Multidimensional measurements using multivariate techniques

The future of particle physics precision measurements?

Dag Gillberg, Carleton & Lund University

LUND UNIVERSITY

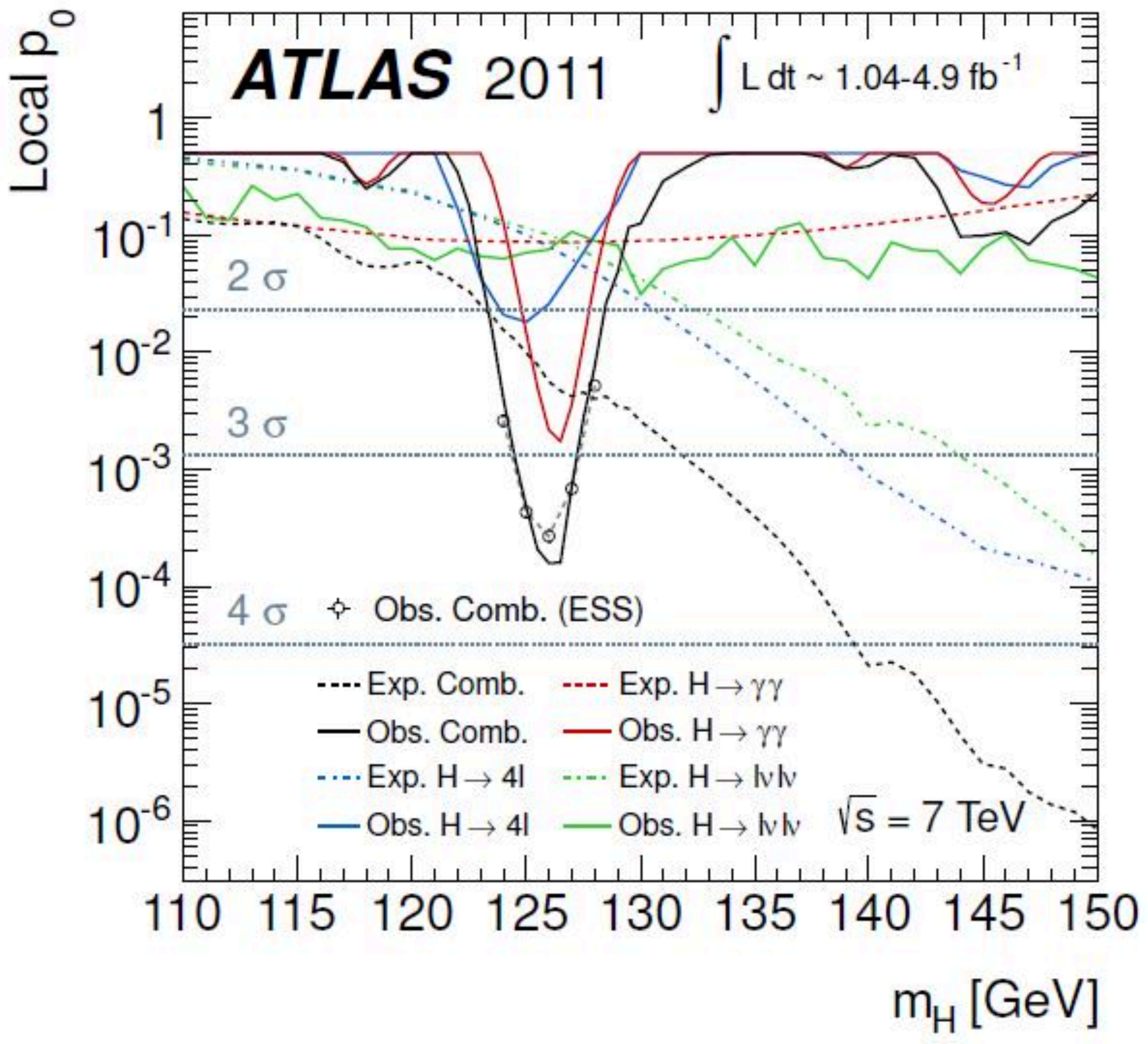
Carleton University

Outline

- Measurements in particle physics
  - Standard approach at the LHC
  - Public data, hypothesis testing
  - Limitations with current approach
- New possibilities following development in machine learning
  - Underlying mechanism
  - New opportunities
  - Challenges and open questions

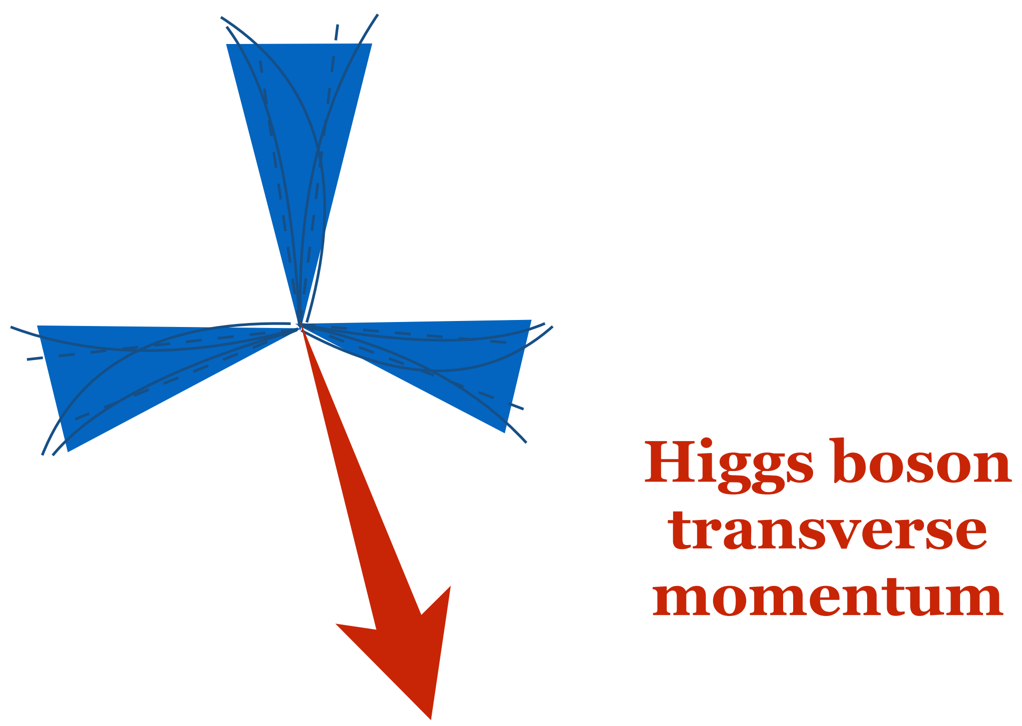
Particle physics measurements

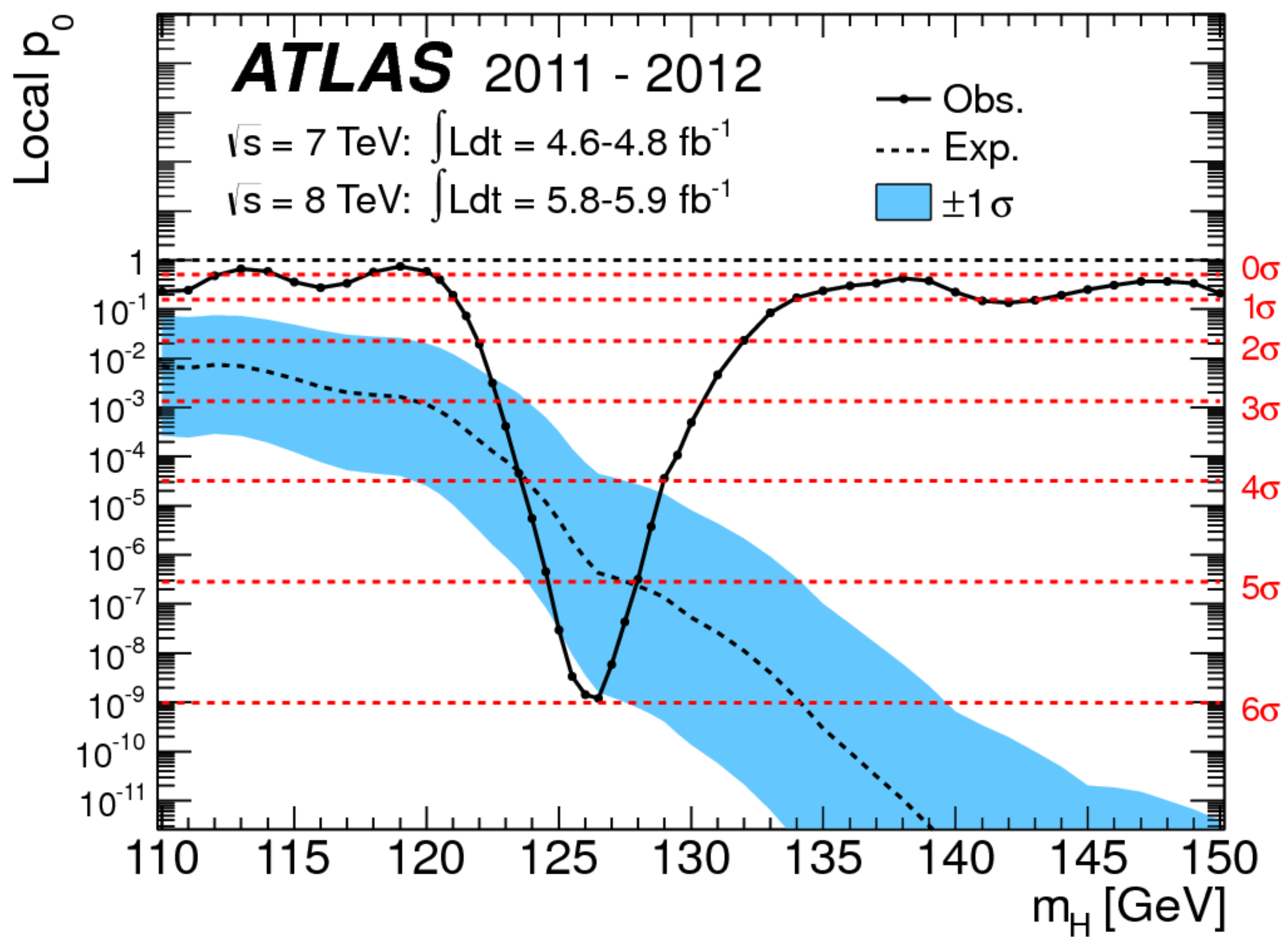
- Two main classes of experimental analyses
  - Searches
  - Measurements

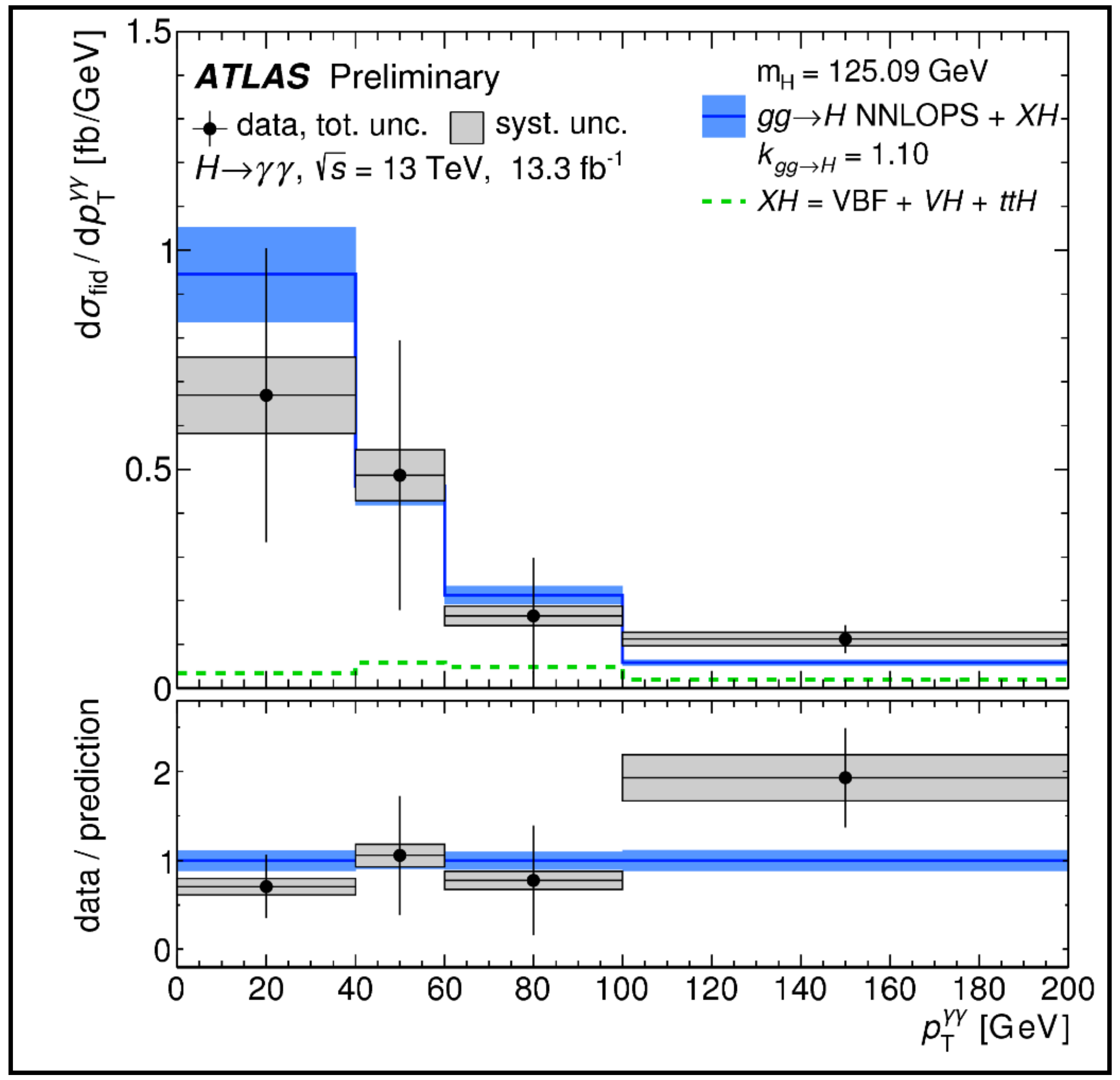


Particle physics measurements

- Two main classes of experimental analyses
  - Searches
  - Measurements



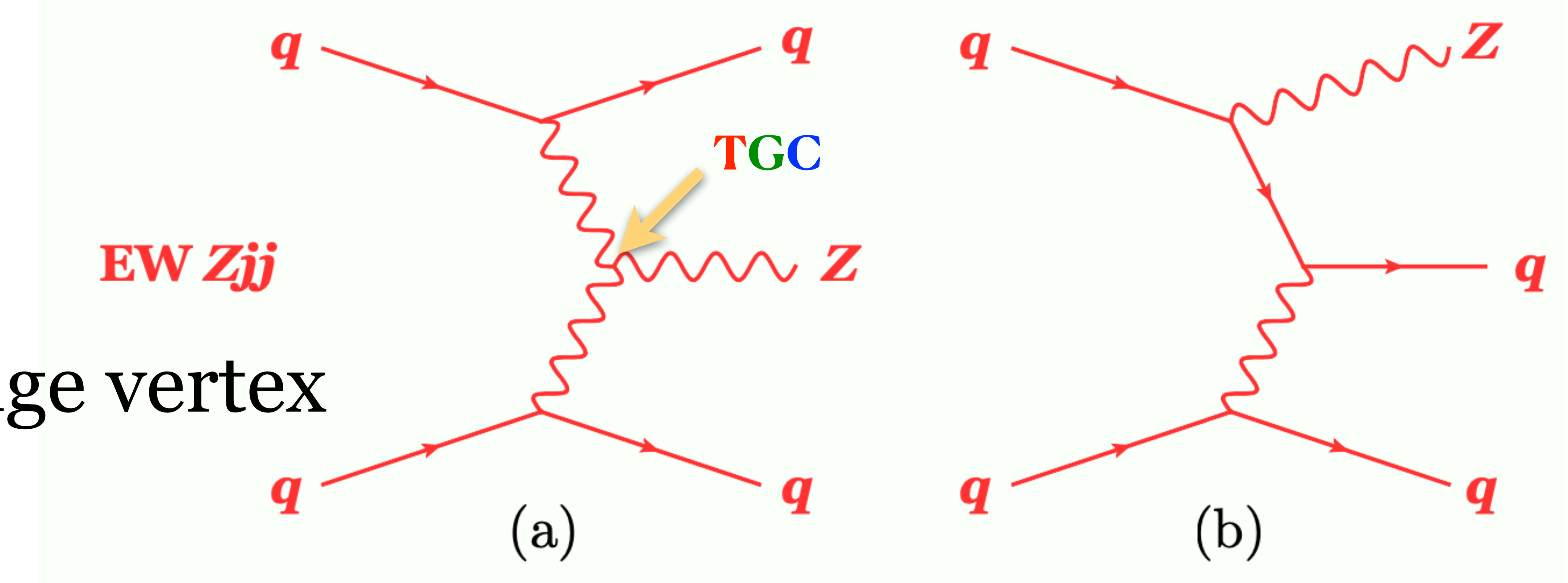


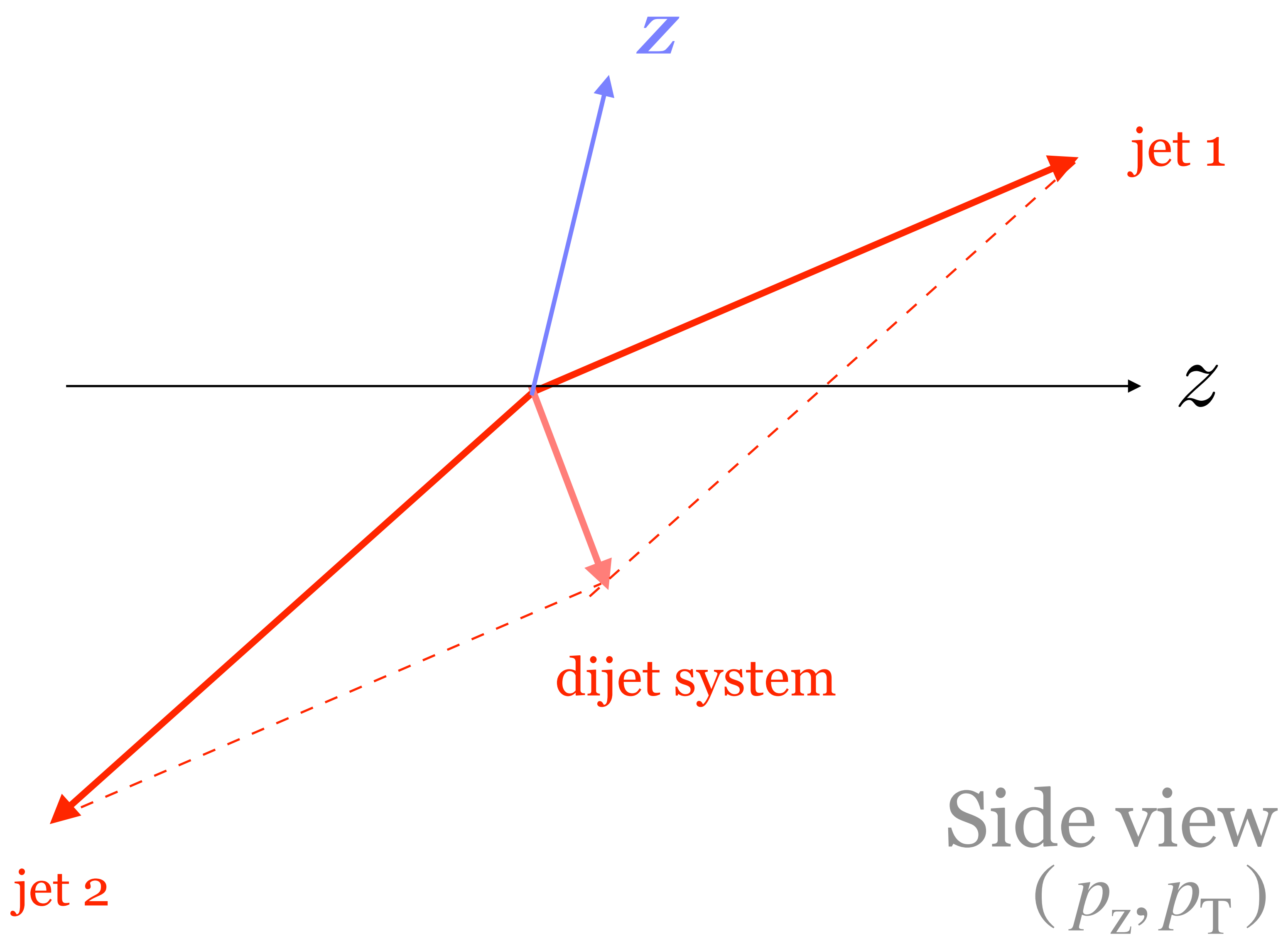


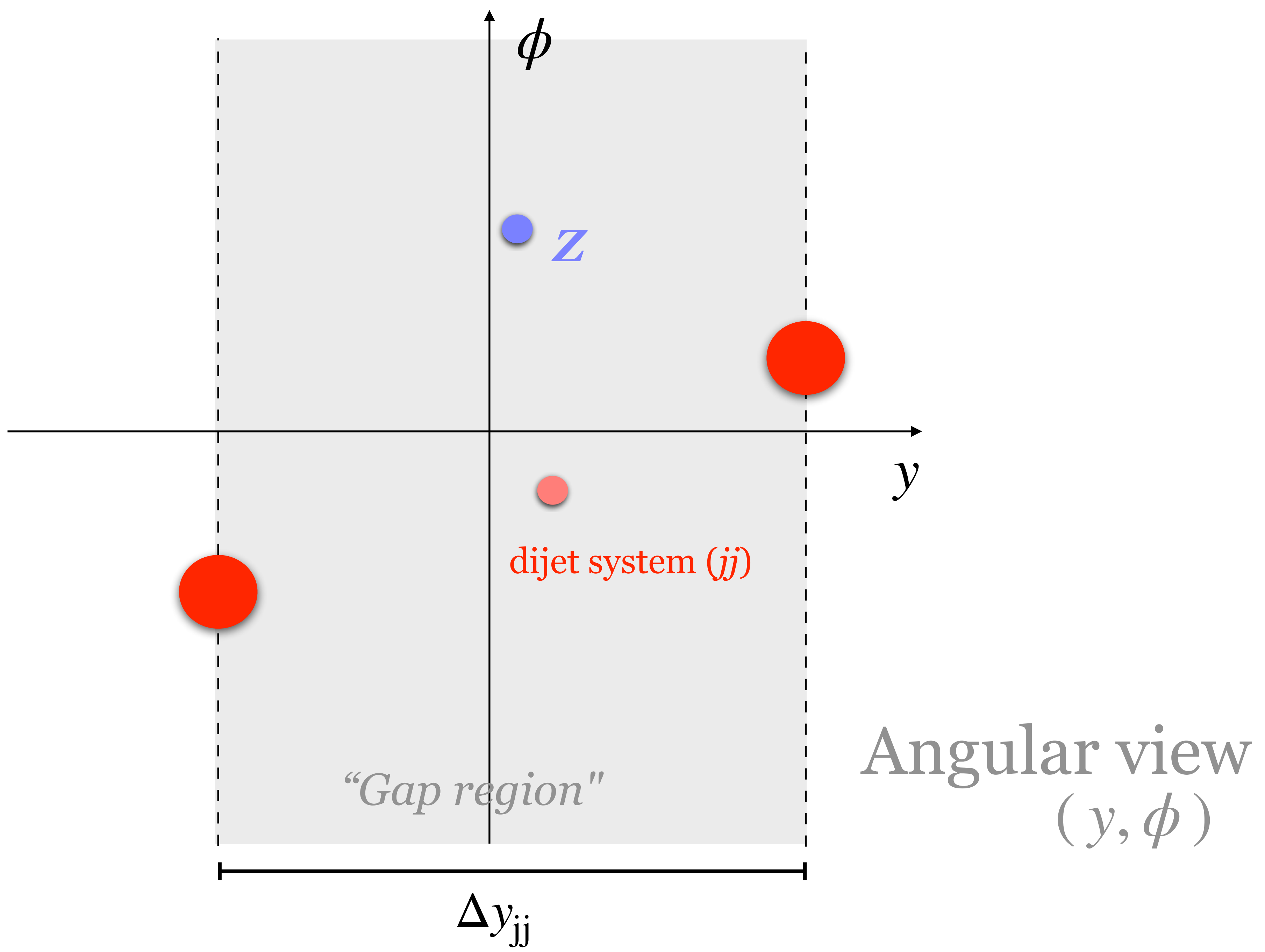
Early Higgs boson transverse momentum measurement

Example of a measurement

- Analysis close to heart
- Measurement of electroweak *Zjj* production
  - Probes gauge boson self-interaction via triple gauge vertex
  - Sensitive to CP asymmetry
- Final state: *Z* boson and two forward jets

