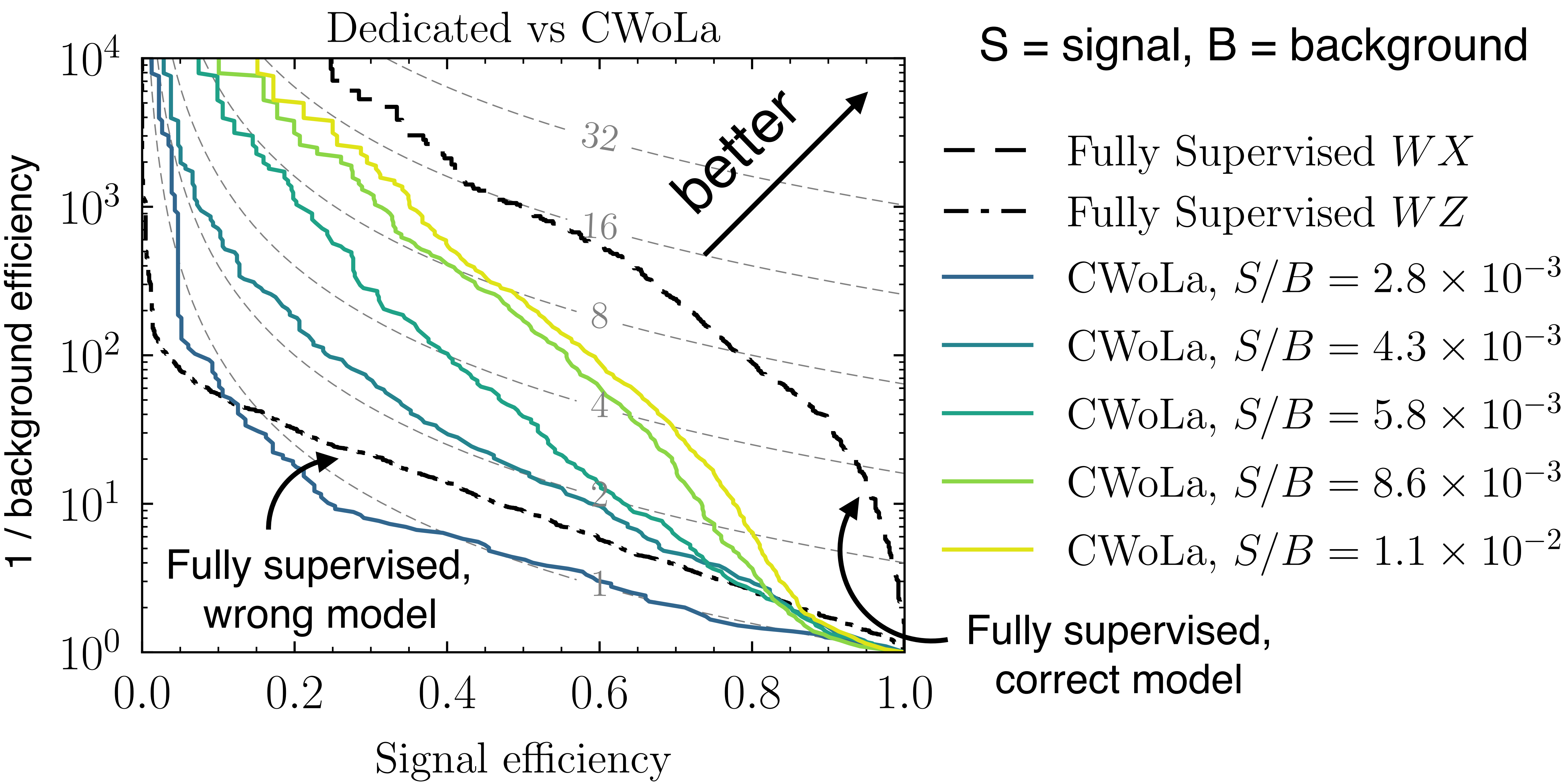




Learns to find the signal !