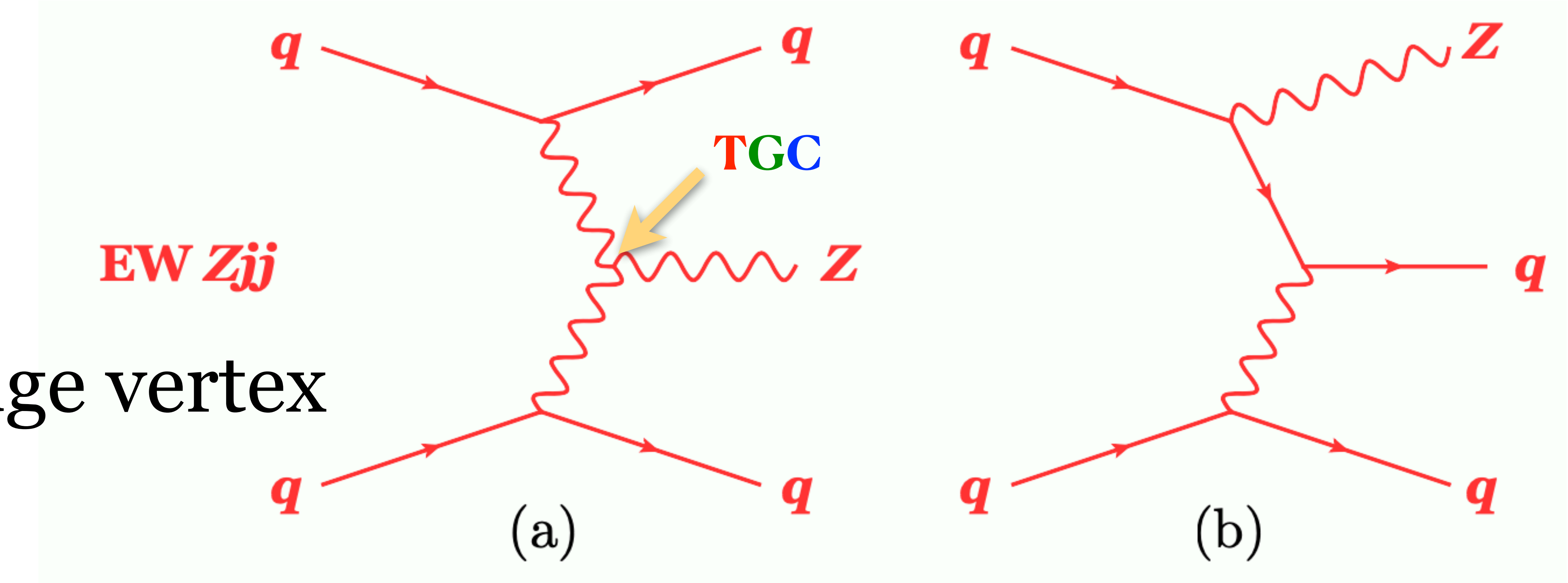


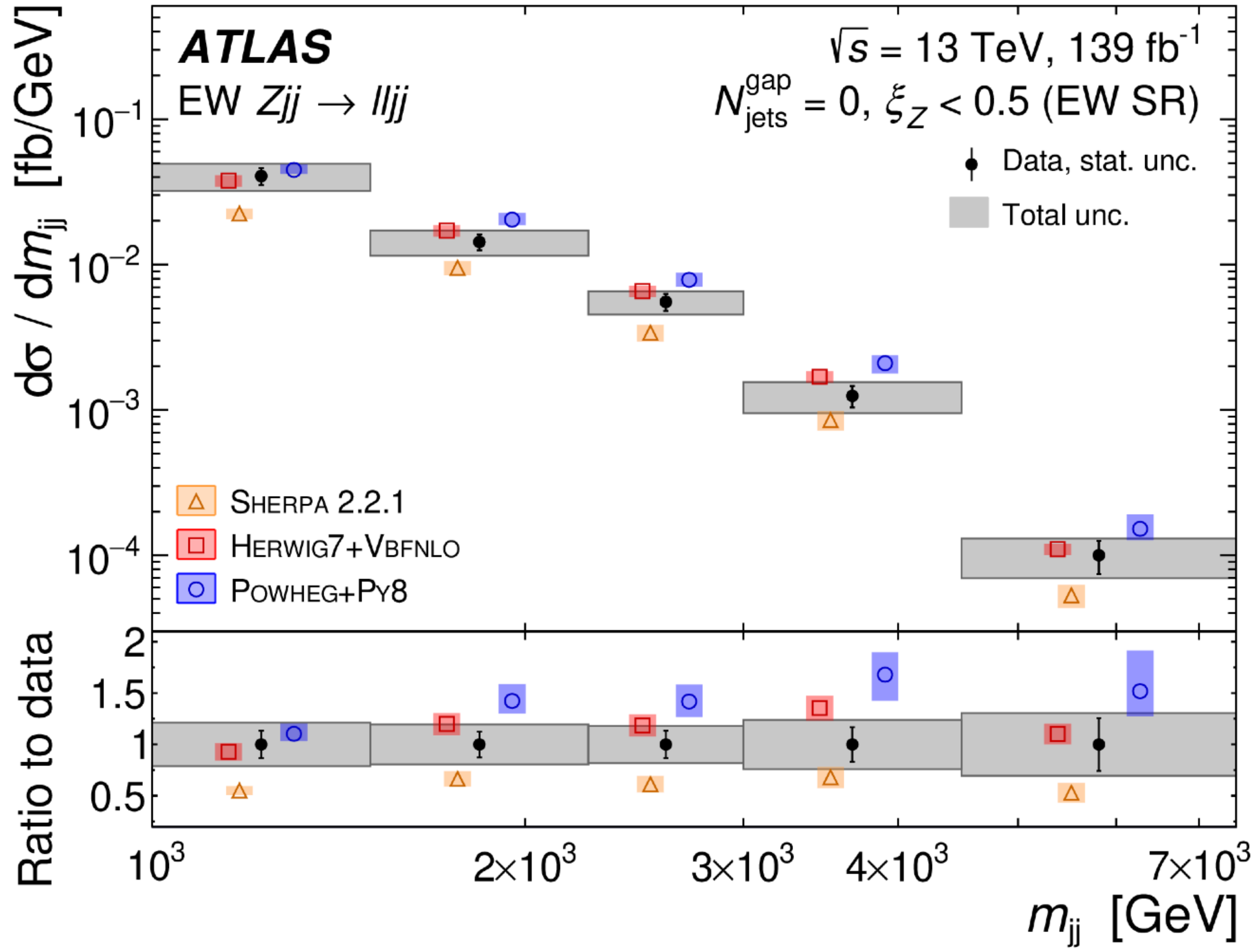


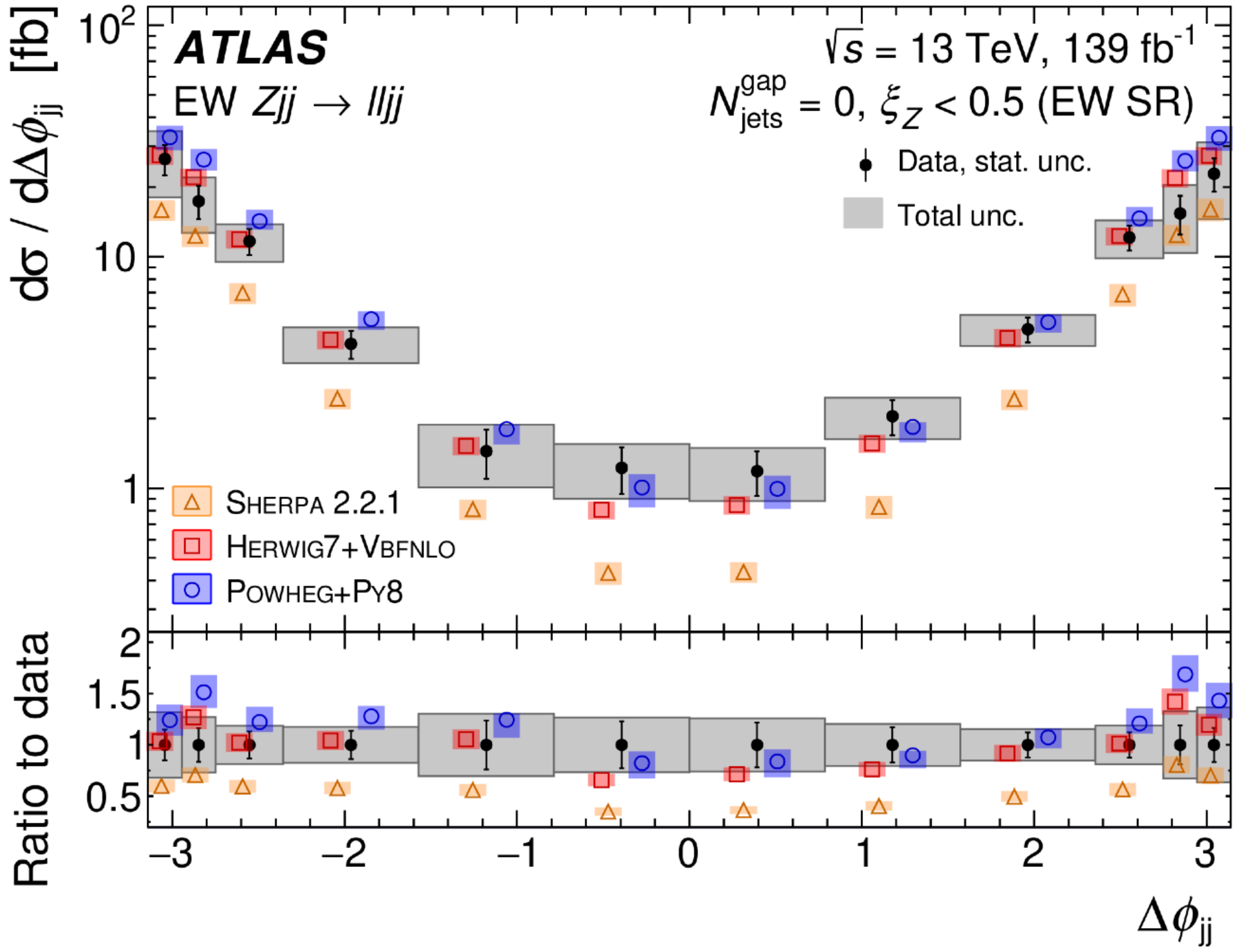


Example of a measurement

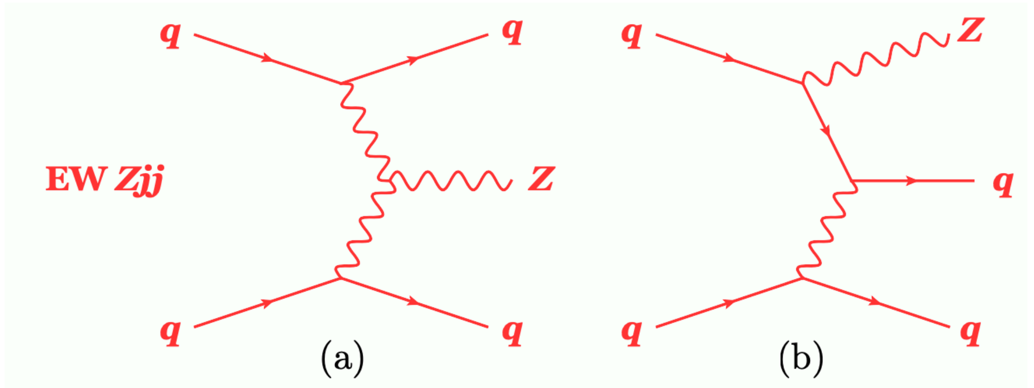
- Analysis close to heart
- Measurement of electroweak *Zjj* production
  - Probes gauge boson self-interaction via triple gauge vertex
  - Sensitive to CP asymmetry
- Final state: *Z* boson and two forward jets





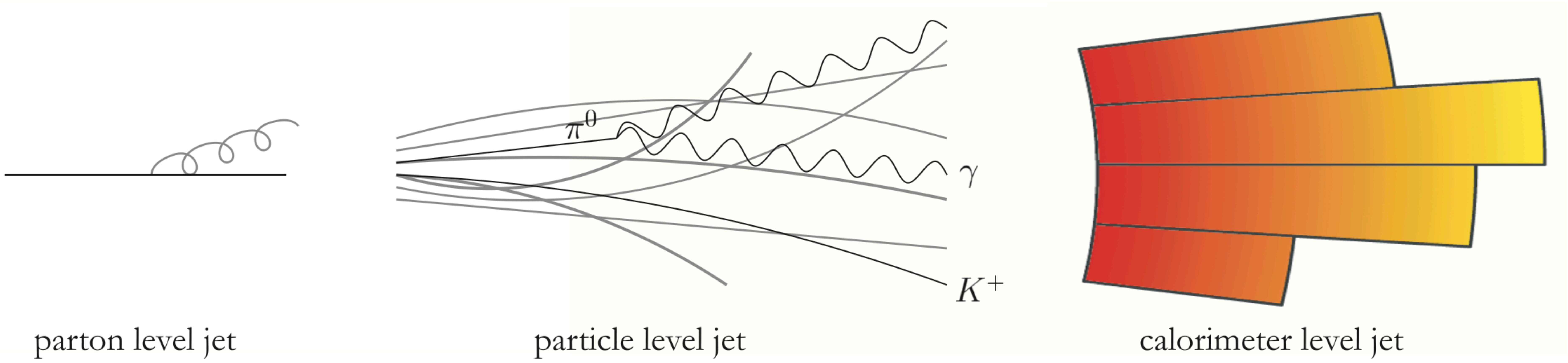


r view
y, ϕ)



Precision measurements

- For a measurement to be useful, it needs a **precise definition**
- We define measurement at the particle level
  - Real particles with life time $c\,\tau_0 > 10$ mm ($\pi^{\pm}, p, n, K, e^{-}, e^{+}$ ...)



Quarks/gluons don't exist as free particles

Cannot be observed

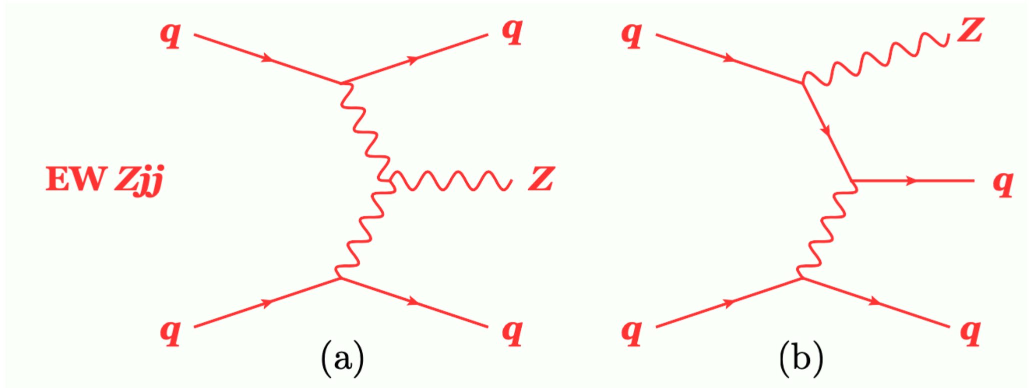
Final state! Observable in ***nature***

"What a perfect detector would see"

Reconstructed level

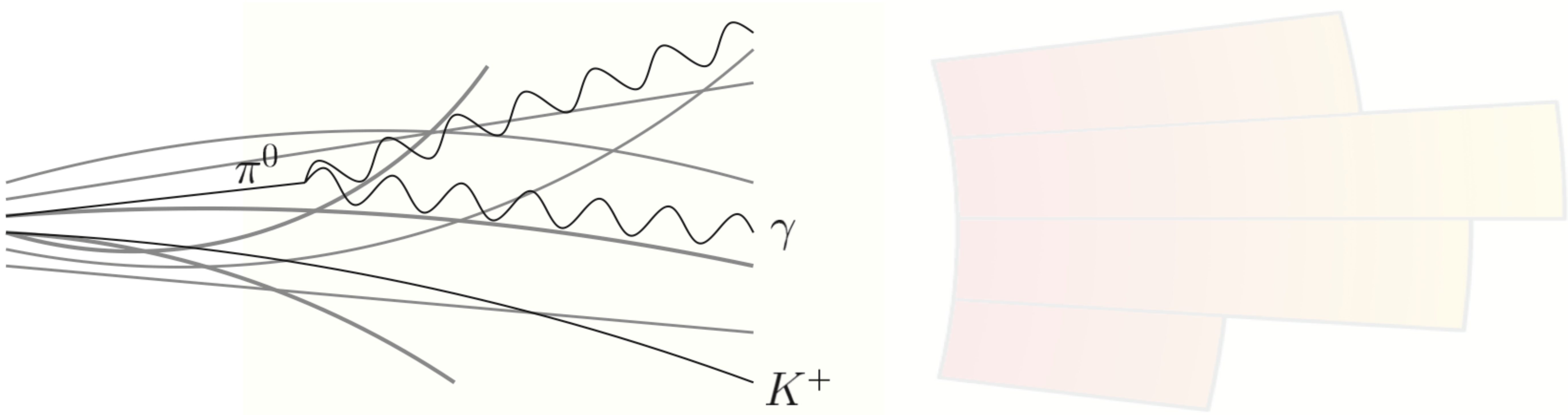
What we measure in the detector

Precision measurements



- For a measurement to be useful, it needs a **precise definition**
- We define measurement at the particle level
  - Real particles with life time $c\,\tau_0 > 10$ mm ($\pi^{\pm}, p, n, K, e^{-}, e^{+}$ ...)

Dressed muons	$p_{\mathrm{T}} > 25$ GeV and $\|\eta\| < 2.4$
Dressed electrons	$p_{\mathrm{T}} > 25$ GeV and $\|\eta\| < 2.37$ (excluding $1.37 < \|\eta\| < 1.52$)
Jets	$p_{\mathrm{T}} > 25$ GeV and $\|y\| < 4.4$
VBF topology	$N_\ell = 2$ (same flavour, opposite charge), $m_{\ell\ell} \in (81, 101)$ GeV
	$\Delta R_{\min}(\ell_1, j) > 0.4,\ \Delta R_{\min}(\ell_2, j) > 0.4$
	$N_{\mathrm{jets}} \geq 2,\ p_{\mathrm{T}}^{j1} > 85$ GeV$,\ p_{\mathrm{T}}^{j2} > 80$ GeV
	$p_{\mathrm{T},\ell\ell} > 20$ GeV$,\ p_{\mathrm{T}}^{\mathrm{bal}} < 0.15$
	$m_{jj} > 1000$ GeV$,\ \|\Delta y_{jj}\| > 2,\ \xi_Z < 1$



parton level jet

particle level jet

calorimeter level jet

Quarks/gluons don't exist as free particles

Cannot be observed

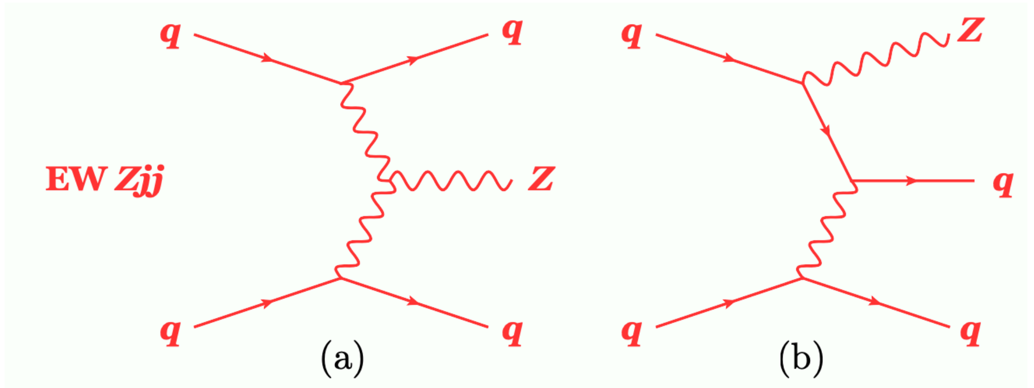
Final state!
*Observable in **nature***

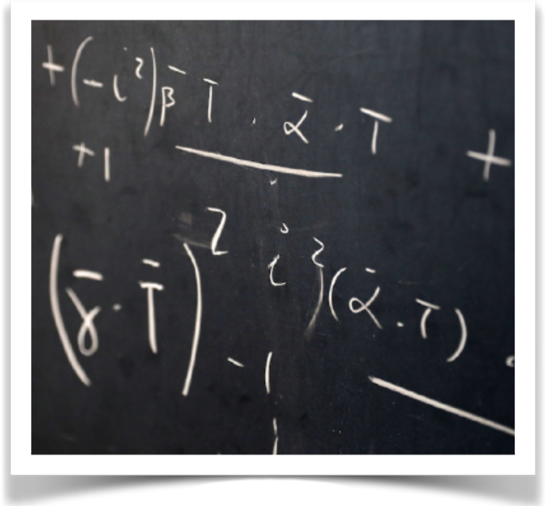
"What a perfect detector would see"

Reconstructed level

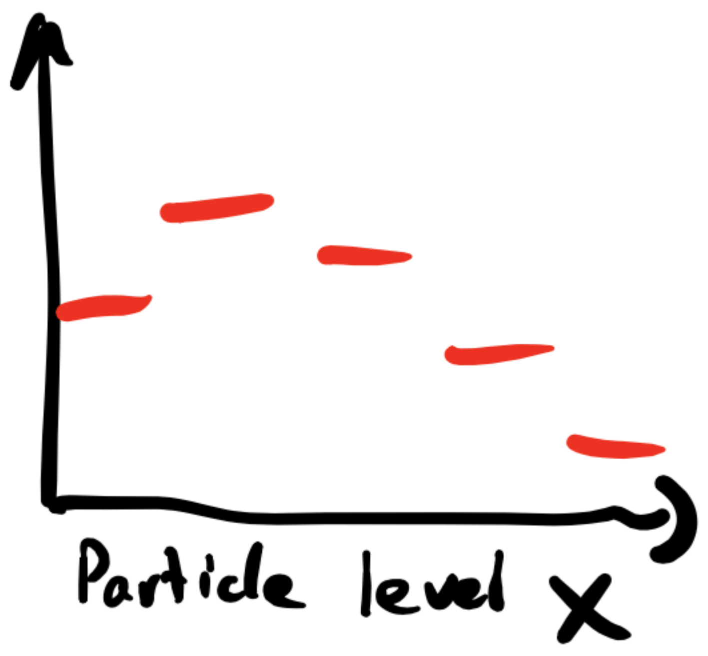
What we measure in the detector

Science at work

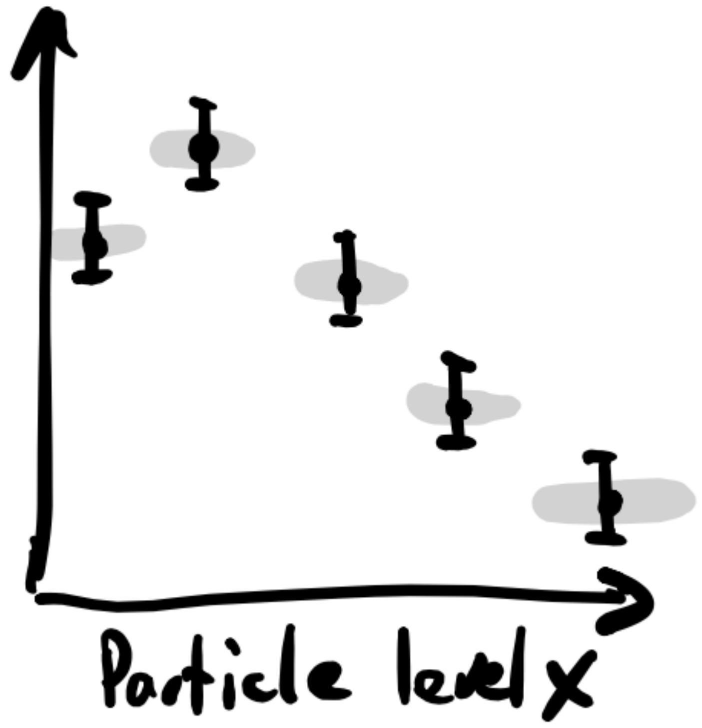


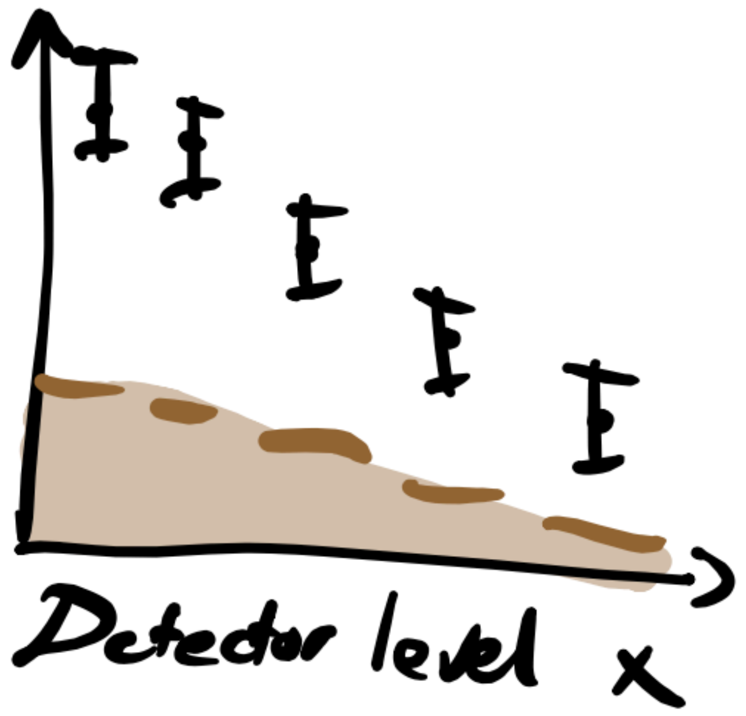


Prediction

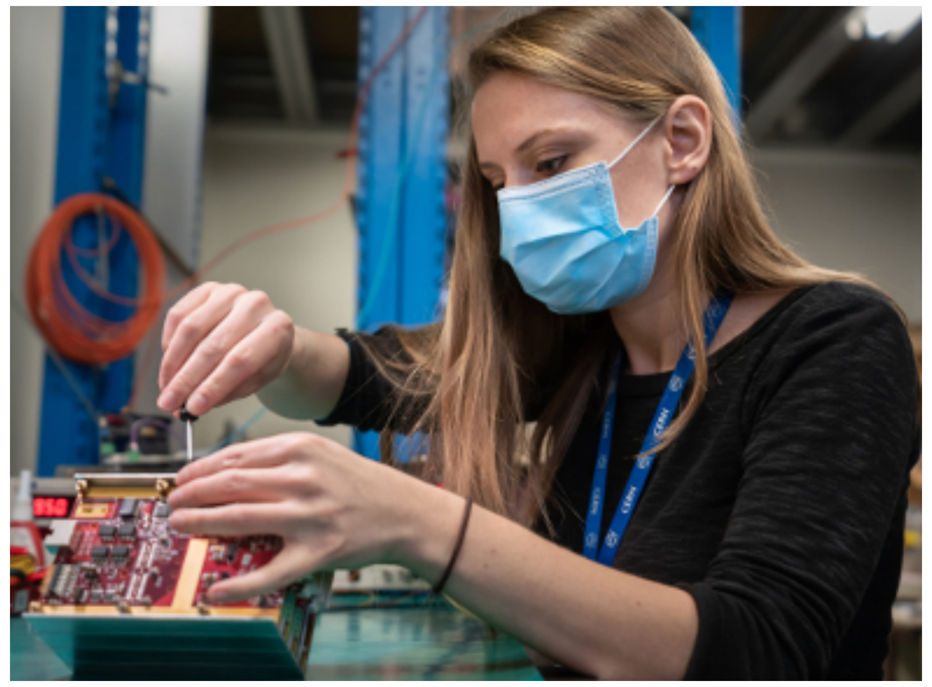


Measurement





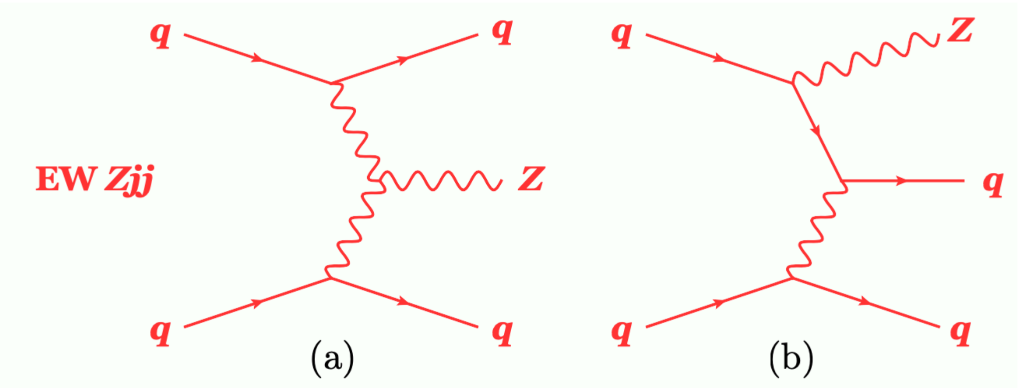
Hypothesis test!

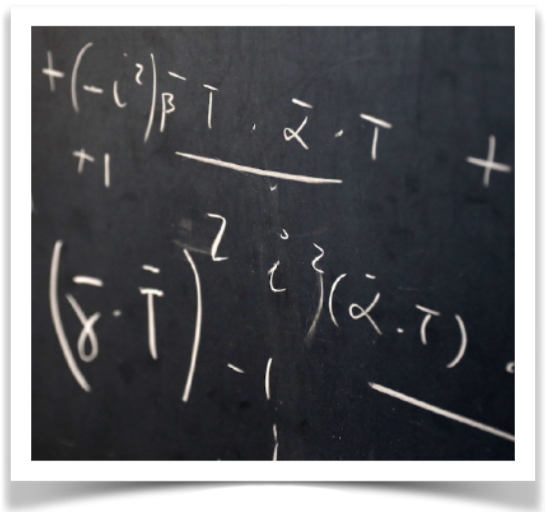


Theorists

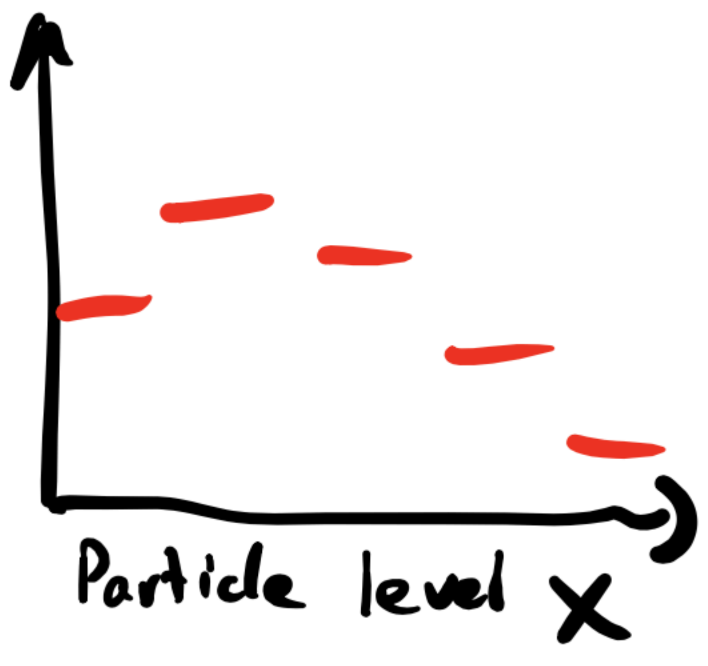
Experimentalists

Science at work

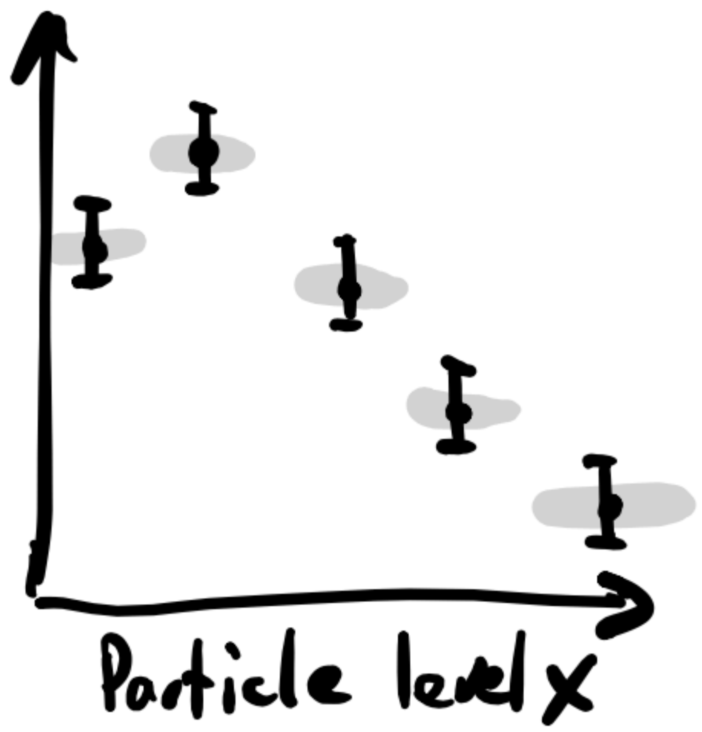


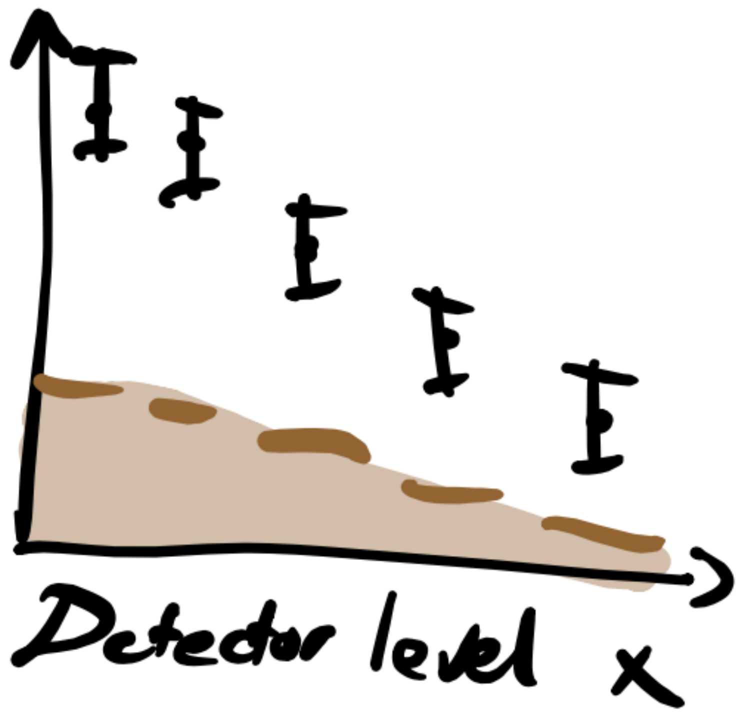


Prediction



Measurement





Example workflow

1. UFO module → MadGraph5
2. Generte events with parton shower and hadronization (e.g. MG5+Py8)
3. Feed to Rivet

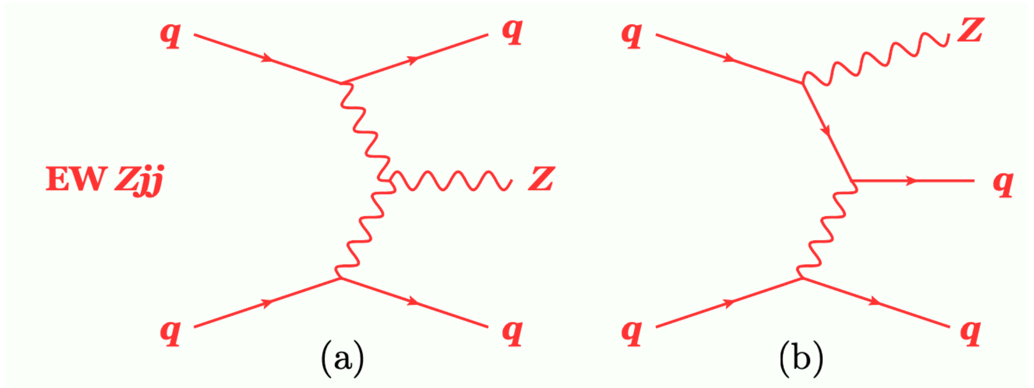
Theorists

Example workflow

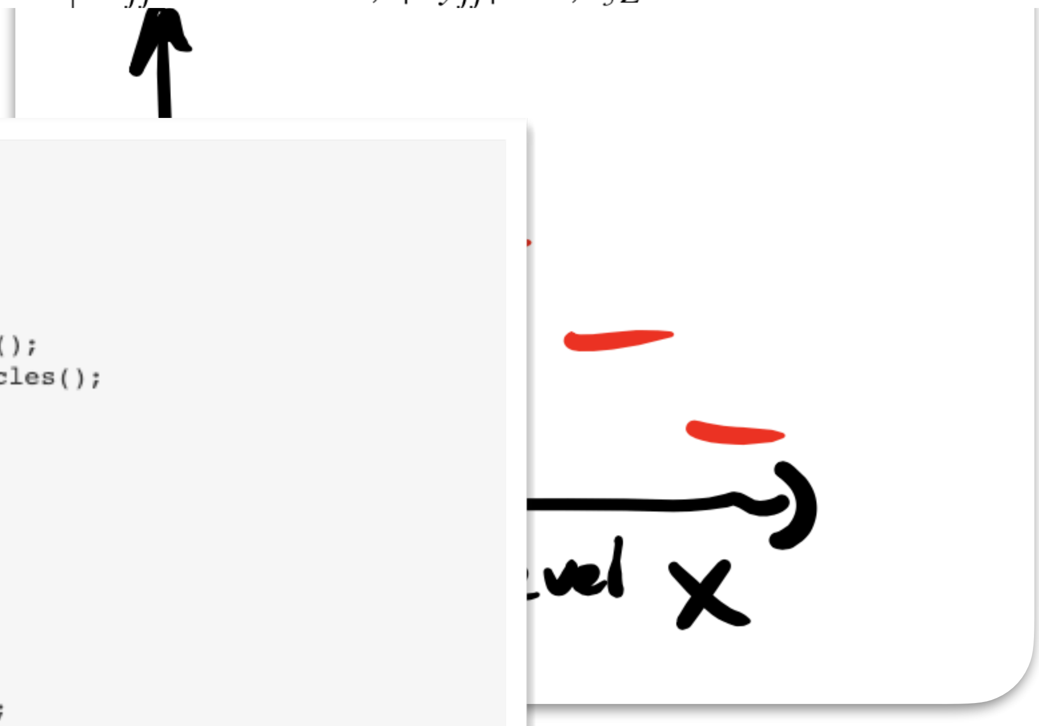
0. (Build detector, operate, calibrate)
1. Event reconstruction+analysis
2. Correct for detector effects
3. Make data public

Experimentalists

Science at work

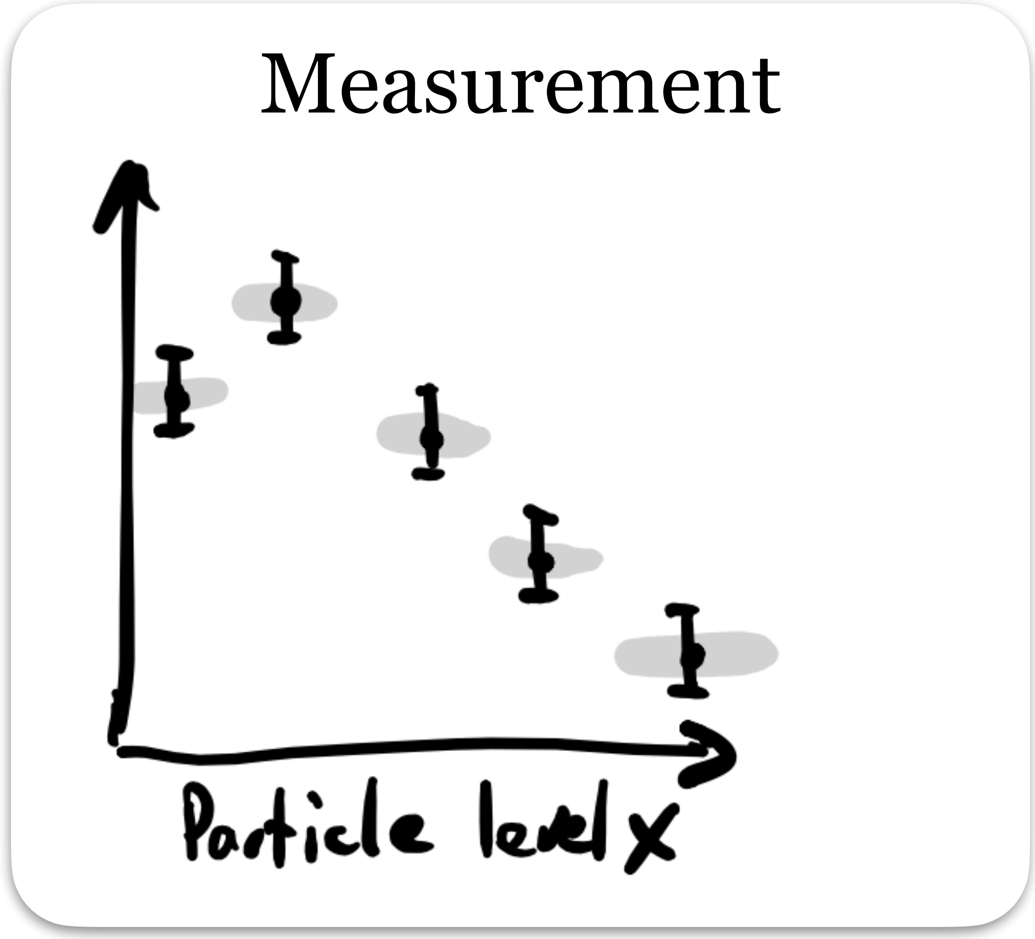


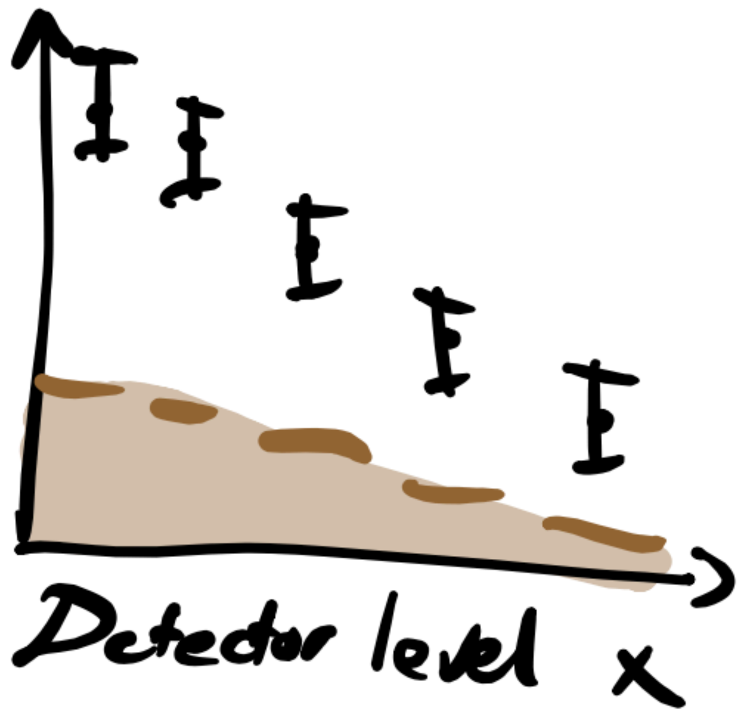
Dressed muons	$p_T > 25$ GeV and $\lvert\eta\rvert < 2.4$
Dressed electrons	$p_T > 25$ GeV and $\lvert\eta\rvert < 2.37$ (excluding $1.37 < \lvert\eta\rvert < 1.52$)
Jets	$p_T > 25$ GeV and $\lvert y\rvert < 4.4$
VBF topology	$N_\ell = 2$ (same flavour, opposite charge), $m_{\ell\ell} \in (81, 101)$ GeV
	$\Delta R_{\min}(\ell_1, j) > 0.4$, $\Delta R_{\min}(\ell_2, j) > 0.4$
	$N_{\text{jets}} \geq 2$, $p_T^{j1} > 85$ GeV, $p_T^{j2} > 80$ GeV
	$p_{T,\ell\ell} > 20$ GeV, $p_T^{\text{bal}} < 0.15$
	$m_{jj} > 1000$ GeV, $\lvert\Delta y_{jj}\rvert > 2$, $\xi_Z < 1$




```
  /// Perform the per-event analysis
  void analyze(const Event& event) {

    // Access fiducial electrons and muons
    const Particle *l1 = nullptr, *l2 = nullptr;
    Particles muons = apply<DressedLeptons>(event, "DressedMuons").particles();
    Particles elecs = apply<DressedLeptons>(event, "DressedElectrons").particles();

    // Dilepton selection 1: =2 leptons of the same kind
    if (muons.size()+elecs.size() != 2) vetoEvent;
    if      (muons.size()==2) { l1=&muons[0]; l2=&muons[1]; }
    else if (elecs.size()==2) { l1=&elecs[0]; l2=&elecs[1]; }
    else vetoEvent;

    // Dilepton selection 2: oppostie-charge and in mass range
    if ( !oppCharge(*l1, *l2) )  vetoEvent;
    if ( !inRange((l1->mom()+l2->mom()).mass()/GeV, 81.0, 101.0) ) vetoEvent;

    // Electron-jet overlap removal (note: muons are not included in jet finding)
    // make sure jets do not overlap with an electron within DR<0.2
    Jets jets;
    for (const Jet& j : apply<FastJets>(event, "Jets").jetsByPt(Cuts::pT > 25*GeV && Cuts::absrap < 4.4)) {
      if (elecs.size() == 2 && (deltaR(j, *l1, RAPIDITY) < 0.2 || deltaR(j, *l2, RAPIDITY) < 0.2 )) {
        continue;
      }
      jets += j;
    }

    // Require 2 jets with pT > 85 and 80 GeV
    if (jets.size() < 2) vetoEvent;

    // Calculate the observables
    Variables vars(jets, l1, l2);

    // make sure neither lepton overlaps with a jet within 0.4
    for (const Jet& j : jets) {
      if (deltaR(j, *l1, RAPIDITY) < 0.4 || deltaR(j, *l2, RAPIDITY) < 0.4)  vetoEvent;
    }
```

3. Feed to Rivet





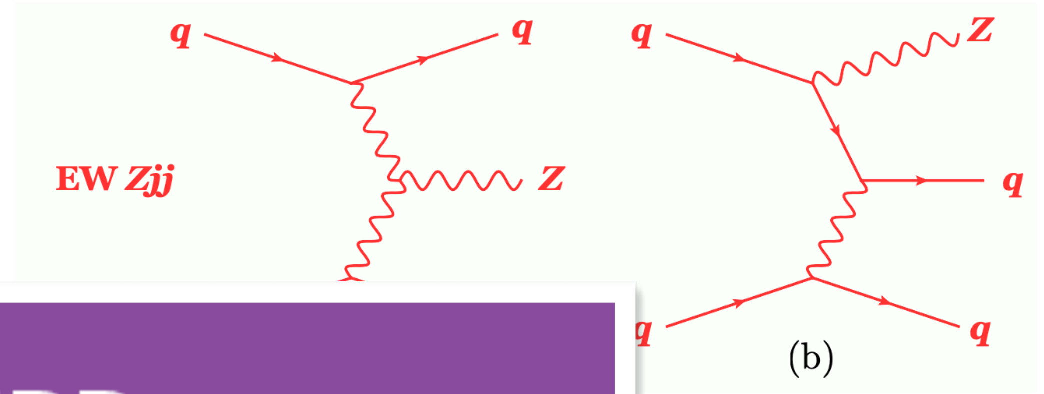
Example workflow

0. (Build detector, operate, calibrate)
1. Event reconstruction+analysis
2. Correct for detector effects
3. Make data public

Theorists

Experimentalists

Science at work



Rivet

Dressed muons	$p_{\mathrm{T}} > 25$ GeV and $\lvert\eta\rvert < 2.4$
Dressed electrons	$p_{\mathrm{T}} > 25$ GeV and $\lvert\eta\rvert < 2.37$ (excluding $1.37 < \lvert\eta\rvert < 1.52$)
Jets	$p_{\mathrm{T}} > 25$ GeV and $\lvert y\rvert < 4.4$
VBF topology	$N_\ell = 2$ (same flavour, opposite charge), $m_{\ell\ell} \in (81, 101)$ GeV
	$\Delta R_{\mathrm{min}}(\ell_1, j) > 0.4$, $\Delta R_{\mathrm{min}}(\ell_2, j) > 0.4$
	$N_{\mathrm{jets}} \geq 2$, $p_{\mathrm{T}}^{j1} > 85$ GeV, $p_{\mathrm{T}}^{j2} > 80$ GeV
	$p_{\mathrm{T},\ell\ell} > 20$ GeV, $p_{\mathrm{T}}^{\mathrm{bal}} < 0.15$
	$m_{jj} > 1000$ GeV, $\lvert\Delta y_{jj}\rvert > 2$, $\xi_Z < 1$

```
  /// Perform the per-event analysis
  void analyze(const Event& event) {

    // Access fiducial electrons and muons
    const Particle *l1 = nullptr, *l2 = nullptr;
    Particles muons = apply<DressedLeptons>(event, "DressedMuons").particles();
    Particles elecs = apply<DressedLeptons>(event, "DressedElectrons").particles();

    // Dilepton selection 1: =2 leptons of the same kind
    if (muons.size()+elecs.size() != 2) vetoEvent;
    if      (muons.size()==2) { l1=&muons[0]; l2=&muons[1]; }
    else if (elecs.size()==2) { l1=&elecs[0]; l2=&elecs[1]; }
    else vetoEvent;

    // Dilepton selection 2: oppostie-charge and in mass range
    if ( !oppCharge(*l1, *l2) )  vetoEvent;
    if ( !inRange((l1->mom()+l2->mom()).mass()/GeV, 81.0, 101.0) ) vetoEvent;

    // Electron-jet overlap removal (note: muons are not included in jet finding)
    // make sure jets do not overlap with an electron within DR<0.2
    Jets jets;
    for (const Jet& j : apply<FastJets>(event, "Jets").jetsByPt(Cuts::pT > 25*GeV && Cuts::absrap < 4.4)) {
      if (elecs.size() == 2 && (deltaR(j, *l1, RAPIDITY) < 0.2 || deltaR(j, *l2, RAPIDITY) < 0.2 )) {
        continue;
      }
      jets += j;
    }

    // Require 2 jets with pT > 85 and 80 GeV
    if (jets.size() < 2) vetoEvent;

    // Calculate the observables
    Variables vars(jets, l1, l2);

    // make sure neither lepton overlaps with a jet within 0.4
    for (const Jet& j : jets) {
      if (deltaR(j, *l1, RAPIDITY) < 0.4 || deltaR(j, *l2, RAPIDITY) < 0.4)  vetoEvent;
    }
```

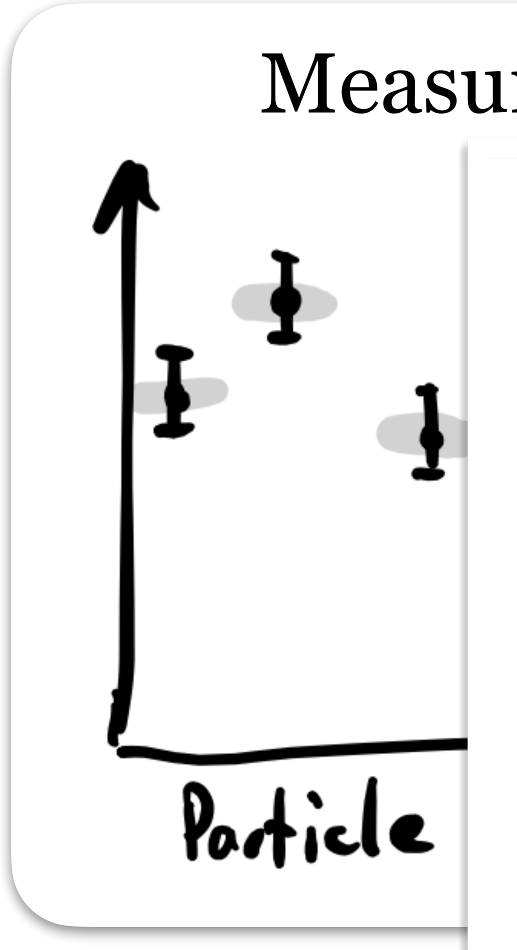
vel x

r

)

3. Feed to Rivet

Theorists



HEPData

Repository for publication-related High-Energy Physics data

Distribution	Data	Powheg + Py8	Herwig7 + VBFNLO
SQRT(S)	13000 GeV		
LUMINOSITY	139 fb^{-1}		
m_{jj} [GeV]	Differential cross-section [fb/GeV]		
1000 - 1500	0.040673 ±0.00536 stat ∓0.00044 JES_EtaIntercalibration_Modelling ∓0.000691 JES_EffectiveNP_Modelling1 + 32 more errors Show all	0.044867 +0.00404 -0.00278	0.03775
1500 - 2250	0.014316 ±0.00179 stat ∓0.00021 JES_EtaIntercalibration_Modelling ∓0.000232 JES_EffectiveNP_Modelling1 + 32 more errors Show all	0.020374 +0.00234 -0.00173	(

∓0.000295 JER_EffectiveNP_4
∓4.79e-05 JER_EffectiveNP_5
±7.6e-05 JER_EffectiveNP_6
∓0.000115 JER_EffectiveNP_7
∓0.000276 JER_EffectiveNP_8
∓0.000641 JER_EffectiveNP_9
∓0.000128 JER_EffectiveNP_10
∓0.000234 JER_EffectiveNP_11
∓0.000125 JER_EffectiveNP_12restTerm
±0.000778 JER_DataVsMC
±4.18e-05 MUON_SAGITTA_RHO
∓0.00027 ELECTRON_ID
∓0.000233 MUON_EFF
±1.76e-05 pileup_model
±0.00463 strongZjj_gen_choice
∓0.000575 strongZjj_pdf
±0.00277 strongZjj_qcd
±0.00137 ewStrong_interference
∓2.33e-06 ewZjj_pdf
±0.00105 ewZjj_qcd
∓0.000924 unf_MCgen
±0.000187 unf_DataRew
±0.000701 Lumi

x

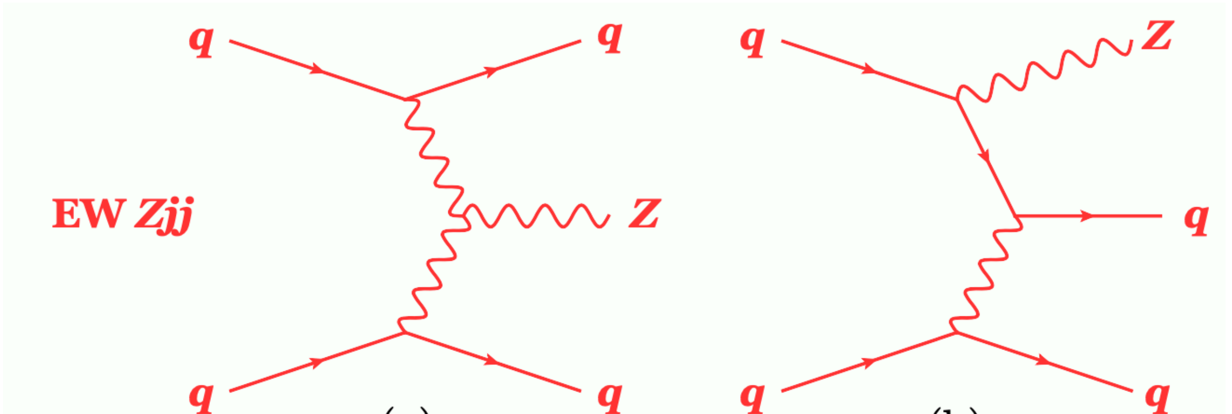
is

2. Correct for detect

3. Make data public

Experiment

Useful tools at hand



(a) (b)