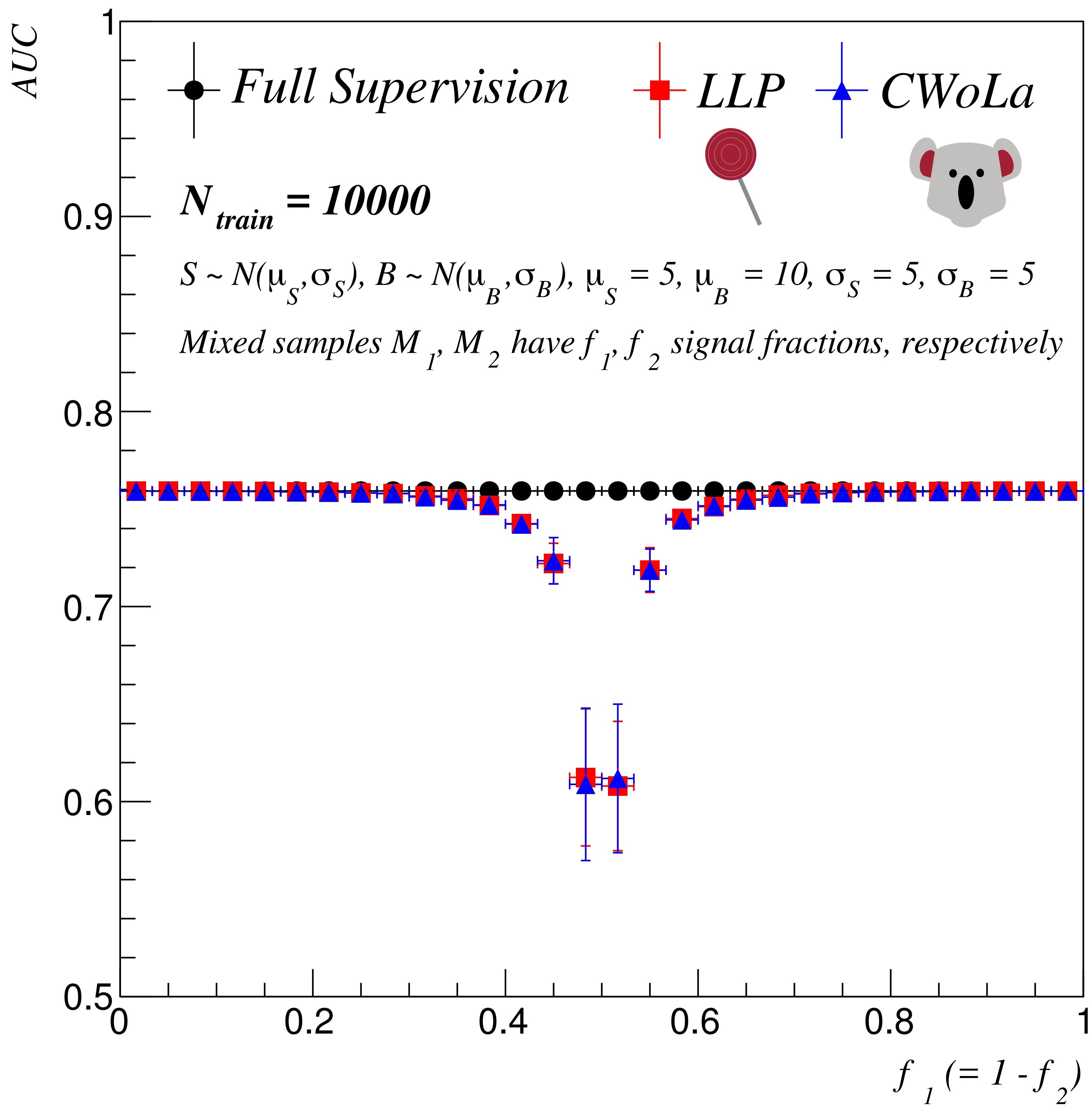
CWoLa hunting vs. Full Supervision

S = signal, B = background



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!

A note about training statistics



signal fraction of mixed sample 1

Can't learn when the two proportions are the same.