- HepData stores the measurements with associated uncertainties
  - hepdata.net
- Rivet is synchronized with the HepData entry
  - Ensures predictions defined in accordance with the data
- Fast and effective

TWiki.org

Differential cross-section measurements for the electroweak production of dijets in association with a Z boson in proton-proton collisions at ATLAS

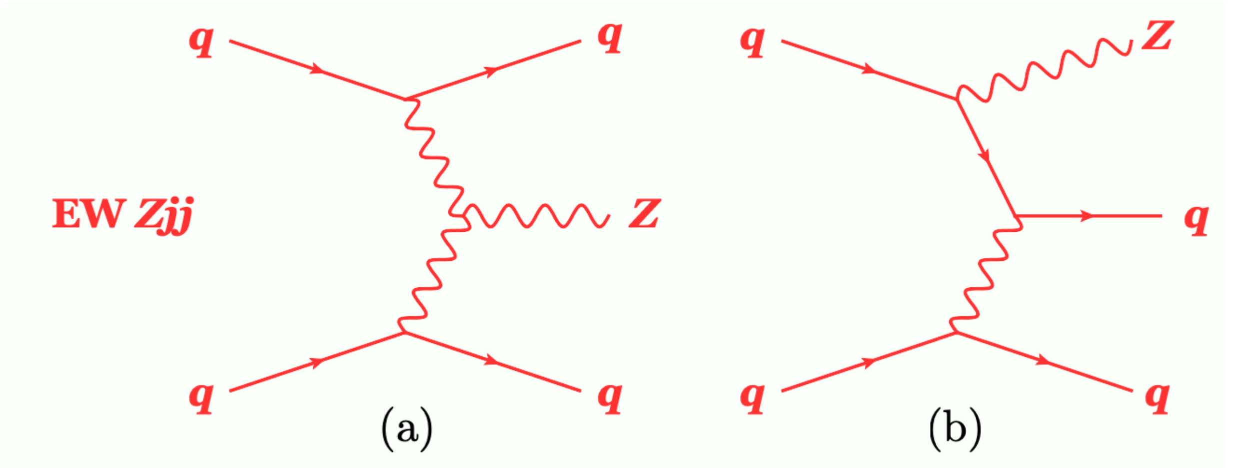
Eur. Phys. J. C 81 (2021) 163

27 June 2020

Contact: ATLAS Standard Model conveners

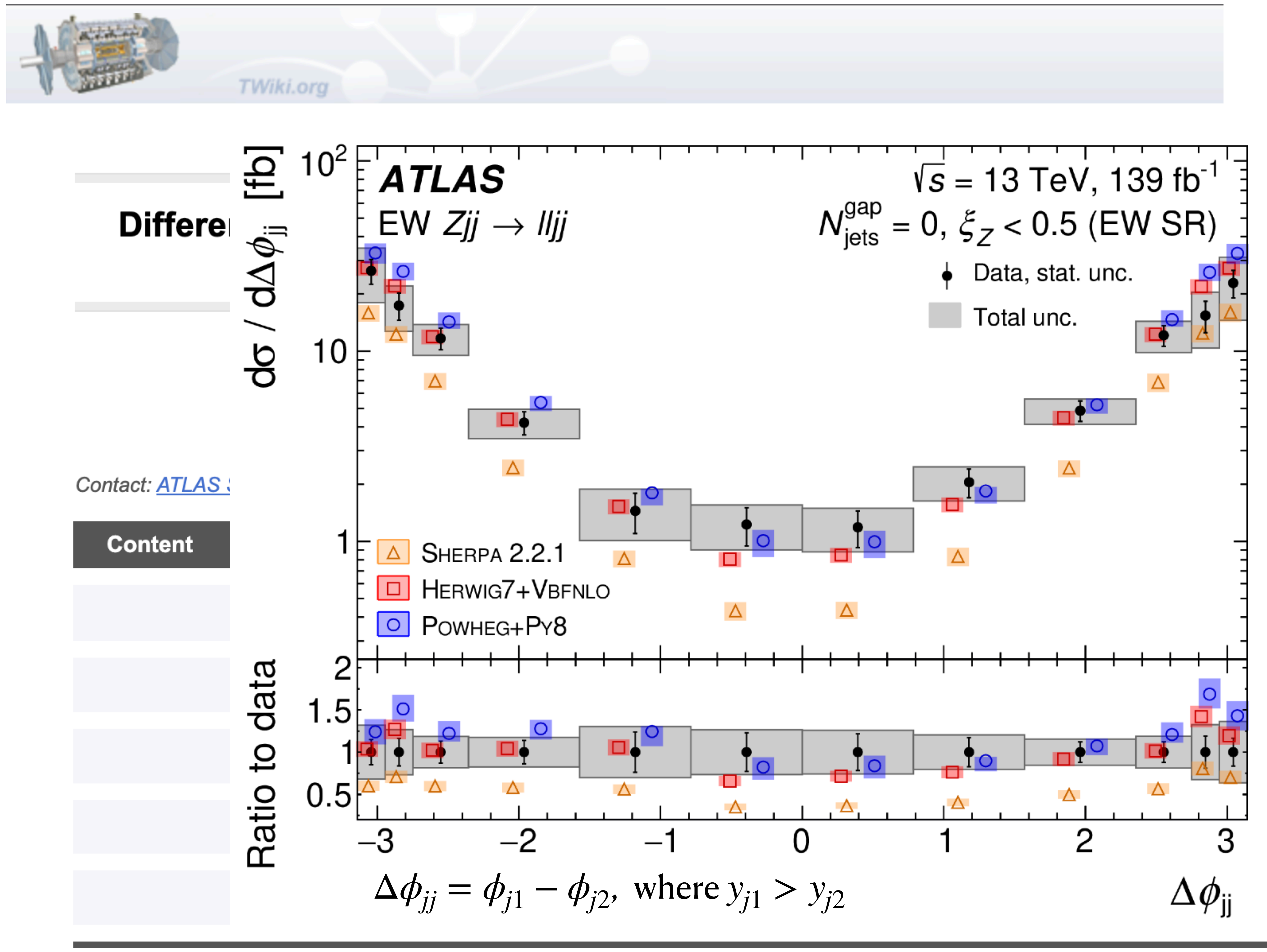
Content	Preview
e-print arXiv:2006.15458 - internal	pdf from arXiv
Inspire record	-
Data points	-
Rivet analysis routine	-
Figures Tables Auxiliary Material	-

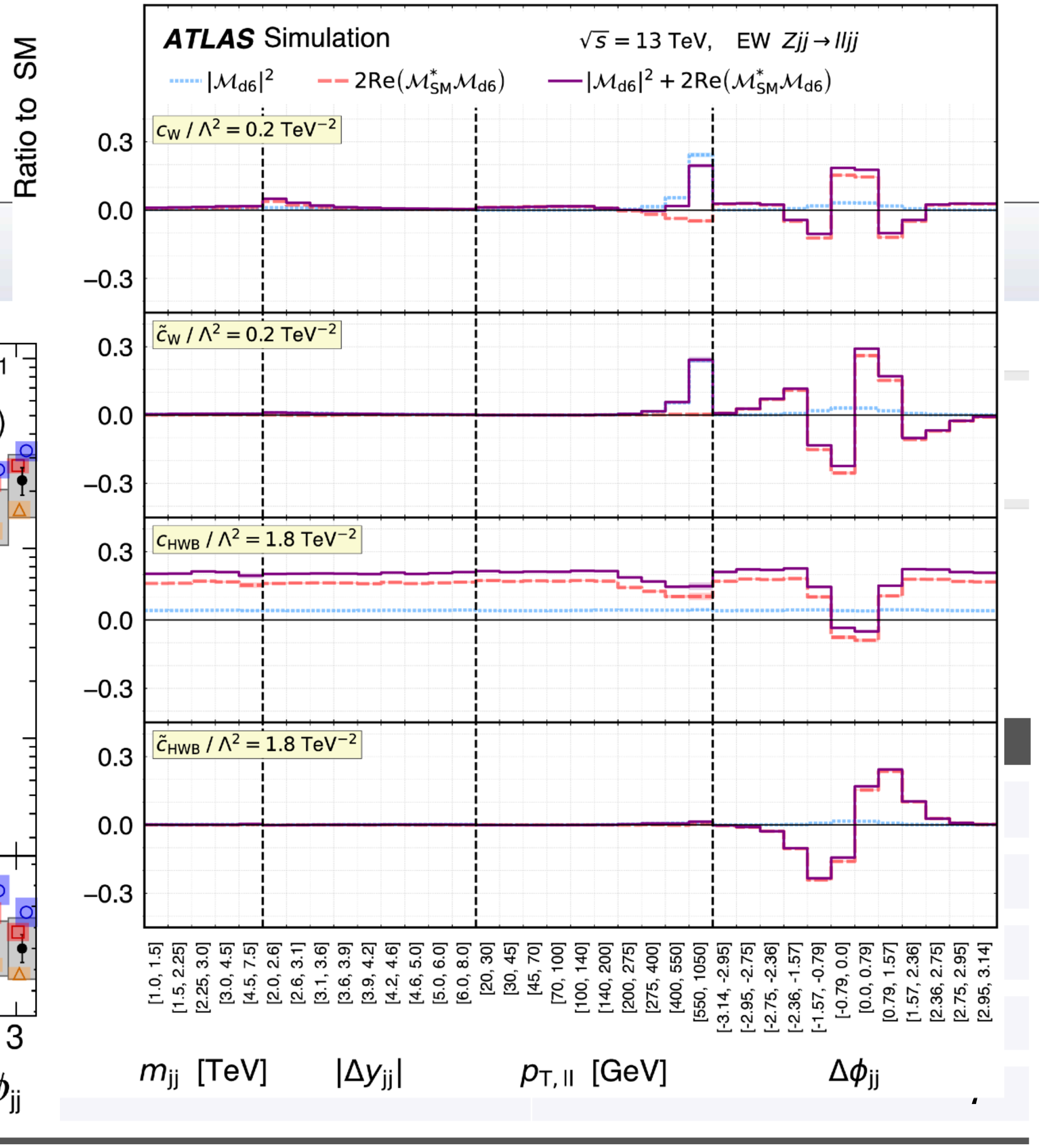
Useful tools at hand



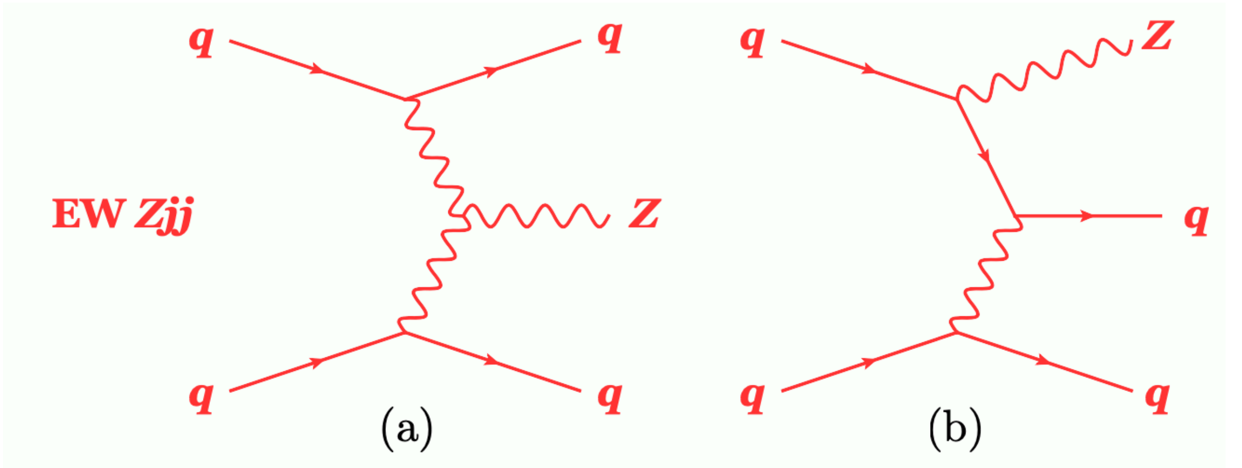
- HepData stores the measurements with associated uncertainties
  - hepdata.net
- Rivet is synchronized with the HepData entry
  - Ensures predictions defined in accordance with the data
- Fast and effective

Impact from BSM modifications on the measured EW Zjj differential cross sections



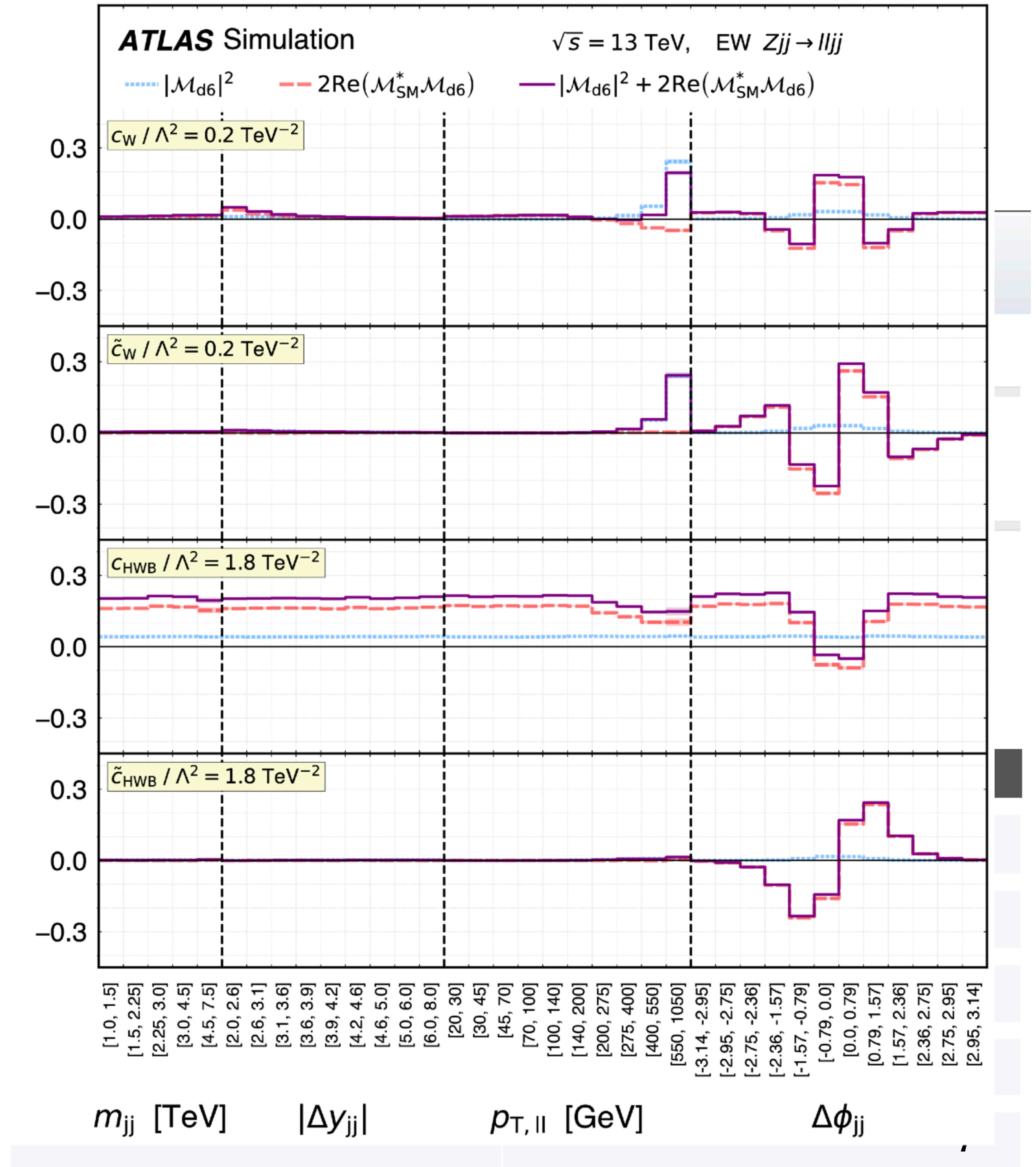


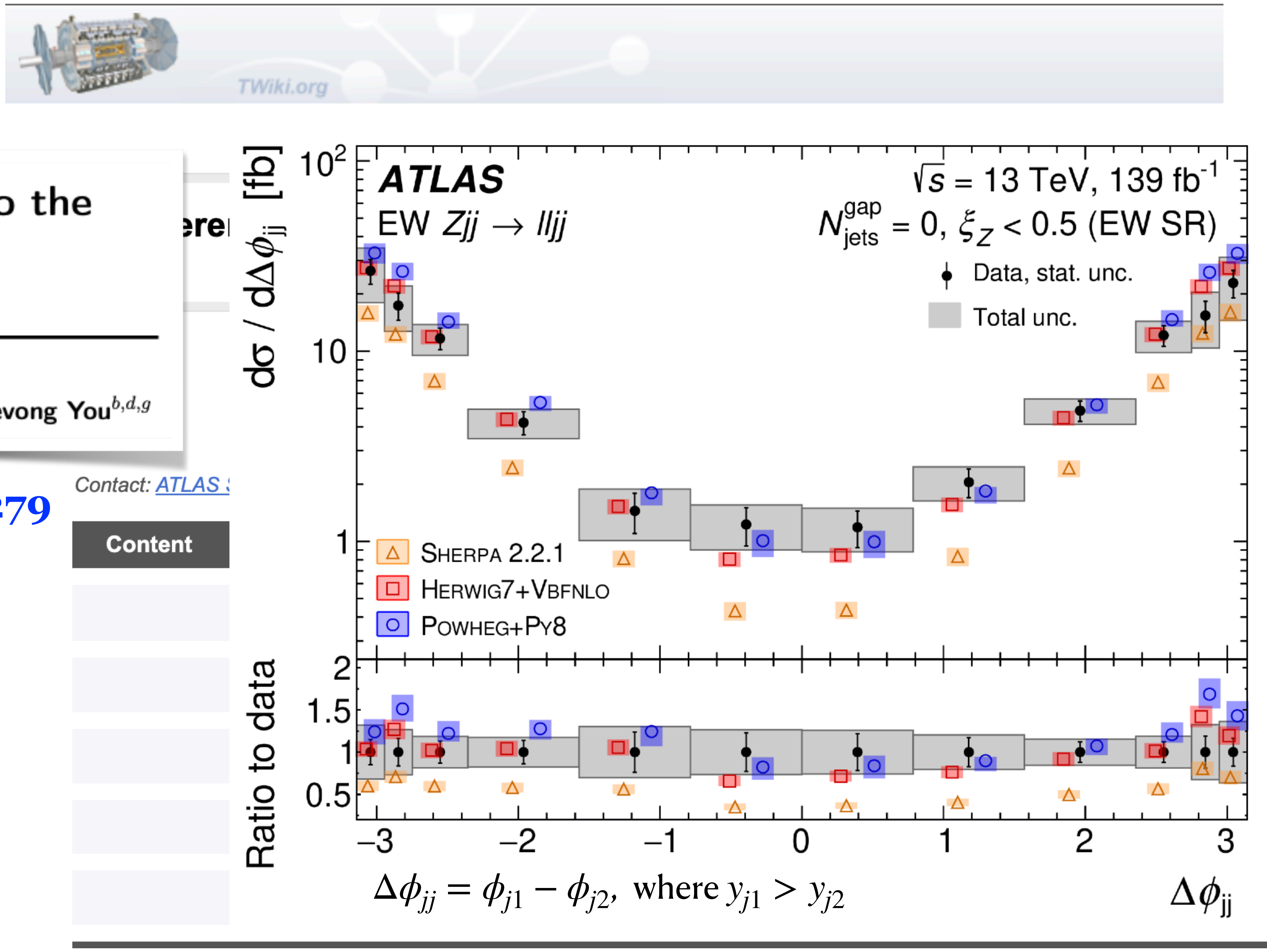
Useful tools at hand



- HepData stores the measurements with associated uncertainties
  - hepdata.net
- Rivet is synchronized with the HepData entry
  - Ensures predictions defined in accordance with the data
- Fast and effective

Impact from BSM modifications on the measured EW Zjj differential cross sections

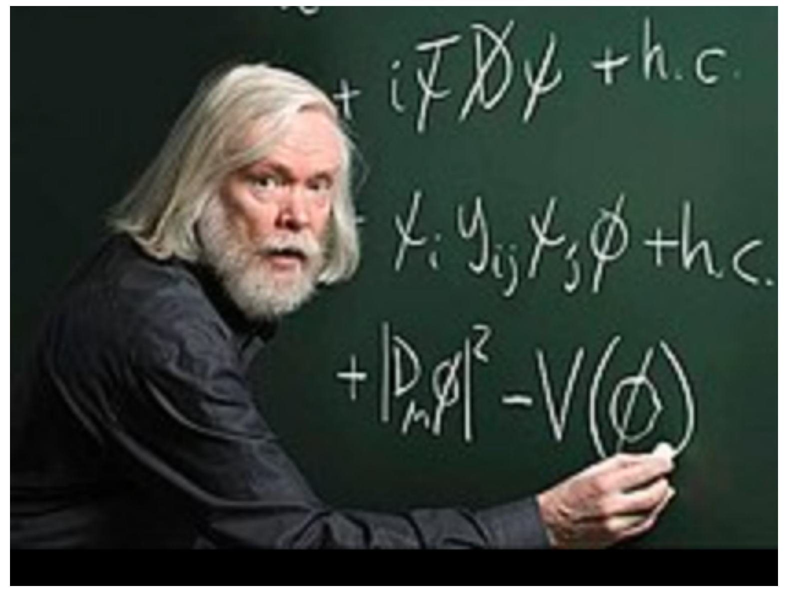




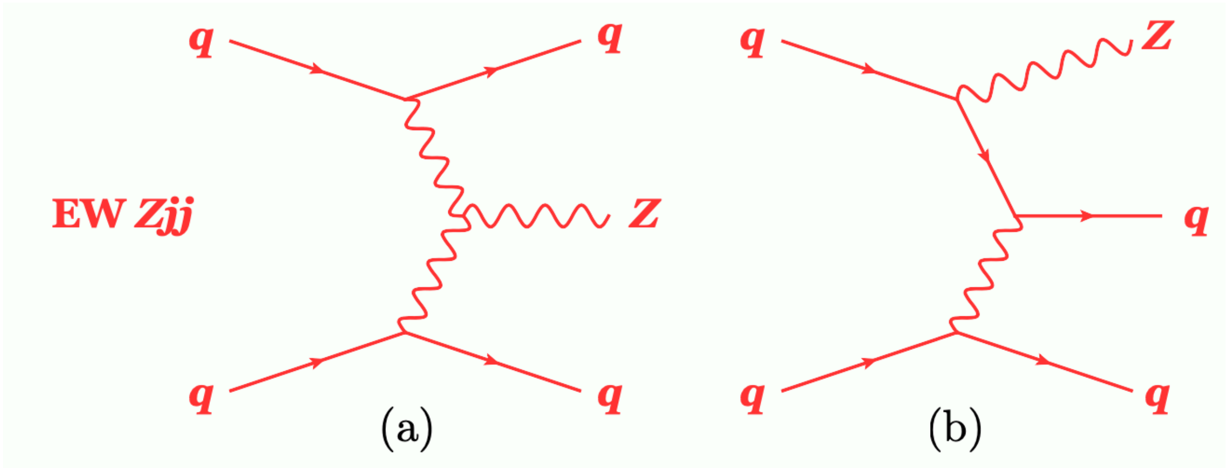
Top, Higgs, Diboson and Electroweak Fit to the Standard Model Effective Field Theory

John Ellis,[a,b,c] Maeve Madigan,[d] Ken Mimasu,[a] Veronica Sanz[e,f] and Tevong You[b,d,g]