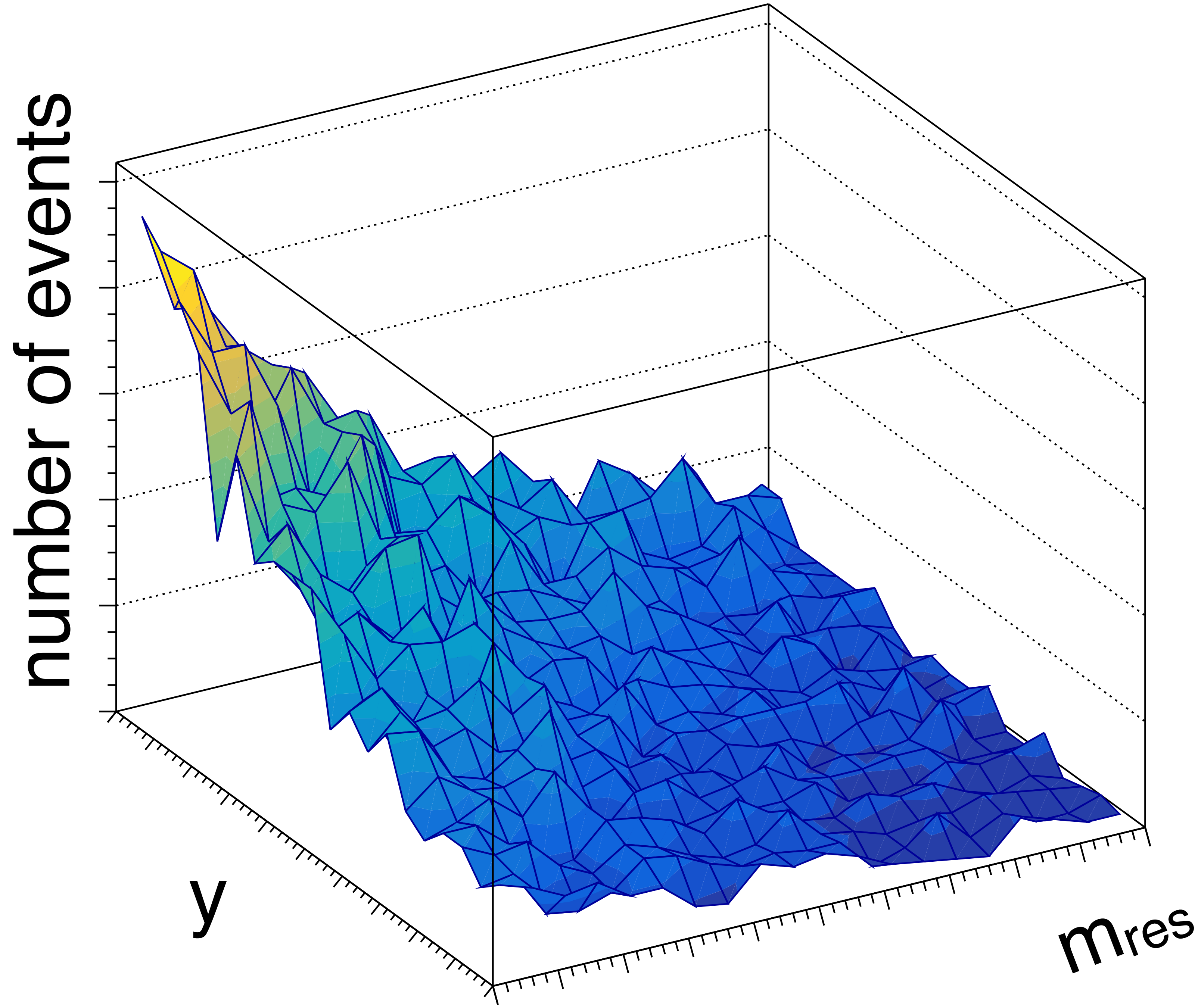
The more similar they are, the worse the performance.

(lower effective statistics)

Overtraining & Look Elsewhere Effect*

Naively, pay a huge penalty because y can be high-dimensional.

i.e. you will sculpt lots of bumps!