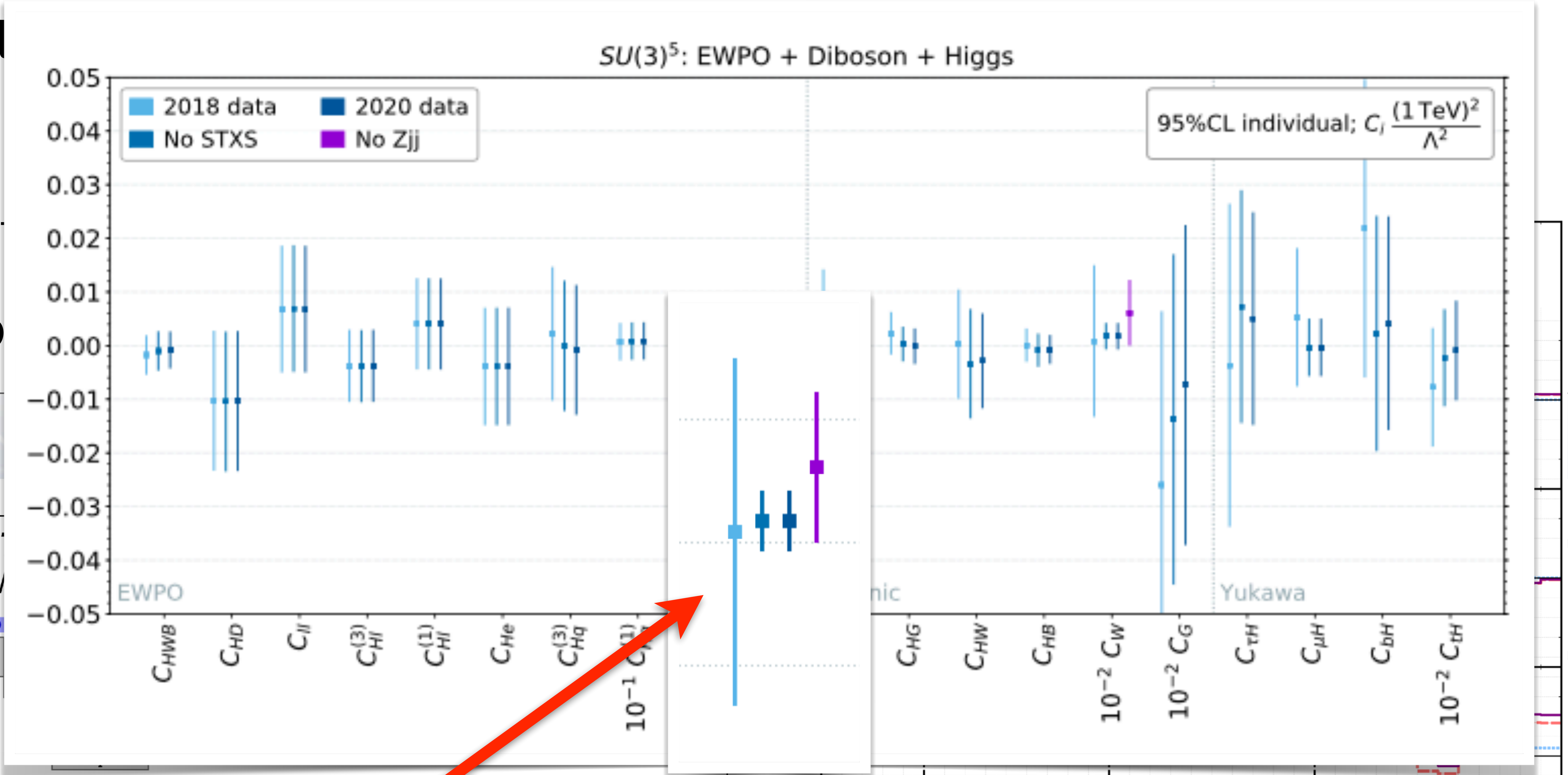
[arXiv:2012.02779, JHEP 04 (2021) 279](https://arxiv.org/abs/2012.02779)

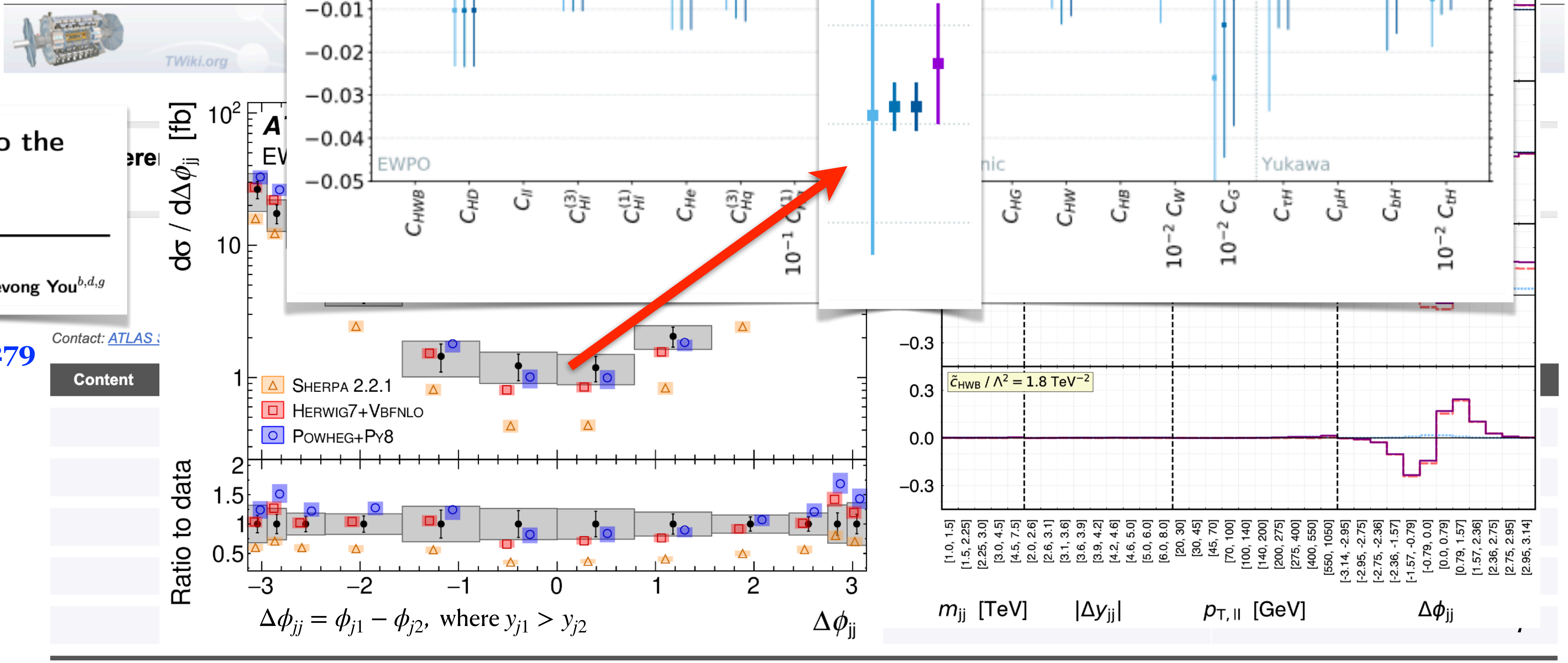


Useful tools at hand



- HepData stores the measurements wi
  - hepdata.net
- Rivet is synchronized with the HepDa
  - Ensures predictions defined in acco
- Fast and effective

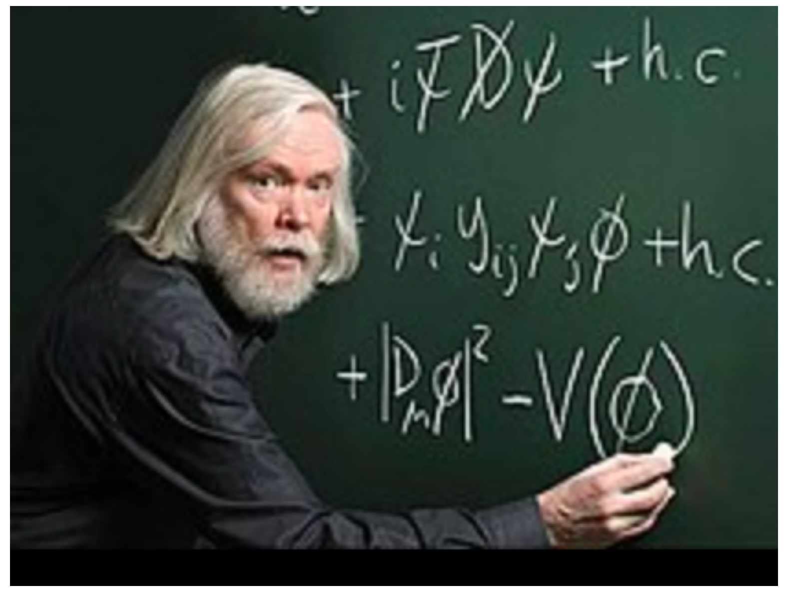




Top, Higgs, Diboson and Electroweak Fit to the Standard Model Effective Field Theory

John Ellis,[a,b,c] Maeve Madigan,[d] Ken Mimasu,[a] Veronica Sanz[e,f] and Tevong You[b,d,g]

[arXiv:2012.02779, JHEP 04 (2021) 279](https://arxiv.org/abs/2012.02779)



Limitations with current approach

- As we have seen, current approach for precision measurements is quite nice
- However there are a few short-comings
- When designing our measurement, we need to a-priori settle on

  A. Exact list of observables to measure

  B. Bin-boundaries for each measurement

  C. We are limited to measure one (or a few) observables at the time

Recent developments in machine learning opens up new possibilities

Limitations with current approach

- As we have seen, current approach for precision measurements is quite nice
- However there are a few short-comings
- When designing our measurement, we need to a-priori settle on
  A. Exact list of observables to measure — **User can combine measured variables**
  B. Bin-boundaries for each measurement — **Unbinned**
  C. We are limited to measure one (or a few) observables at the time — **High dimensionality**

Recent developments in machine learning opens up new possibilities

Classification

Classifier
$f(\vec{x})$

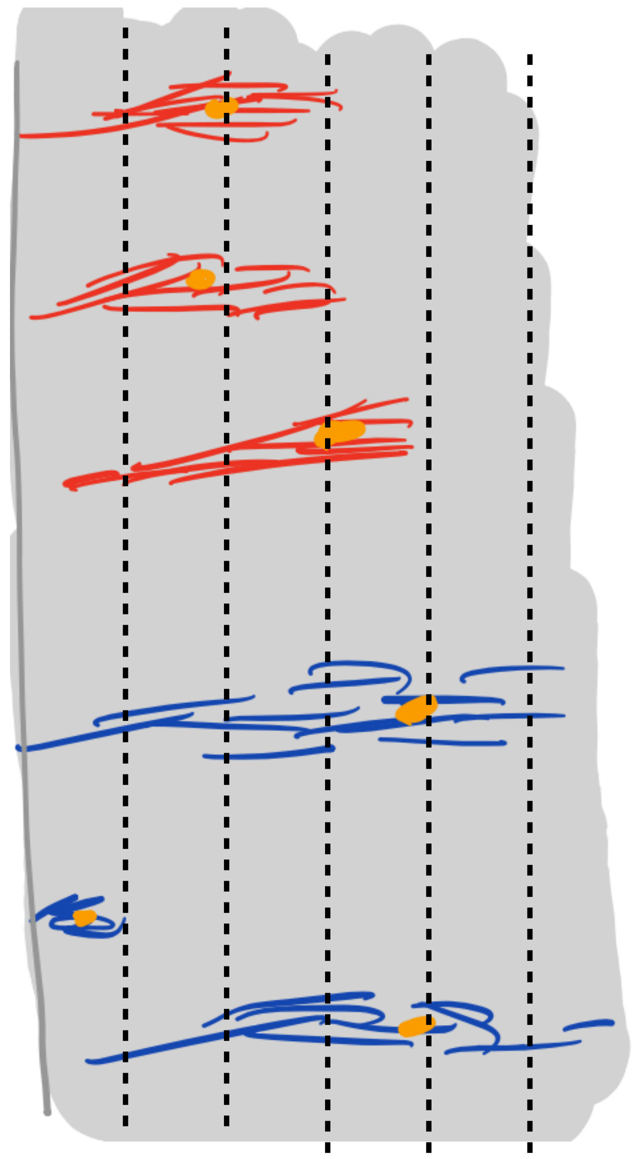
- Most common application of machine learning in particle physics is classification
- Goal: discriminate 'signal' from 'background'
  - Example: Detector signals from real electrons vs hadrons/photons

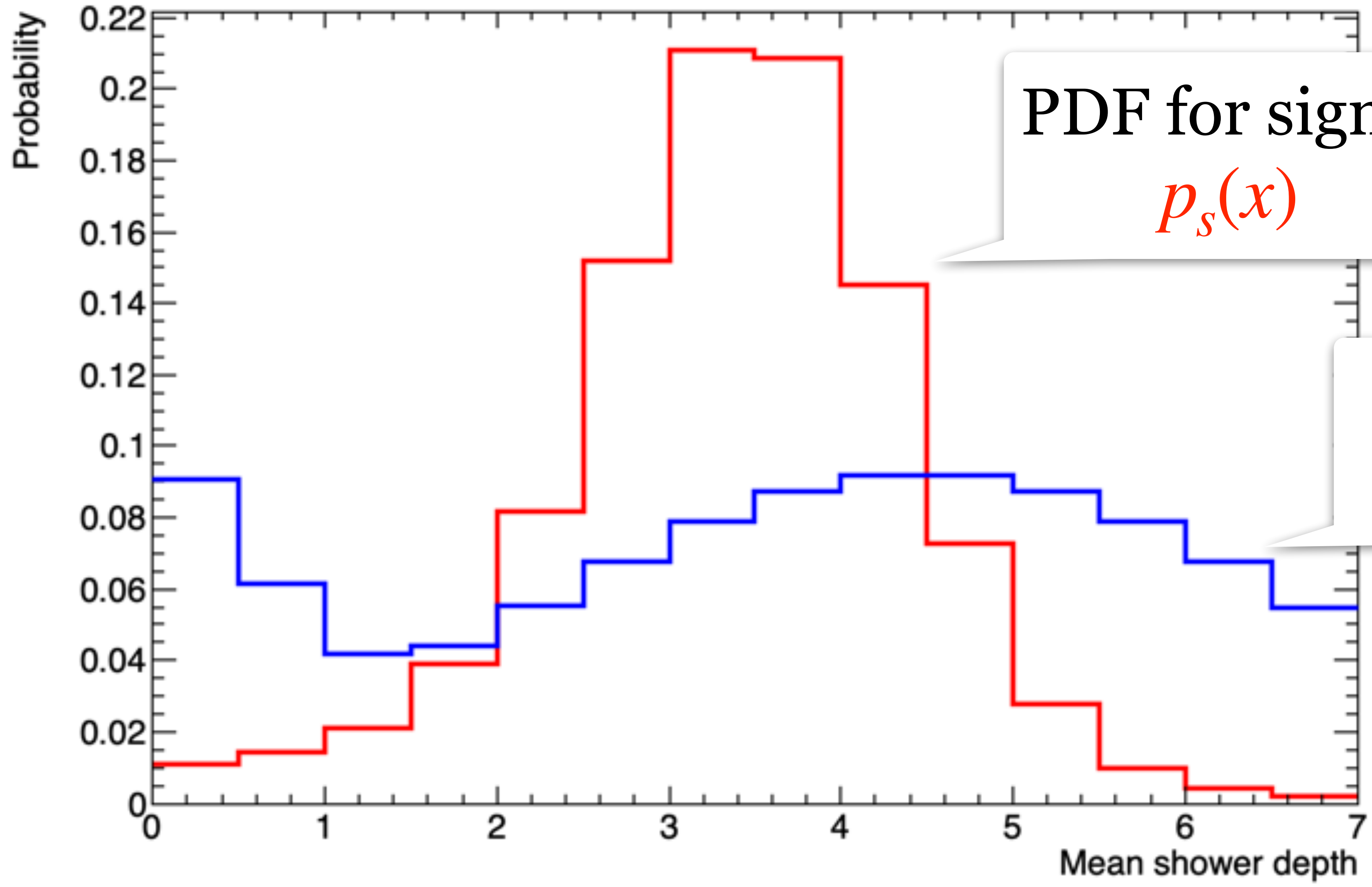
PDF:
$\int p(x)\, dx = 1$

3 real electrons
(signal)

3 fake electrons
(background)

From hadrons,
photons or noise bursts …





PDF for signal
$p_s(x)$

Background
$p_b(x)$

Classification

Classifier
$f(\vec{x})$

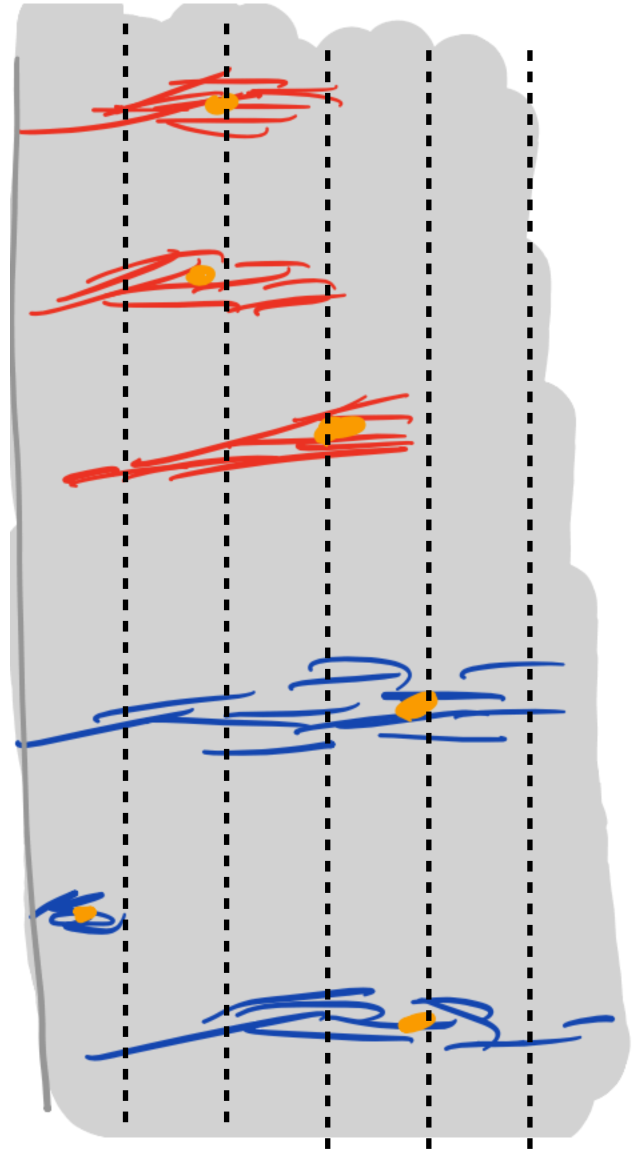
- Most common application of machine learning in particle physics is classification
- Goal: discriminate 'signal' from 'background'
  - Example: Detector signals from real electrons vs hadrons/photons

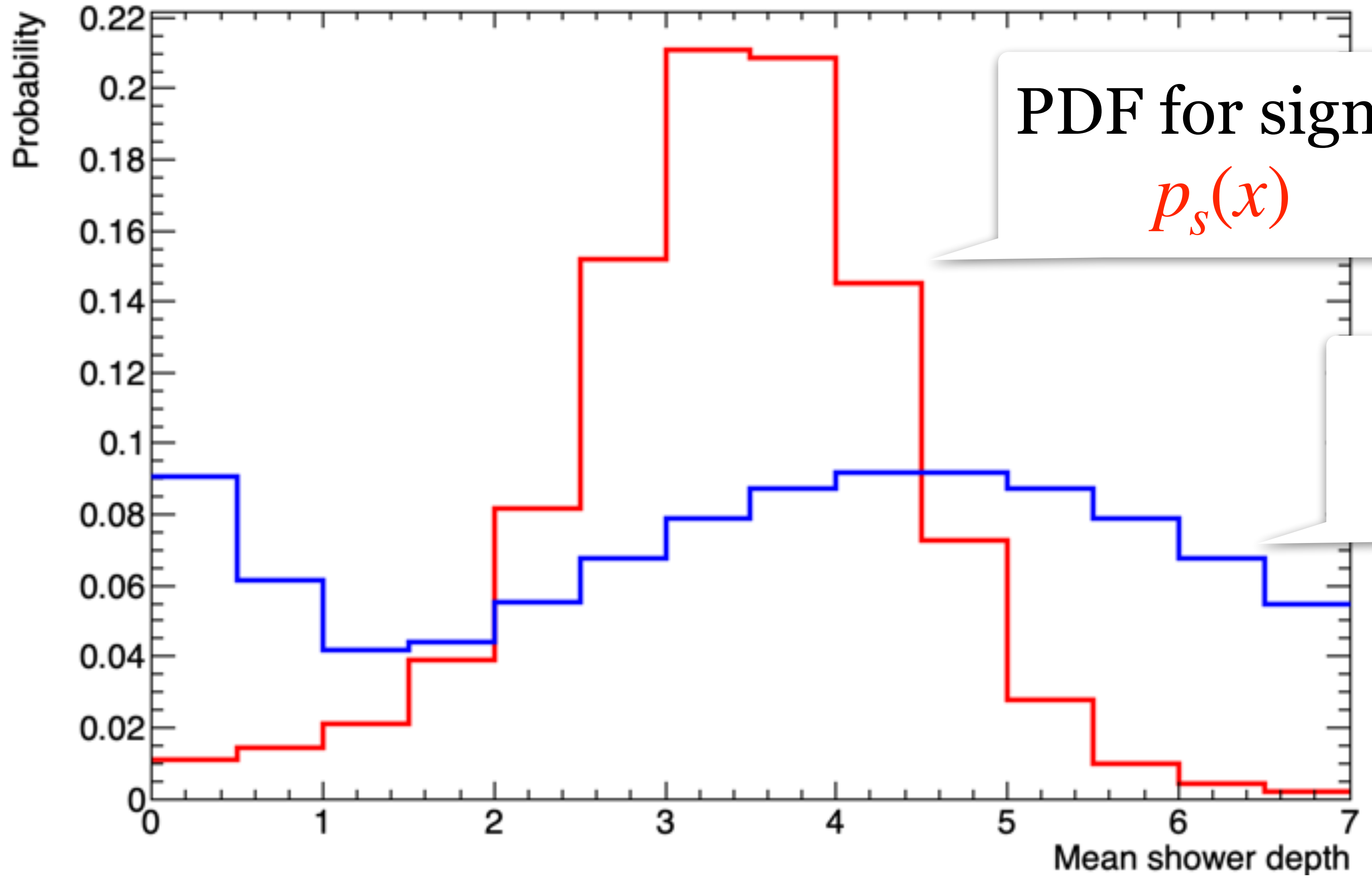
PDF:
$\int p(x)\,dx = 1$

3 real electrons (signal)

3 fake electrons (background)

From hadrons, photons or noise bursts …



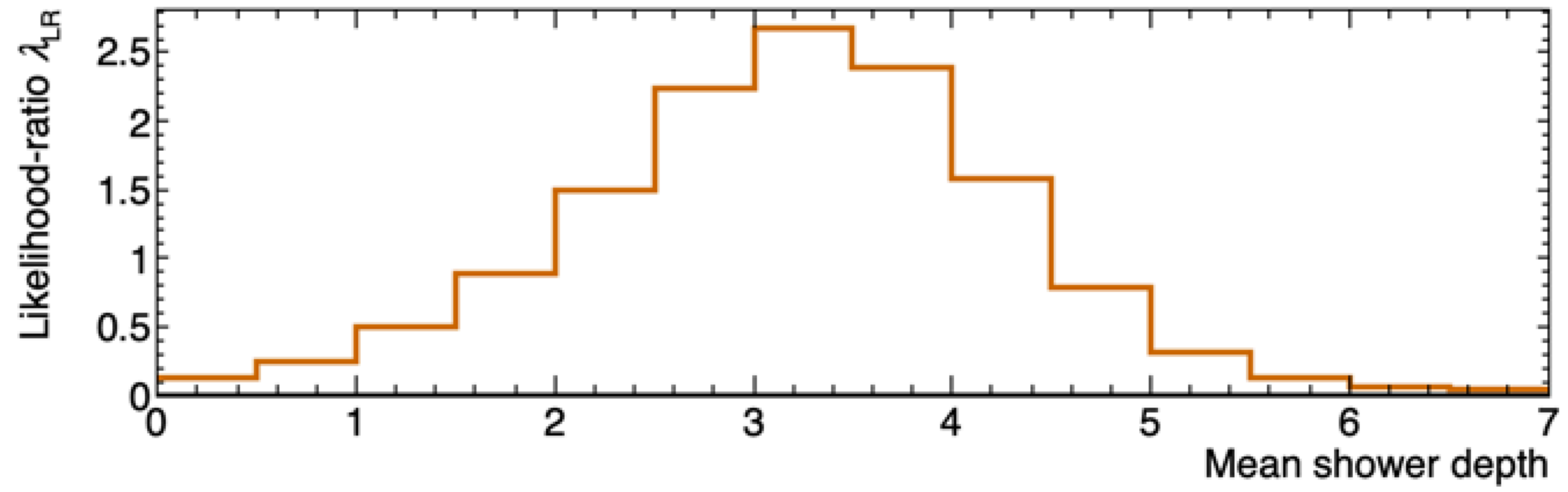


PDF for signal
$p_s(x)$

Background
$p_b(x)$

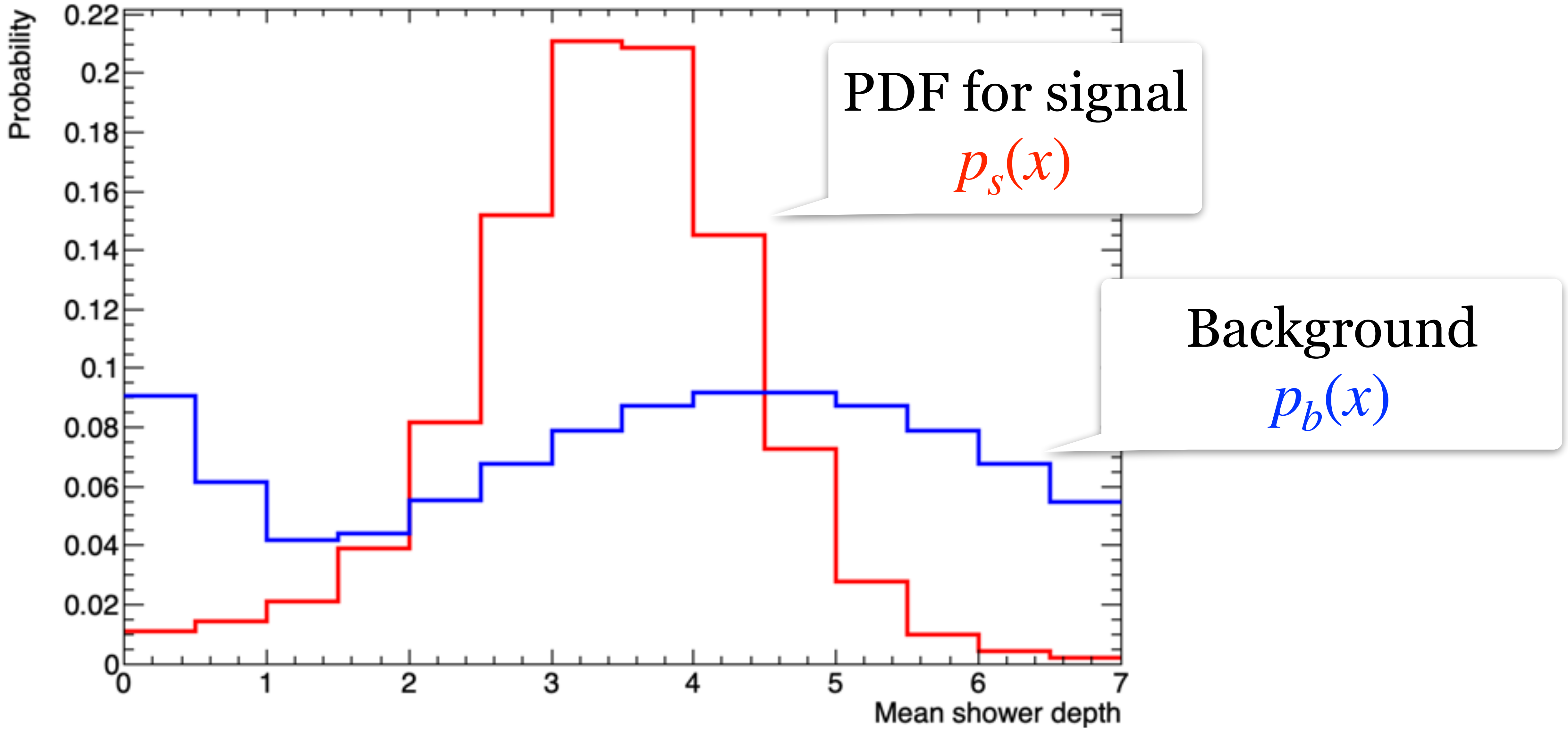
Likelihood ratio:

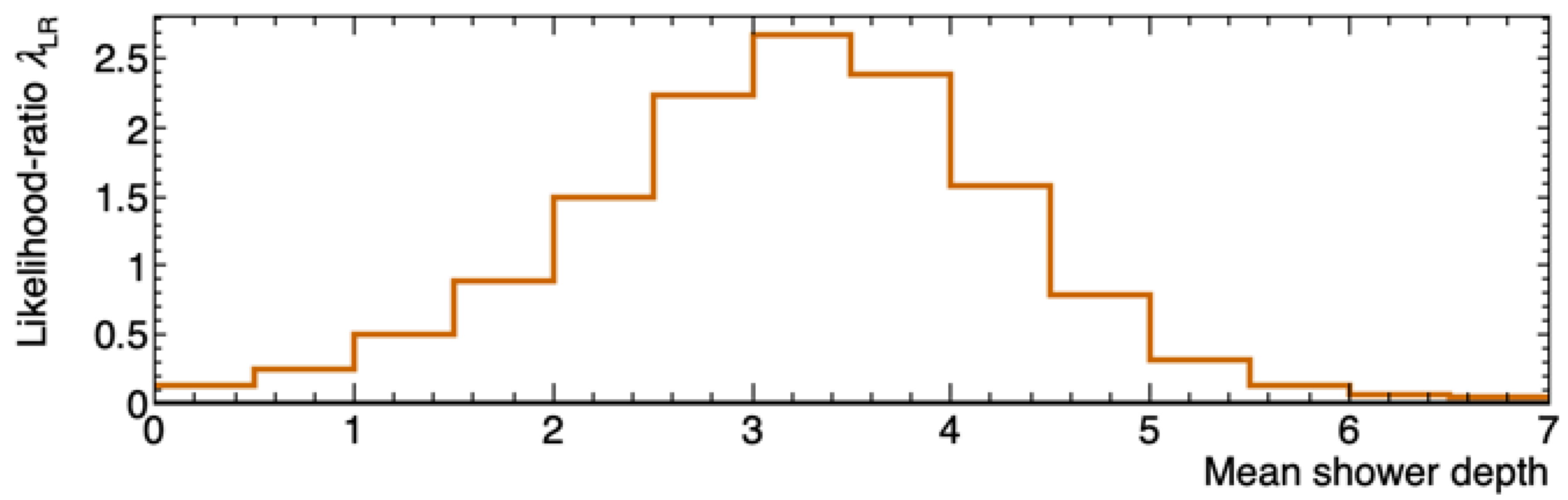
$$\lambda_{\mathrm{LR}} = \frac{p_s(x)}{p_b(x)}$$

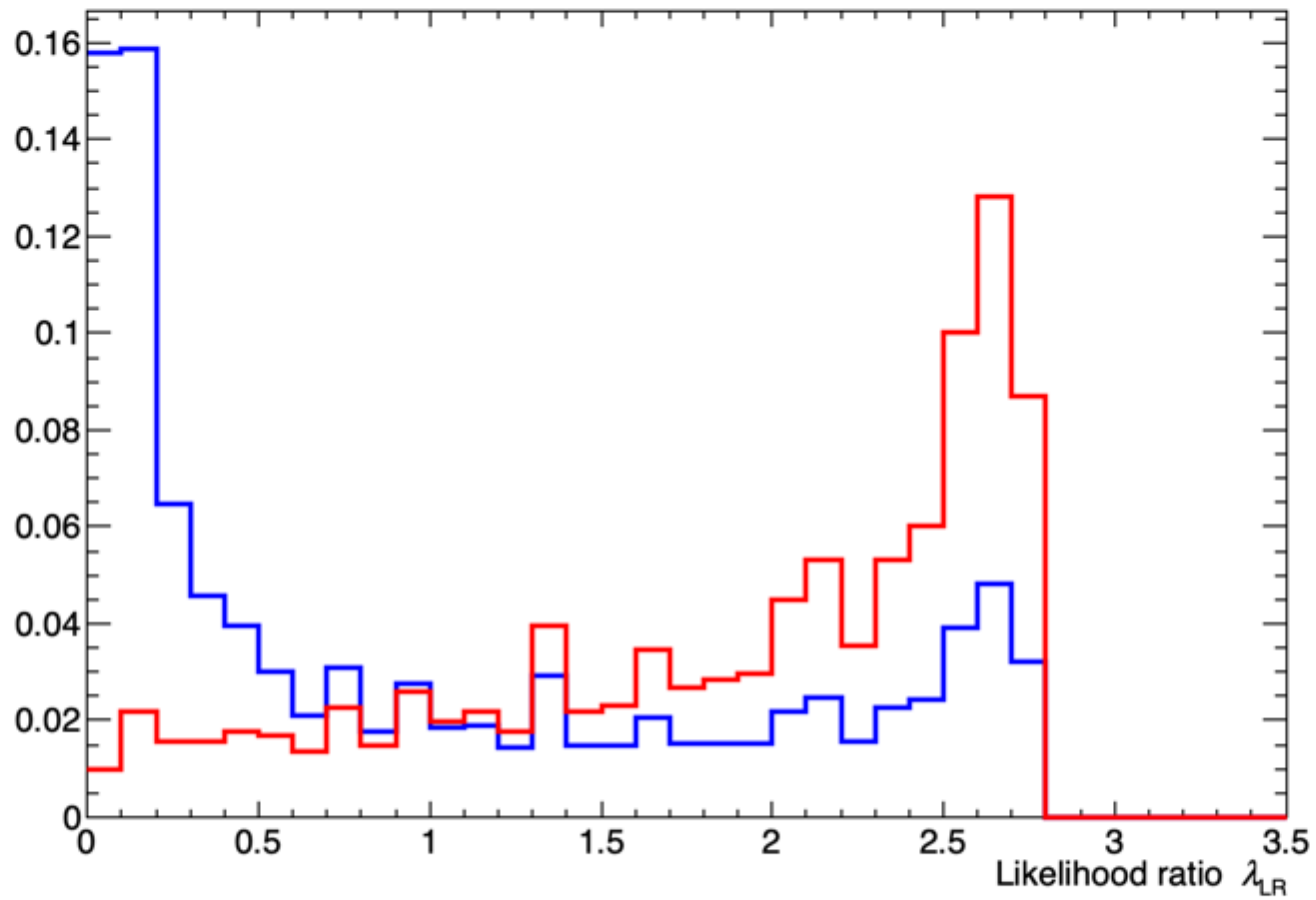


The Neyman-Pearson lemma

- The Neyman-Pearson lemma states that the best achievable discriminant will be the likelihood ratio λ_{LR} (or any monotonic function of it)

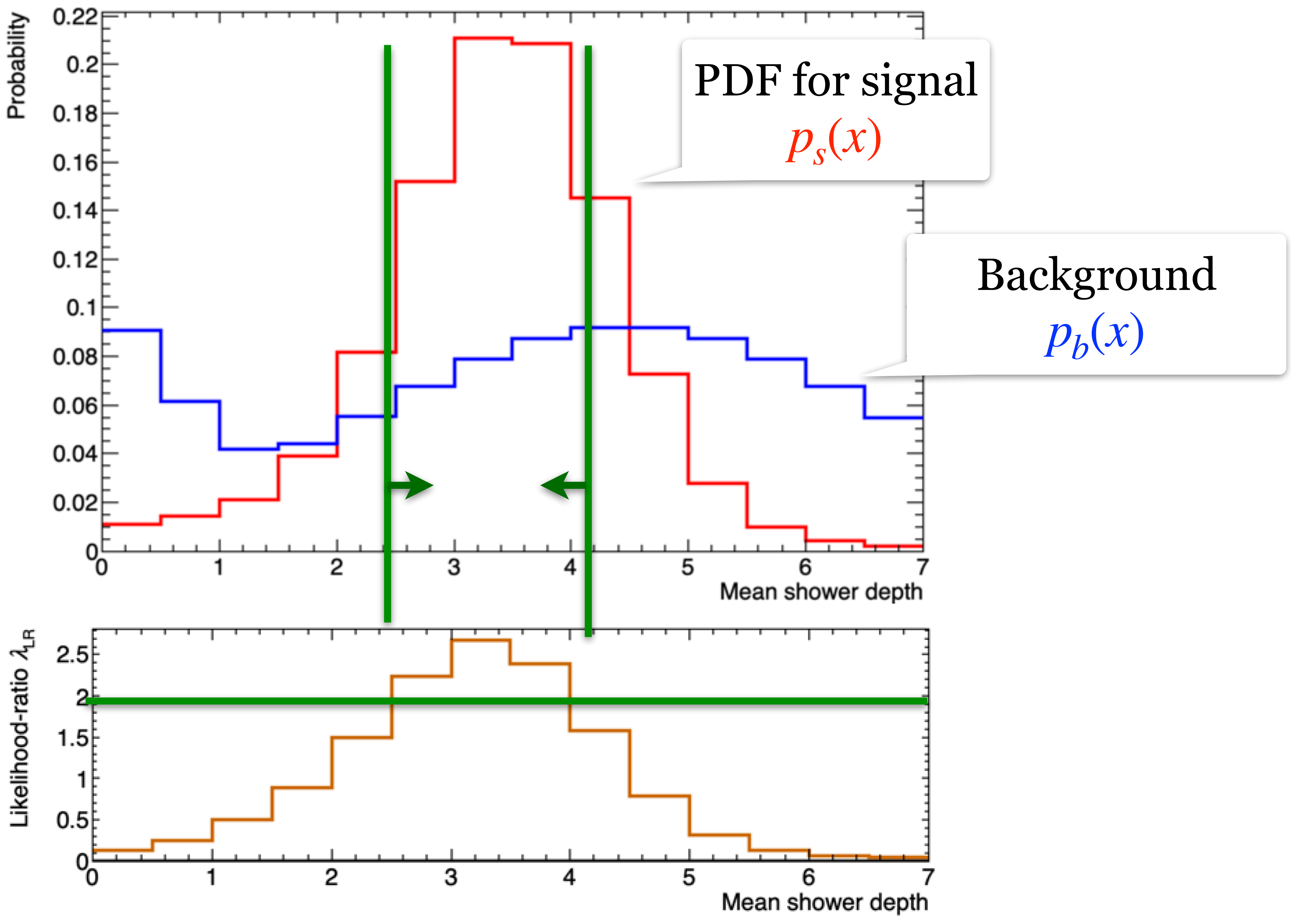


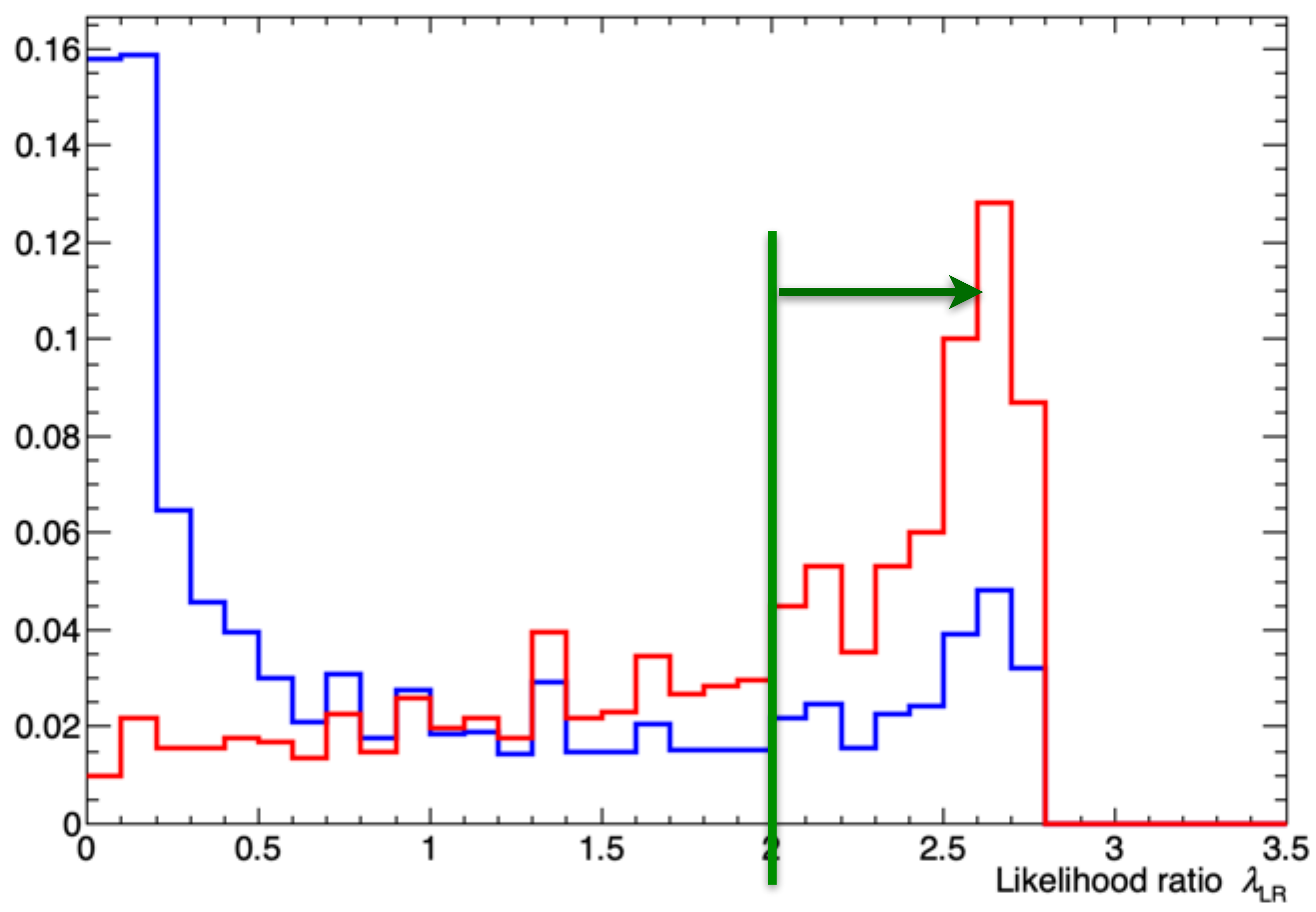




The Neyman-Pearson lemma

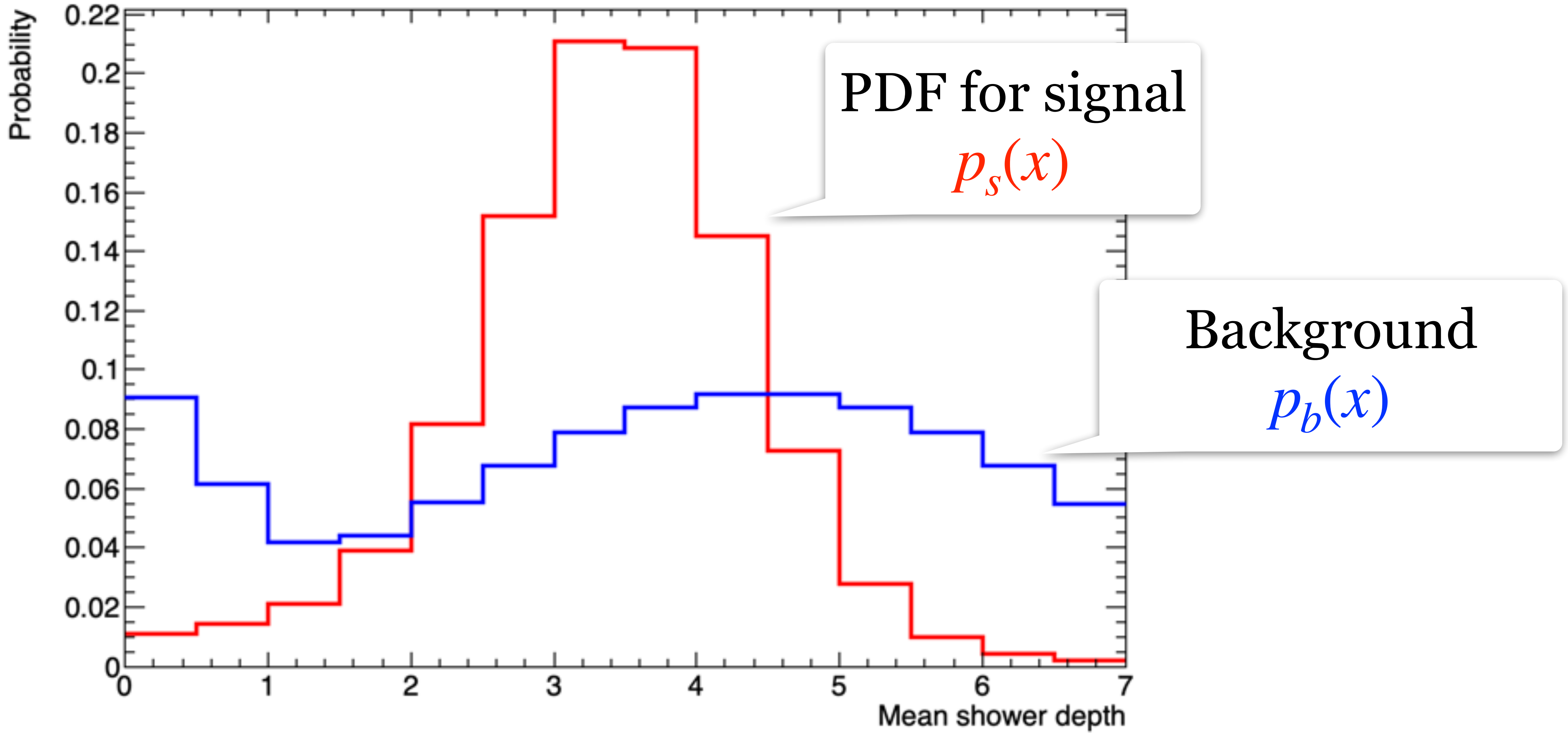
- The Neyman-Pearson lemma states that the best achievable discriminant will be the likelihood ratio λ_{LR} (or any monotonic function of it)

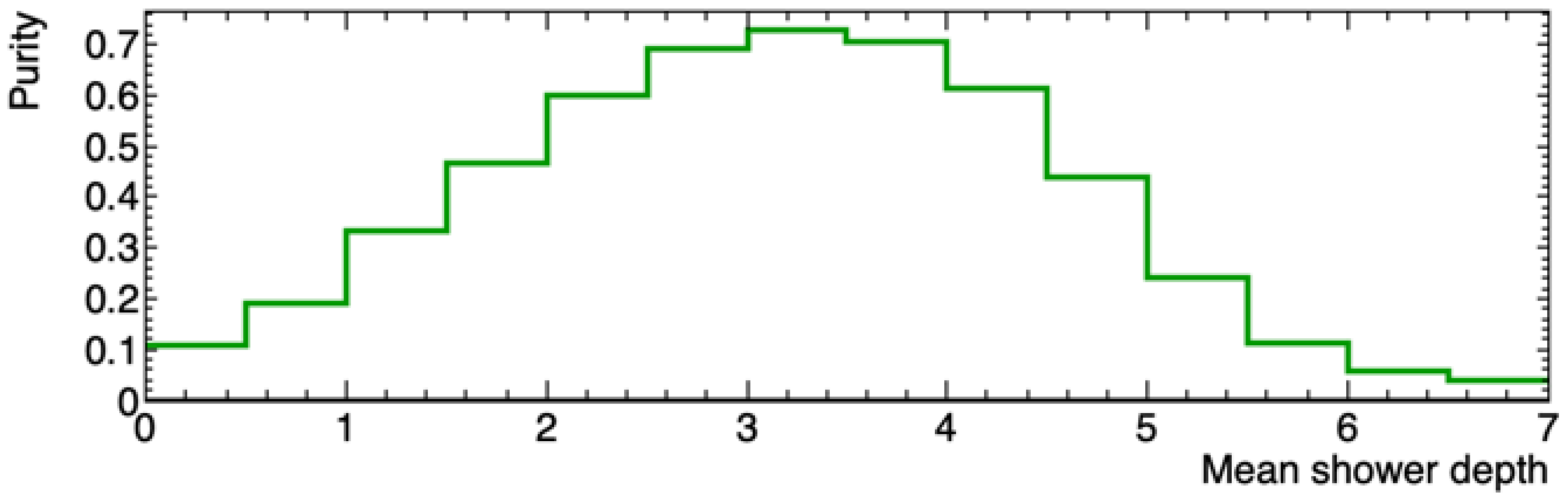




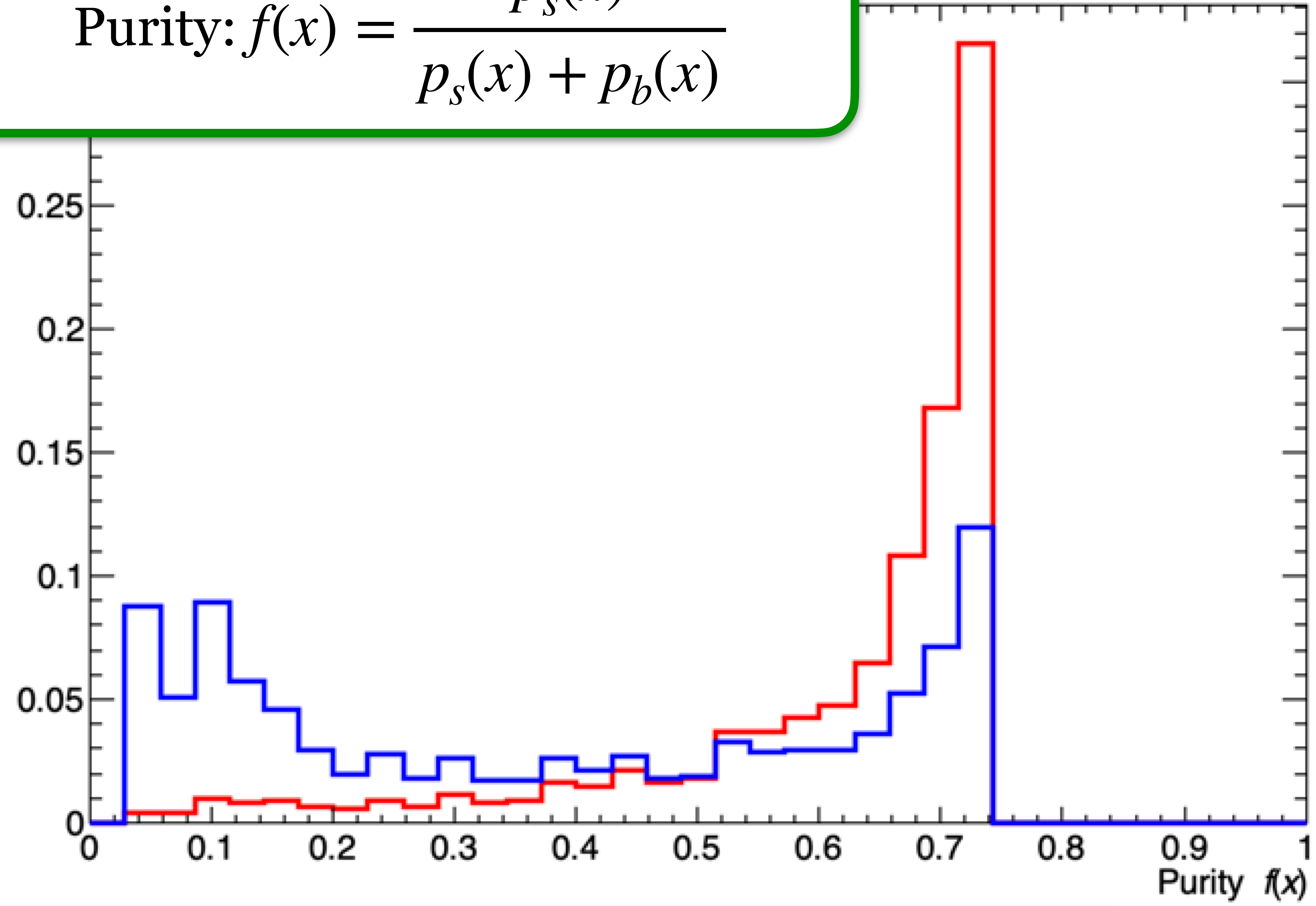
The Neyman-Pearson lemma

- The Neyman-Pearson lemma states that the best achievable discriminant will be the likelihood ratio λ_{LR} (or any monotonic function of it)





Purity: $f(x) = \dfrac{p_s(x)}{p_s(x) + p_b(x)}$



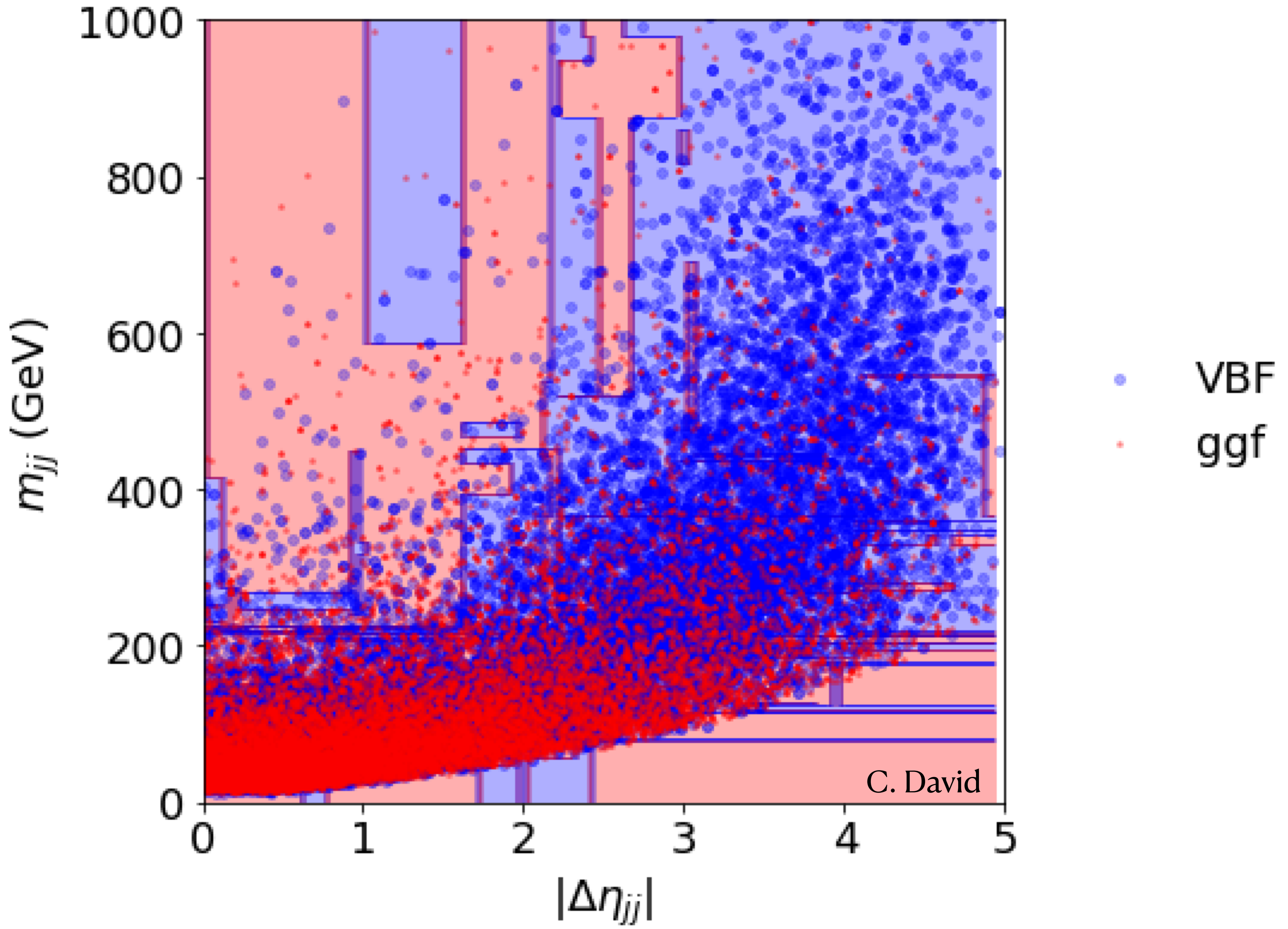
Many machine learning algorithm returns the purity as output
It is closely related to the likelihood ratio