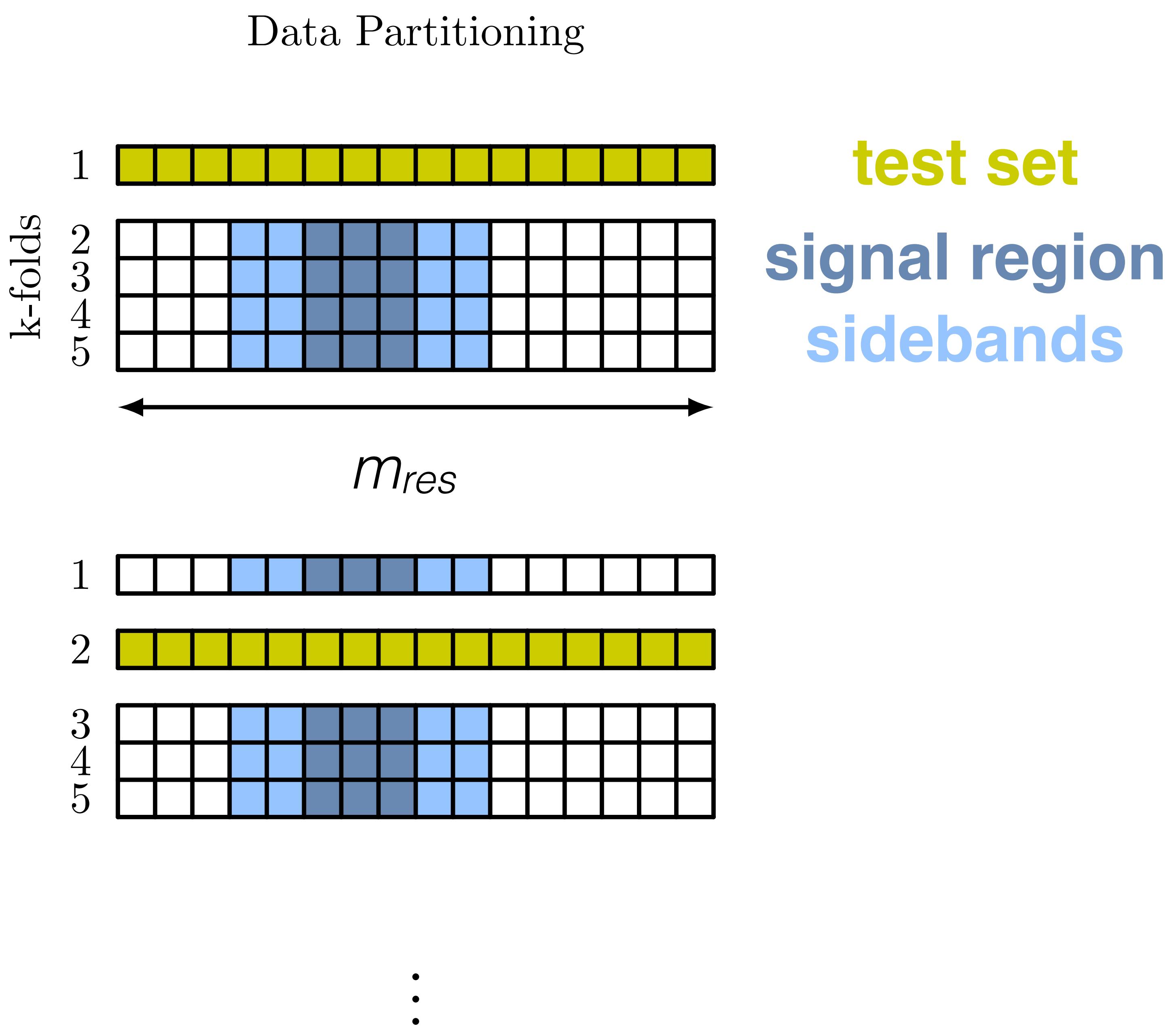


Solution: **(nested) cross-training**

*you may know this as the multiple comparisons problem

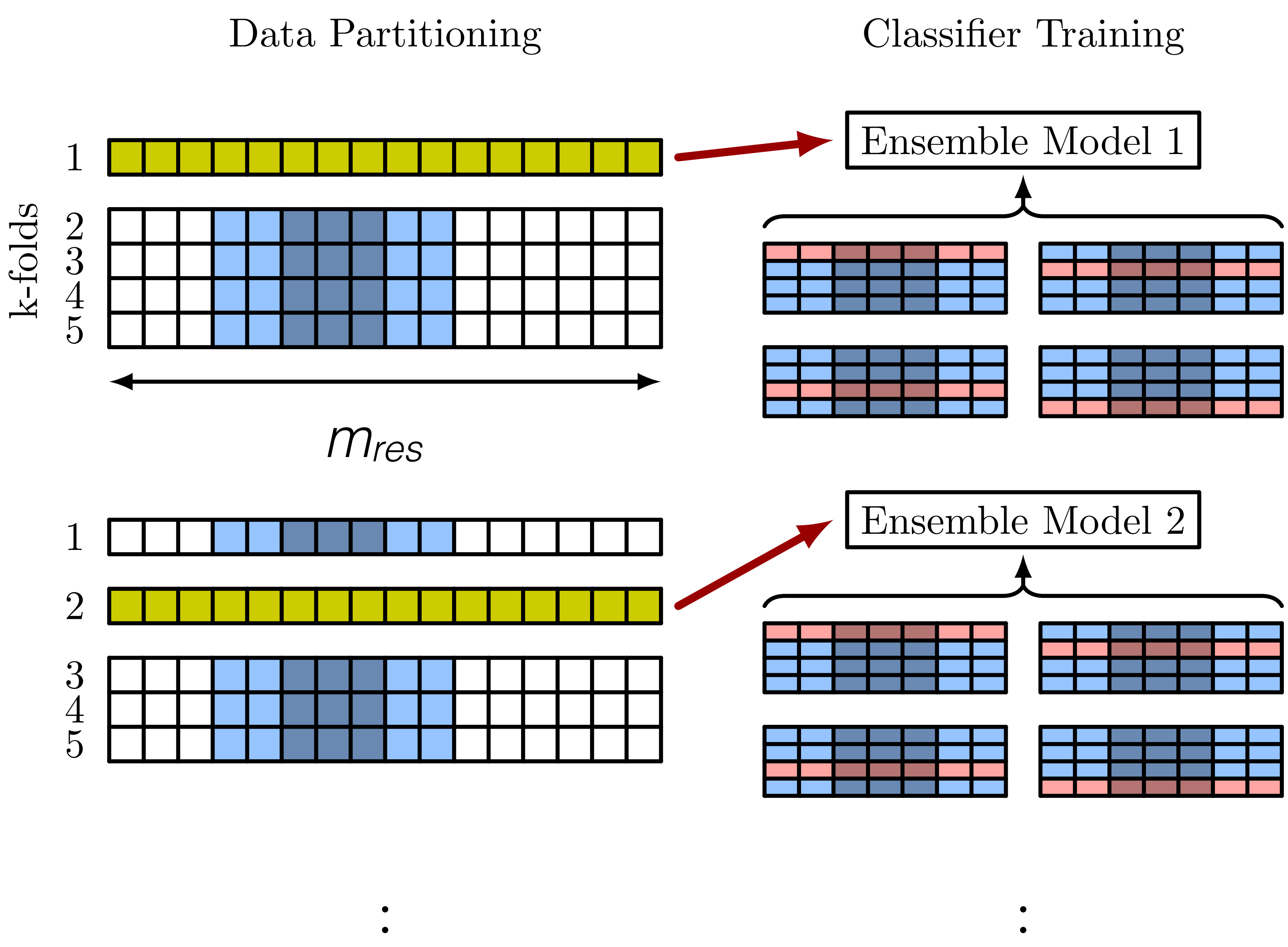
Nested cross-training

(1) Divide the entire dataset into k-folds.



Nested cross-training

(2) Train CWoLa classifiers.