Multivariate analysis

Classifier

$f(\vec{x})$

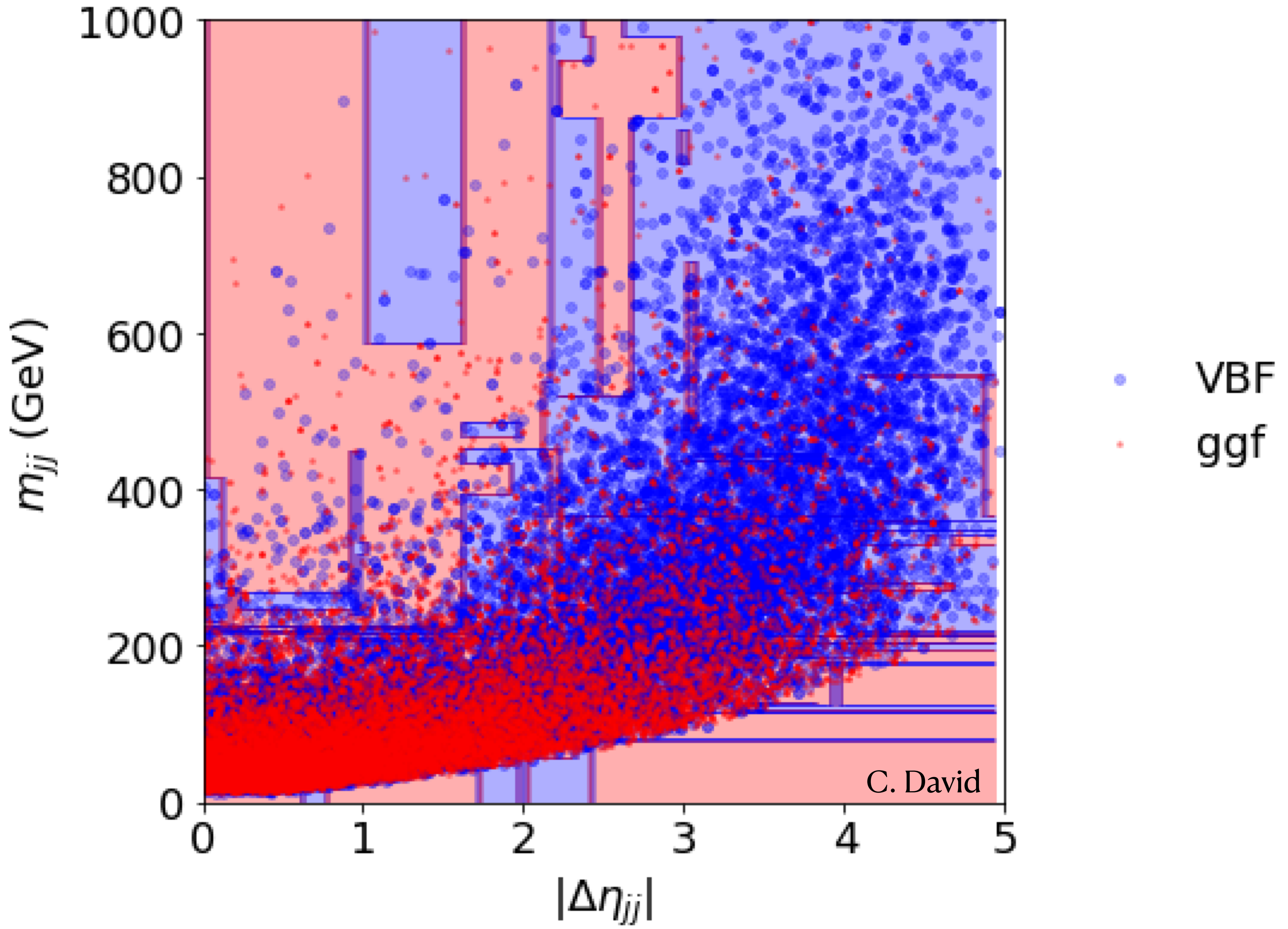
- Previous example only dealt with one input variable (the shower depth)
- We can perform much better if we use more information, i.e. more distinguish features (input variables): $x \rightarrow \vec{x}$
- Neyman-Pearson lemma still holds, but quickly challenging to estimate $p_s(\vec{x})$ and $p_b(\vec{x})$



Multivariate analysis

Classifier
$f(\vec{x})$

- Previous example only dealt with one input variable (the shower depth)
- We can perform much better if we use more information, i.e. more distinguish features (input variables): $x \to \vec{x}$
- Neyman-Pearson lemma still holds, but quickly challenging to estimate $p_s(\vec{x})$ and $p_b(\vec{x})$



Multivariate techniques to the rescue

ML used for classification:
features $\vec{x}$ as input, returns $f(\vec{x})$
$f(\vec{x})$ separates signal from background

For many implementations $f(\vec{x})$ will be an estimate of the purity

True for traditional BDTs, and NNs trained with the cross entropy as loss function

Multivariate analysis

Classifier

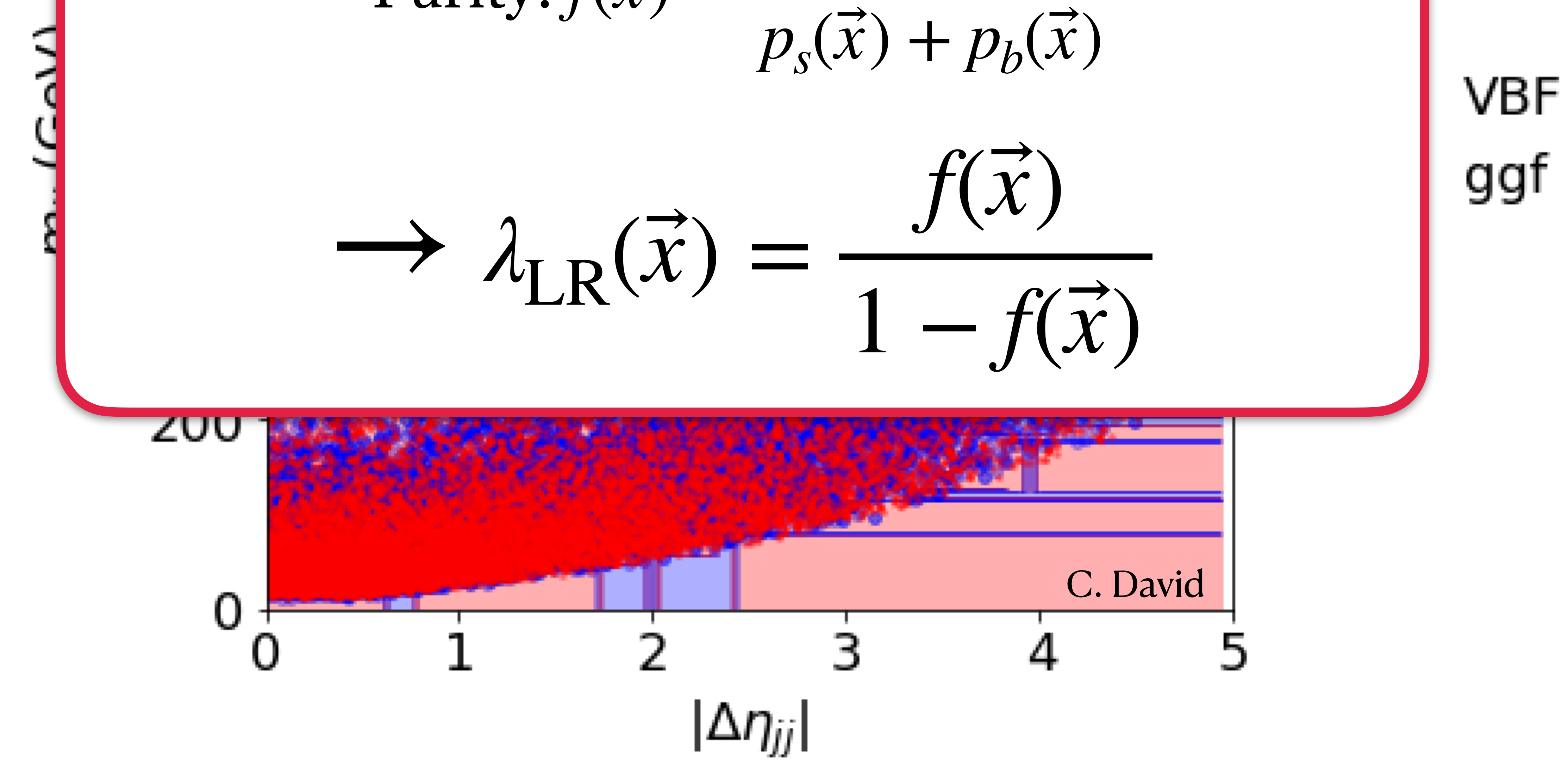
$f(\vec{x})$

- Previous example only dealt with one input variable (the shower depth)
- We can perform much better if we use more information, i.e. more distinguish features (input variables): $x \to \vec{x}$
- Neyman-Pearson lemma still holds, but quickly challenging to estimate $p_s(\vec{x})$ and $p_b(\vec{x})$

Likelihood ratio: $\lambda_{\mathrm{LR}}(\vec{x}) = \dfrac{p_s(\vec{x})}{p_b(\vec{x})}$

Purity: $f(\vec{x}) = \dfrac{p_s(\vec{x})}{p_s(\vec{x}) + p_b(\vec{x})}$

$$\to \lambda_{\mathrm{LR}}(\vec{x}) = \frac{f(\vec{x})}{1 - f(\vec{x})}$$



Multivariate techniques to the rescue

ML used for classification:
features $\vec{x}$ as input, returns $f(\vec{x})$
$f(\vec{x})$ separates signal from background

For many implementations $f(\vec{x})$ will be an estimate of the purity

True for traditional BDTs, and NNs trained with the cross entropy as loss function

Multivariate analysis

Classifier
$f(\vec{x})$

- Previous example only dealt with one input va
- We can perform much better if we use more in atures (input variables): $x \to \vec{x}$
- Neyman-Pearson lemma still holds, but quickl l $p_b(\vec{x})$

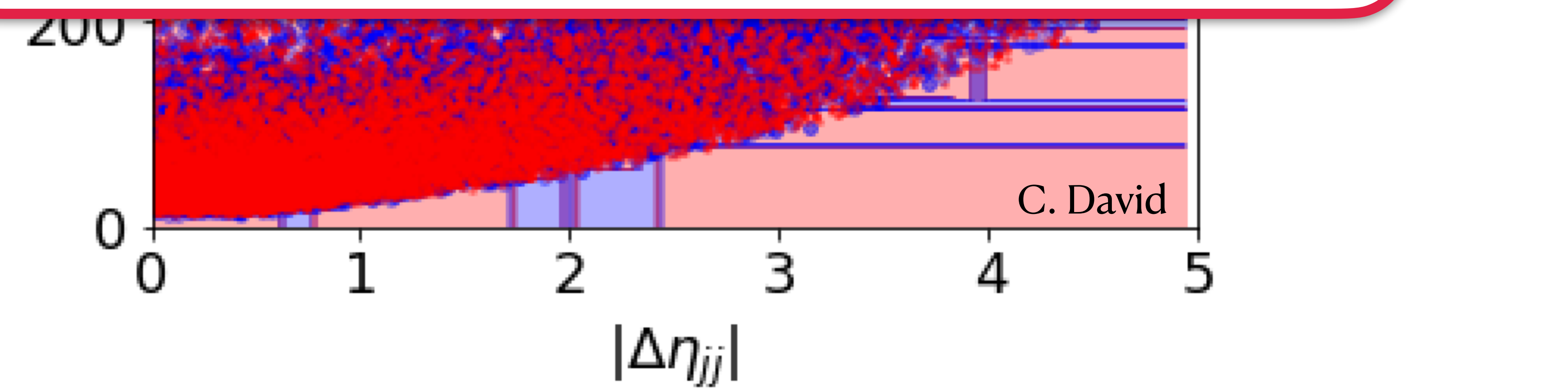
Using **cross entropy** as loss function, finds $f(\vec{x})$ that maximizes:

$$\sum_{\text{sig}} w_i \ln(f(\vec{x}_i)) + \sum_{\text{bkg}} w_i \ln(1 - f(\vec{x}_i))$$

Likelihood ratio: $\lambda_{\mathrm{LR}}(\vec{x}) = \dfrac{p_s(\vec{x})}{p_b(\vec{x})}$

Purity: $f(\vec{x}) = \dfrac{p_s(\vec{x})}{p_s(\vec{x}) + p_b(\vec{x})}$

$$\to \lambda_{\mathrm{LR}}(\vec{x}) = \frac{f(\vec{x})}{1 - f(\vec{x})}$$



Multivariate techniques to the rescue

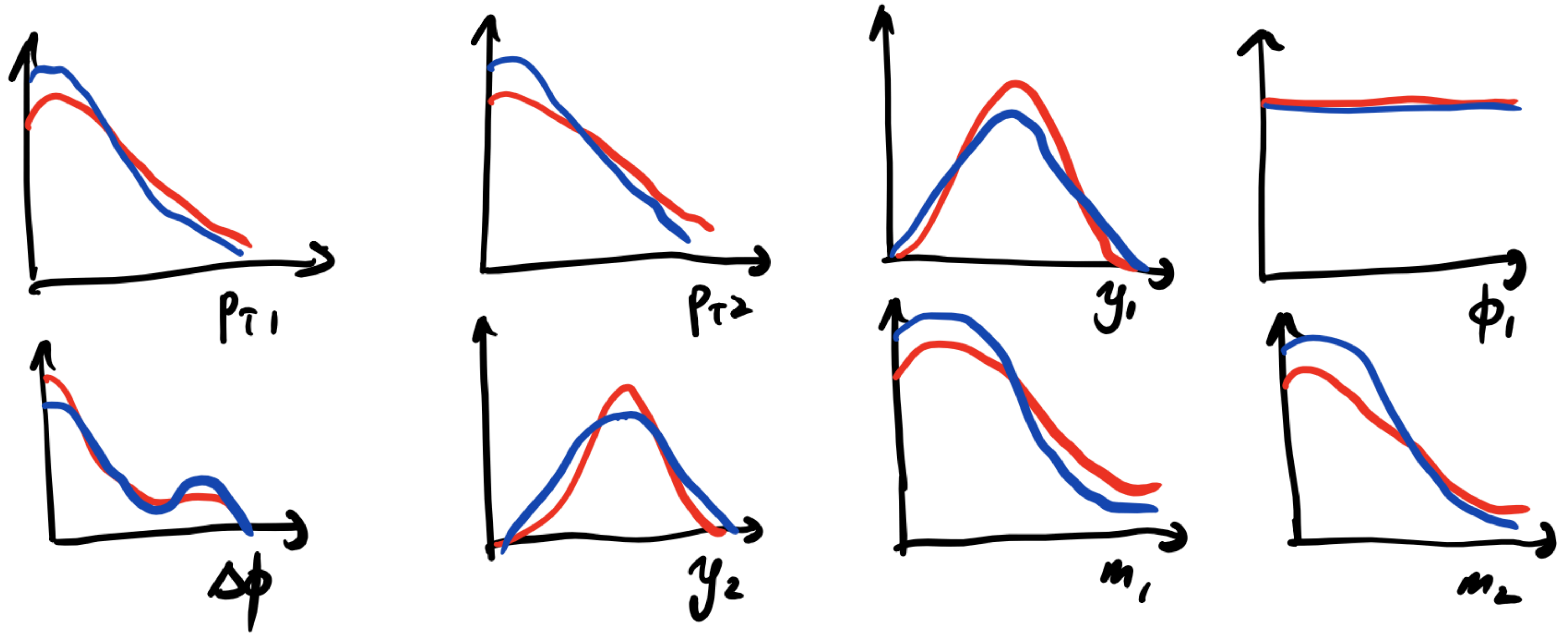
ML used for classification:
features $\vec{x}$ as input, returns $f(\vec{x})$
$f(\vec{x})$ separates signal from background

For many implementations $f(\vec{x})$ will be an estimate of the purity

True for traditional BDTs, and NNs trained with the cross entropy as loss function

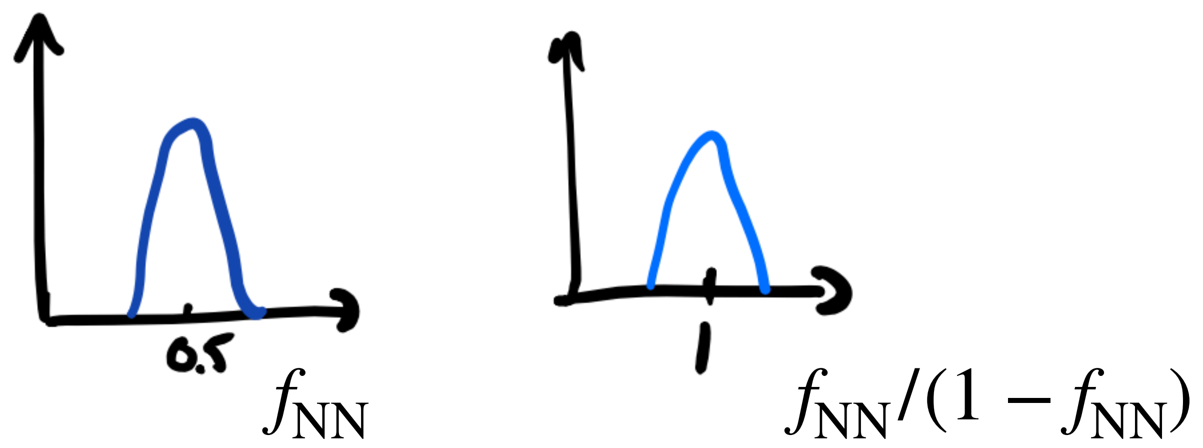
Using ML to reweight event samples

- Consider two MC samples of the same process
  - One fancy MC that takes a lot of computer resources (‘signal’)
  - One simple MC, that is very fast to generate ‘background’
- Next, we train a ML to separate the two using, say 8 input variables $\vec{x} = (x_1, \ldots, x_8)$



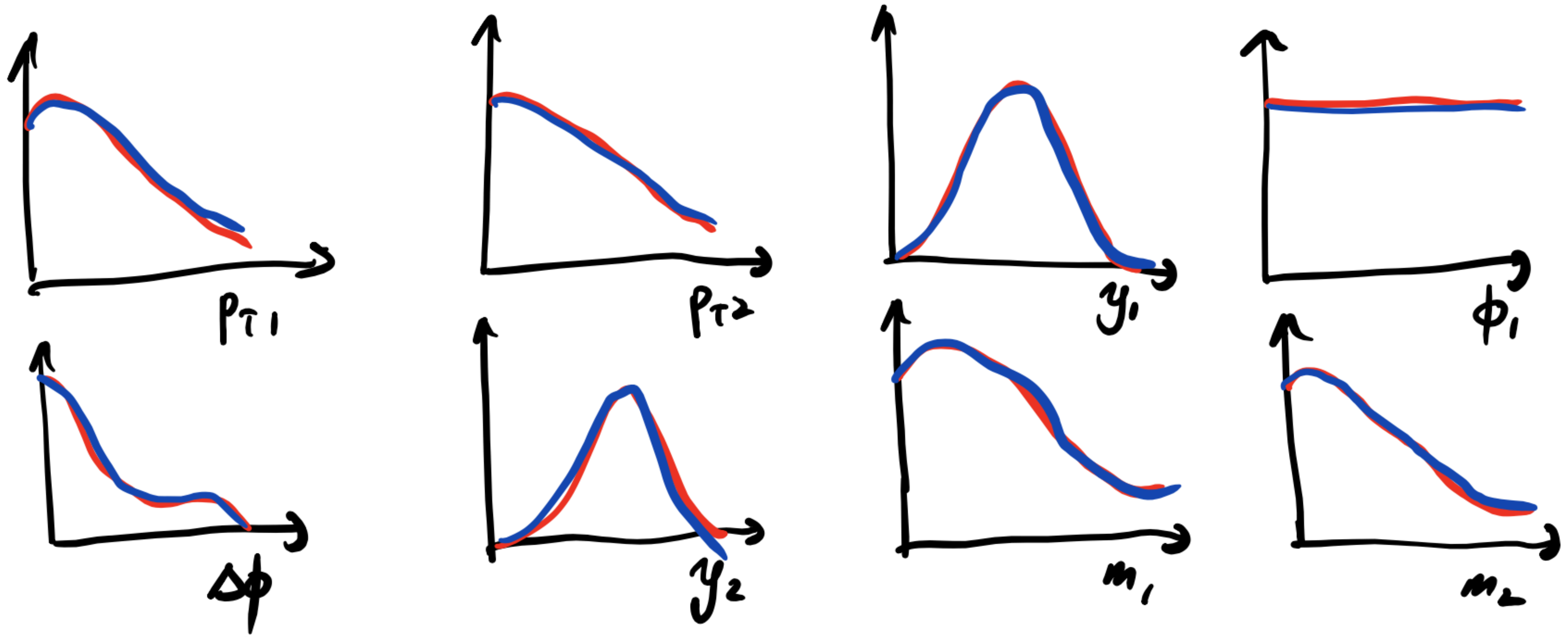
A neural network trained with cross entropy as loss function will return $f_{\mathrm{NN}}(\vec{x})$, that estimates the purity. An estimate of the likelihood ratio is given by

$$\hat{\lambda}(\vec{x}) = \frac{f_{\mathrm{NN}}(\vec{x})}{1 - f_{\mathrm{NN}}(\vec{x})}$$



Using ML to reweight event samples

- Consider two MC samples of the same process
  - One fancy MC that takes a lot of computer resources ('signal')
  - One simple MC, that is very fast to generate 'background'
- Next, we train a ML to separate the two using, say 8 input variables $\vec{x} = (x_1, \ldots, x_8)$



A neural network trained with cross entropy as loss function will return $f_{\mathrm{NN}}(\vec{x})$, that estimates the purity. An estimate of the likelihood ratio is given by

$$\hat{\lambda}(\vec{x}) = \frac{f_{\mathrm{NN}}(\vec{x})}{1 - f_{\mathrm{NN}}(\vec{x})}$$

We can use this quantity as a per-event weight to the cheap MC to make it agree with the fancy one!