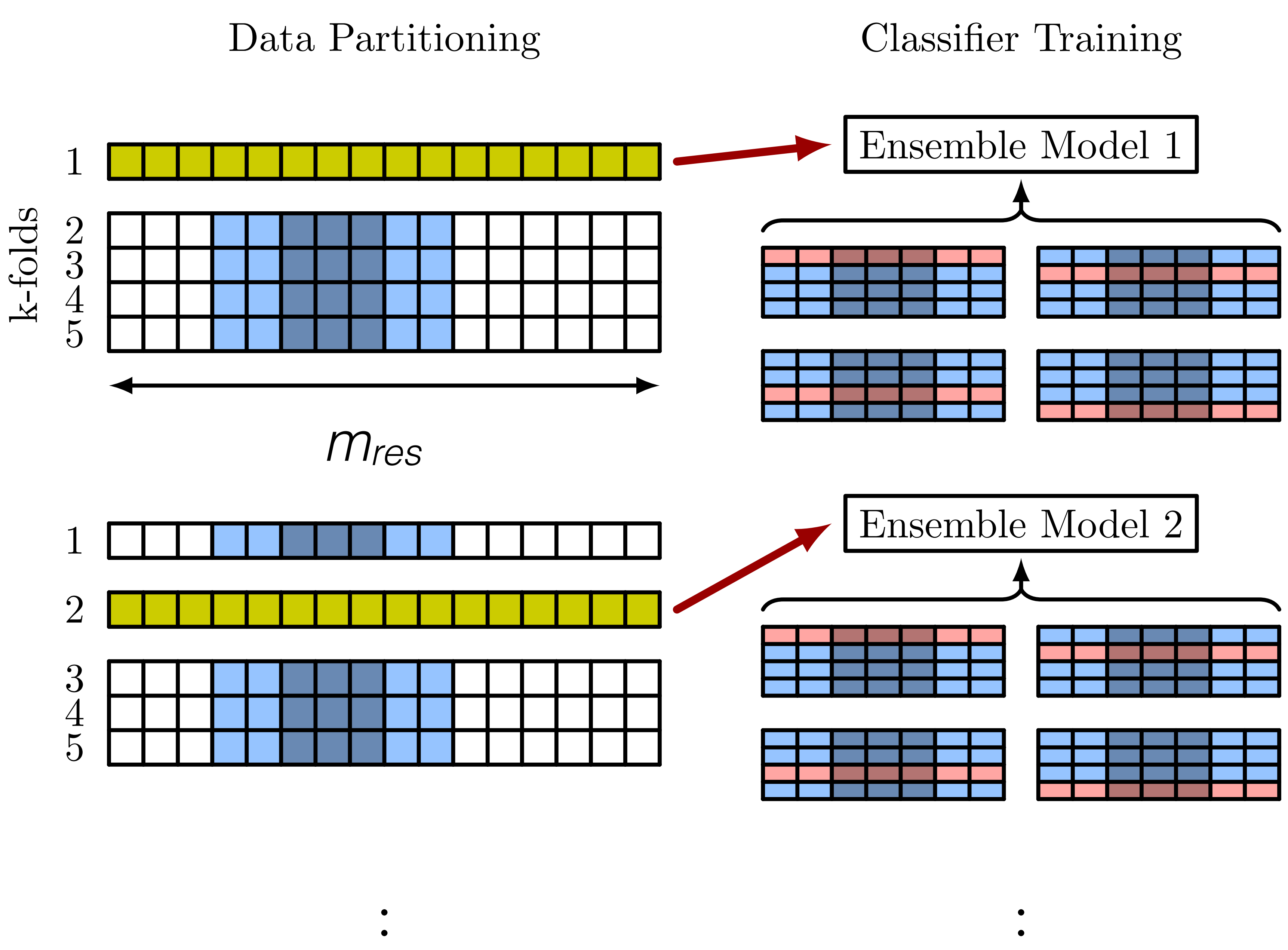


Train signal versus sideband k-1 times rotating the validation set

In practice, also train many networks with a different initialization and take the best one per k-1 rotation.

Nested cross-training

(2) Train CWoLa classifiers.

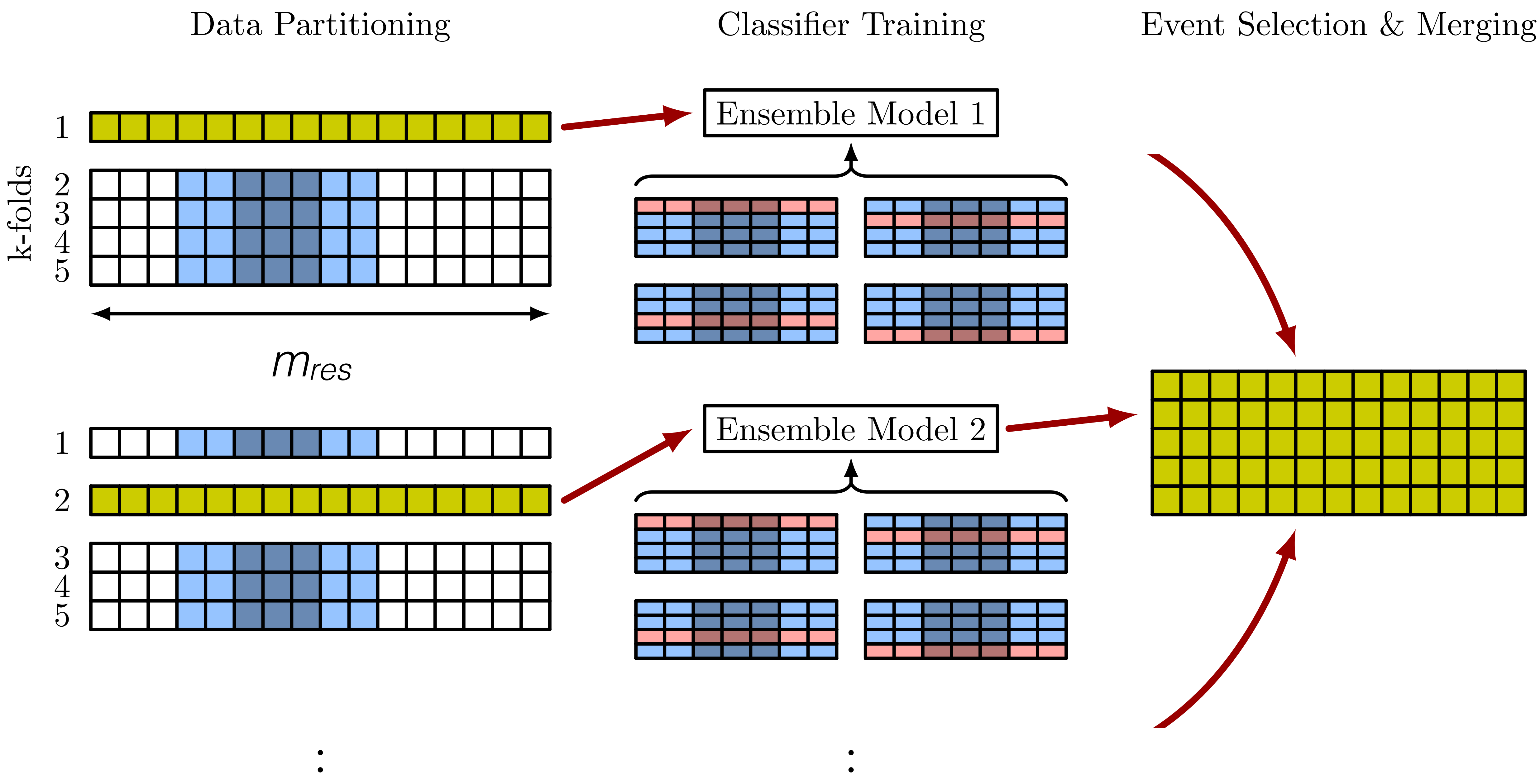


The $\boxed{\text{Ensemble Model}}$ is just the average of the four networks.

Data fluctuations will cancel destructively while signal interferes constructively.

Nested cross-training

(3) Apply classifiers to holdout test sets and sum.



SALAD background