$$w(\vec{x}) = f_{\mathrm{NN}}(\vec{x}) \Big/ 1 - f_{\mathrm{NN}}(\vec{x})$$

The NN → an 8-dimensional reweighing function

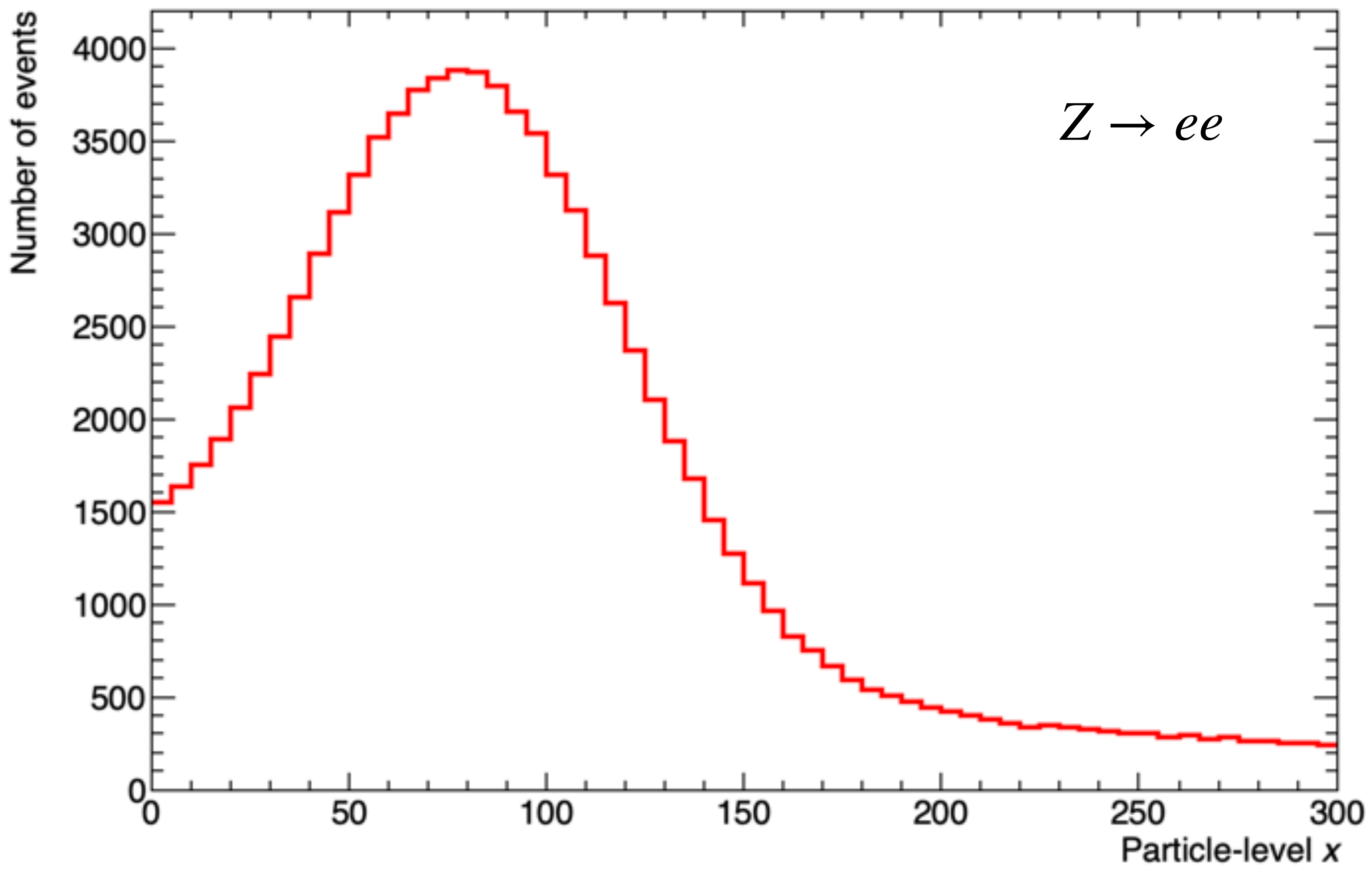
Using ML to weight events

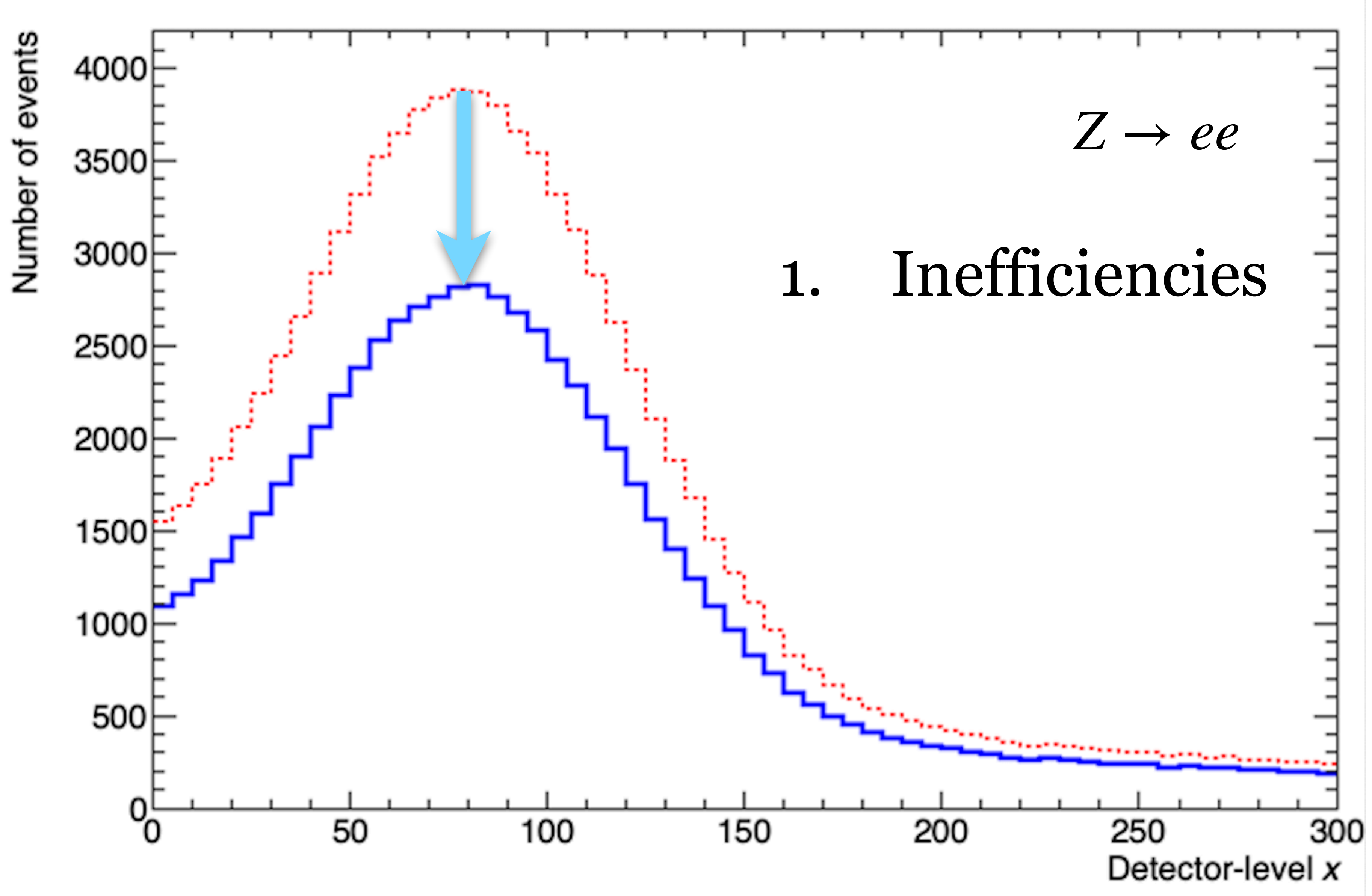
- Using ML classification to estimate the likelihood ratio, and use this as a weighting function has many relevant applications
- Early use/adoption were done by researchers at LHCb in 2015
  - In other fields ‘density ratio estimation’ has been used earlier.
- A few examples of applications in particle physics:
  - Neural networks for full phase-space reweighing and parameter tuning https://arxiv.org/abs/1907.08209
  - Neural resample for MC reweighing and uncertainty preservation https://arxiv.org/abs/2007.11586
  - Omnifold method to perform unfolded precision measurments …

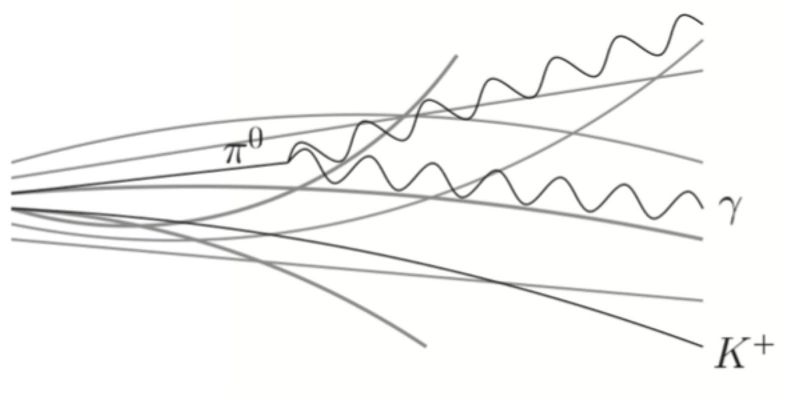
The Omnifold method

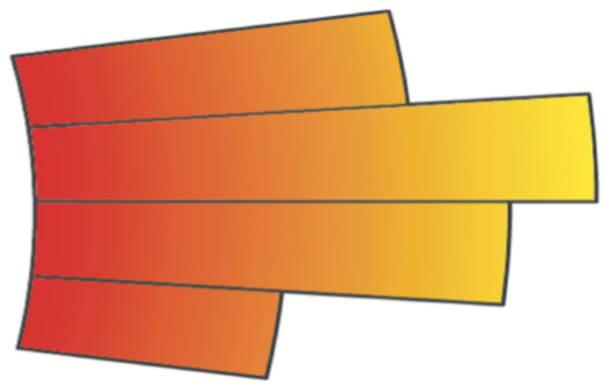
- The Omnifold method uses ML to perform unbinned, high-dimensional measurements
- This includes unfolding to the particle-level

Interaction with the detector, two major effects



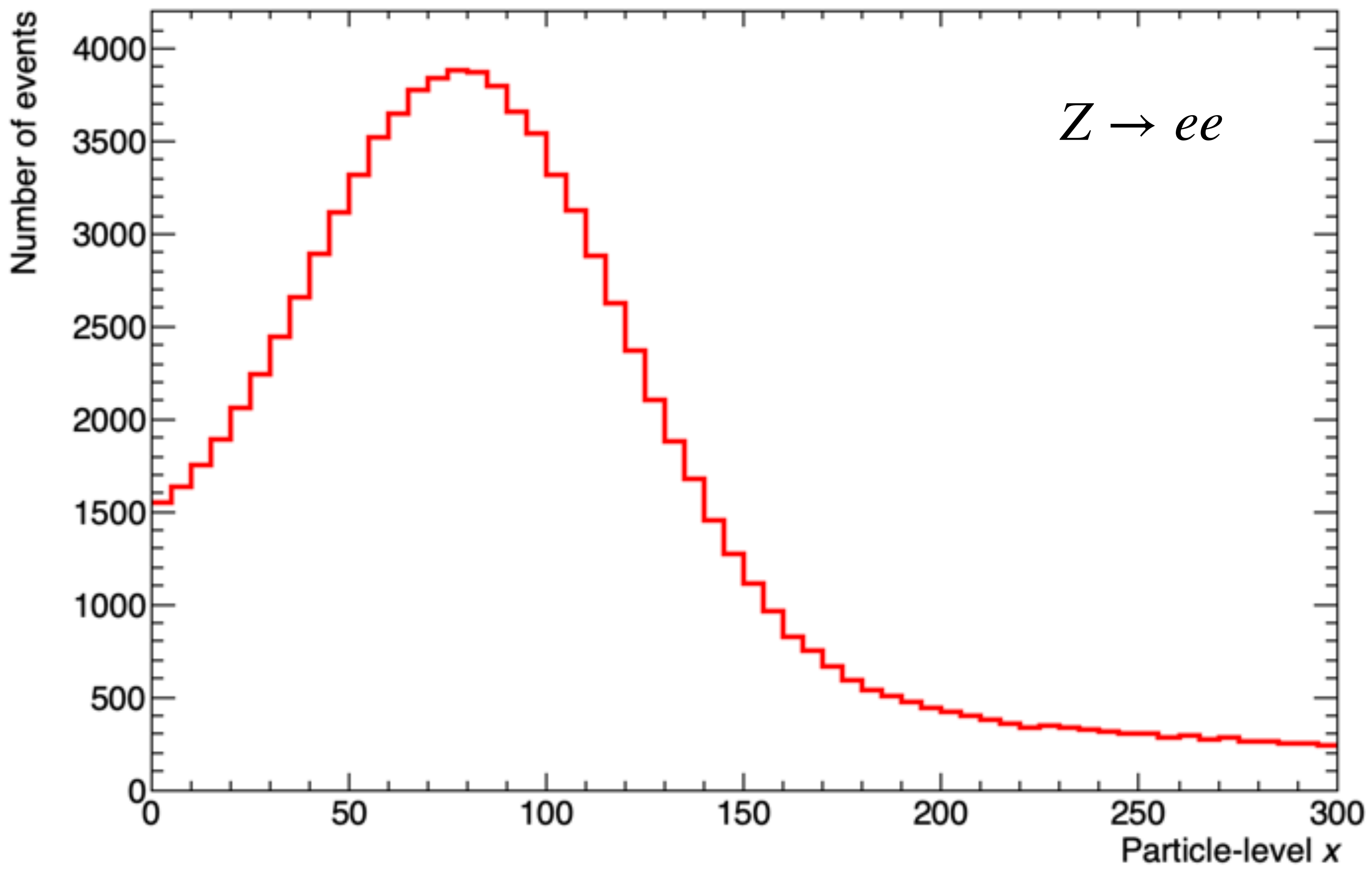


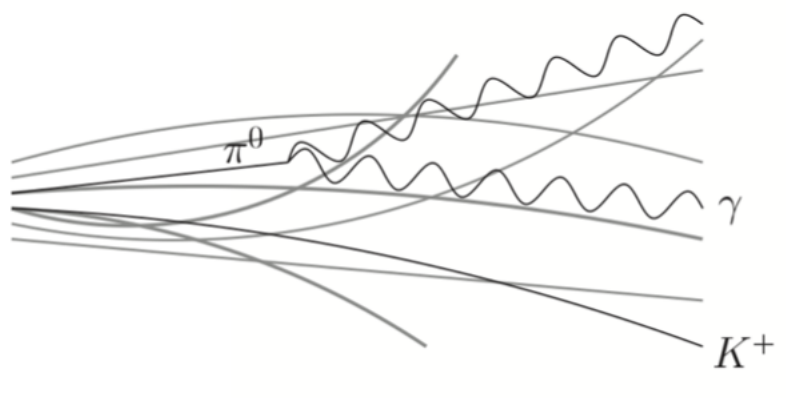




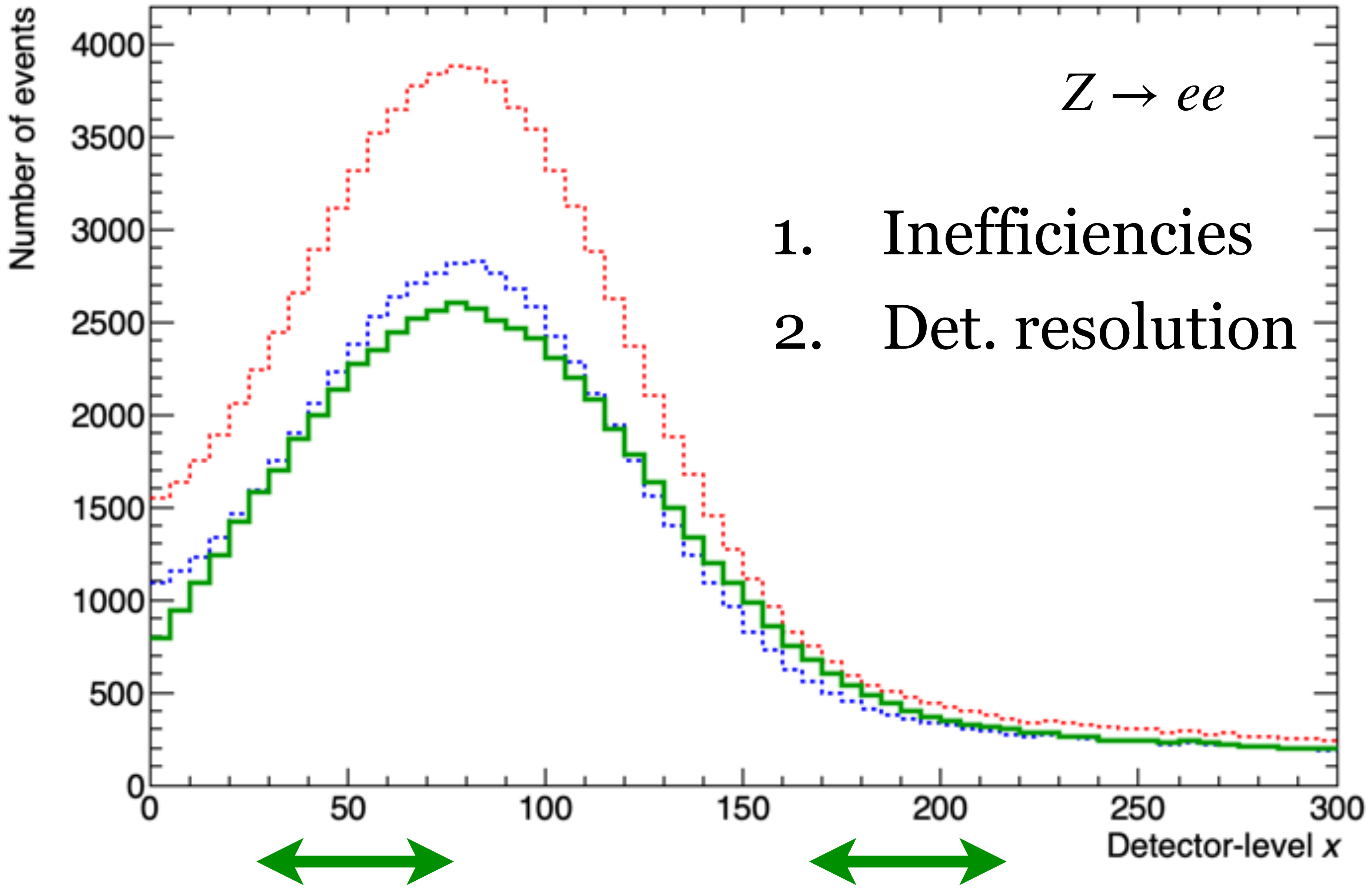
The Omnifold method

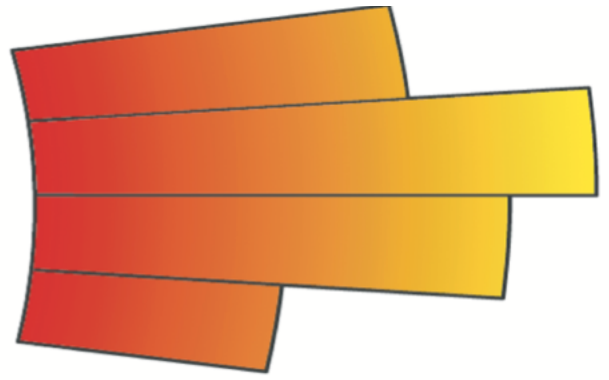
- The Omnifold method uses ML to perform unbinned, high-dimensional measurements
- This includes unfolding to the particle-level





Interaction with the detector, two major effects

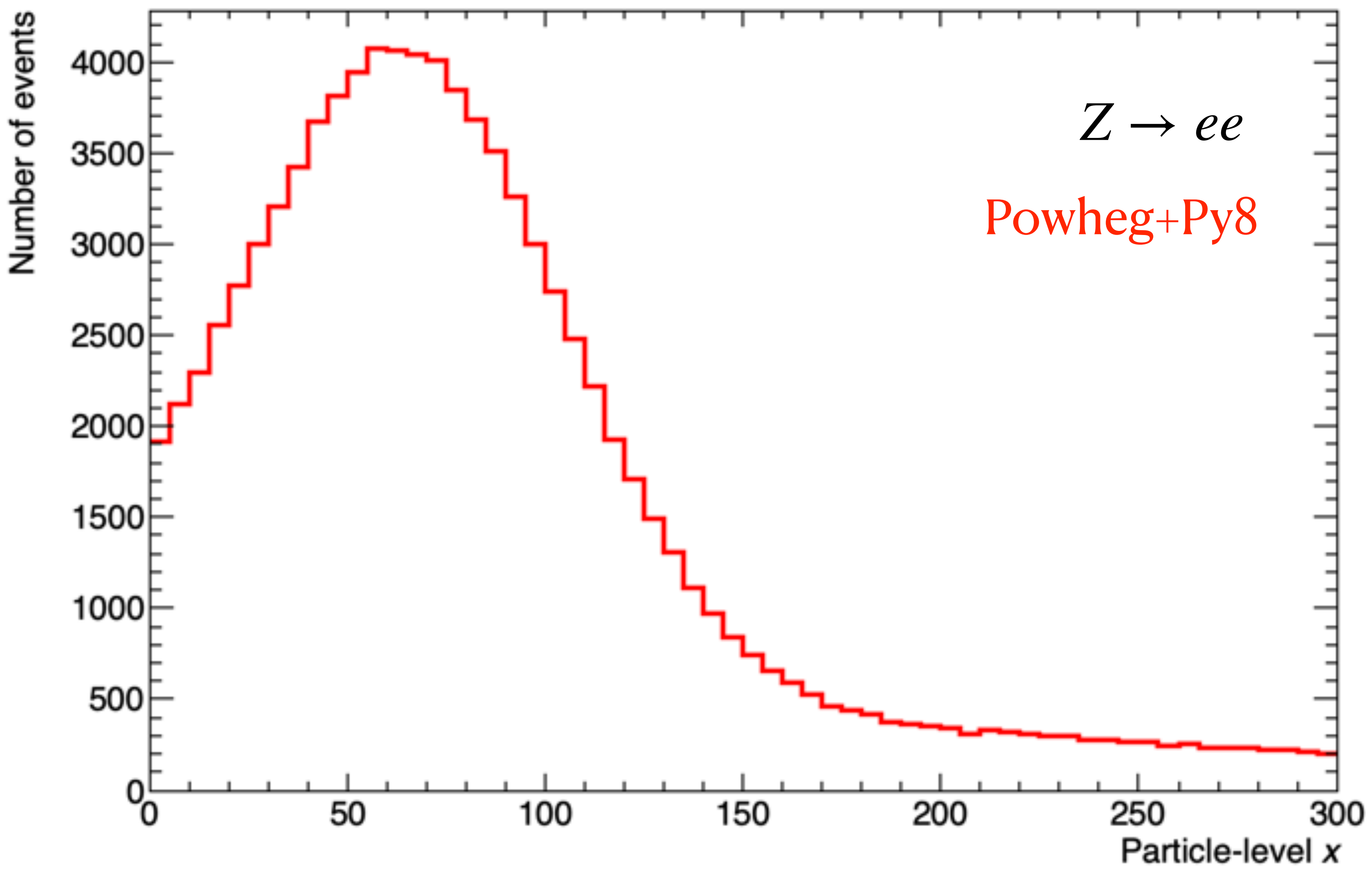


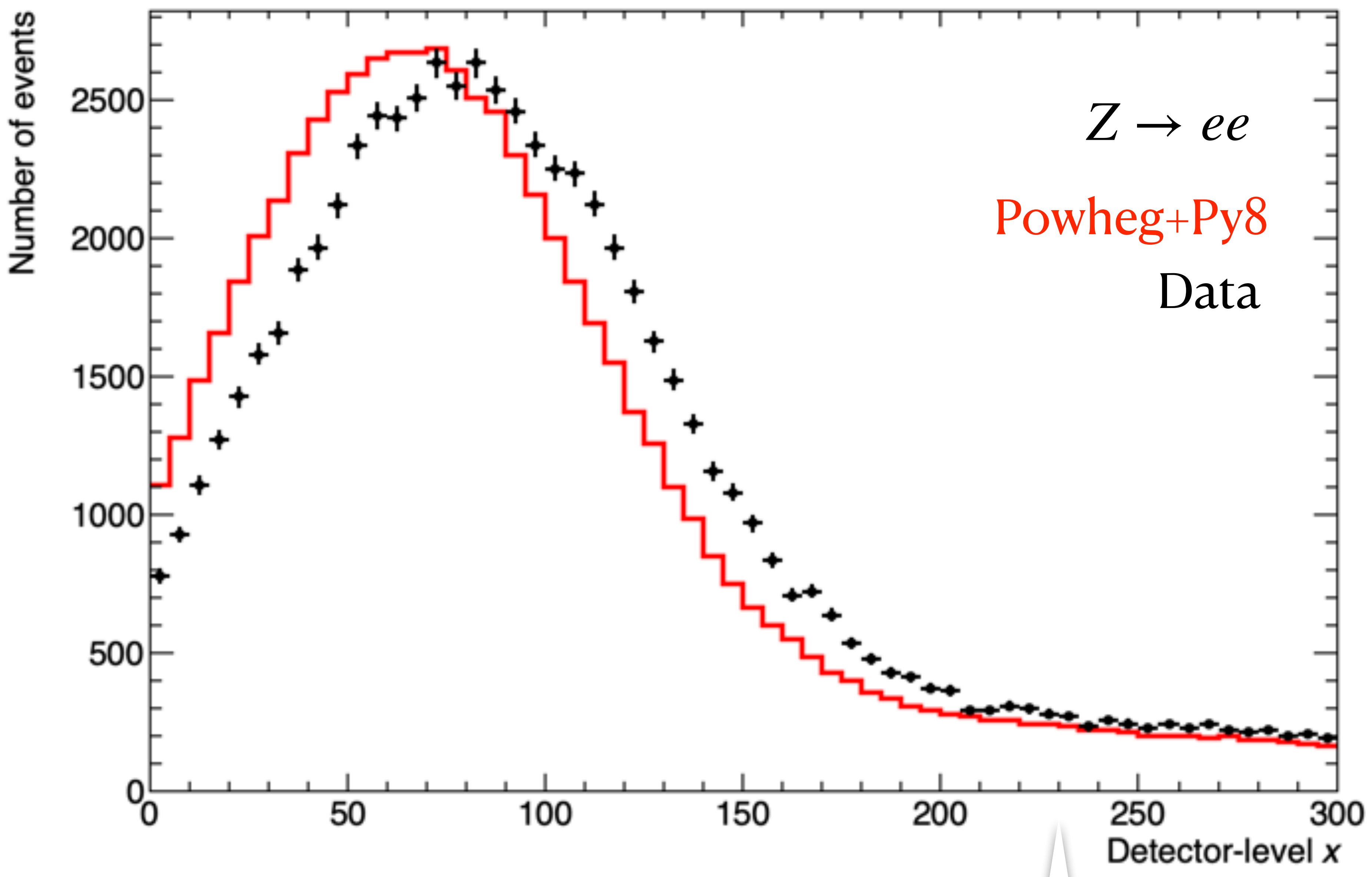


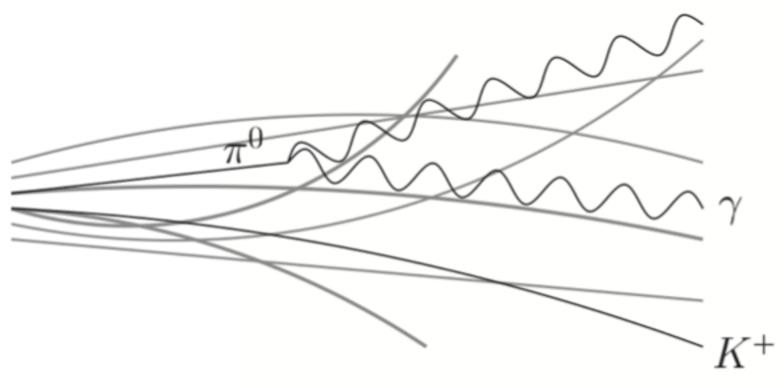
The Omnifold method

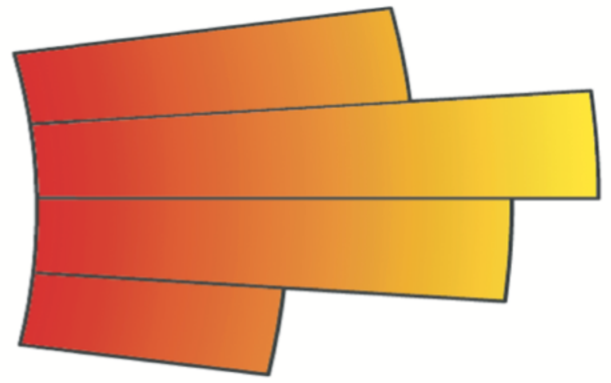
- The Omnifold method uses ML to perform unbinned, high-dimensional measurements
- This includes unfolding to the particle-level

OmniFold: A Method to Simultaneously Unfold All Observables
Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler
Phys. Rev. Lett. **124**, 182001 – Published 7 May 2020







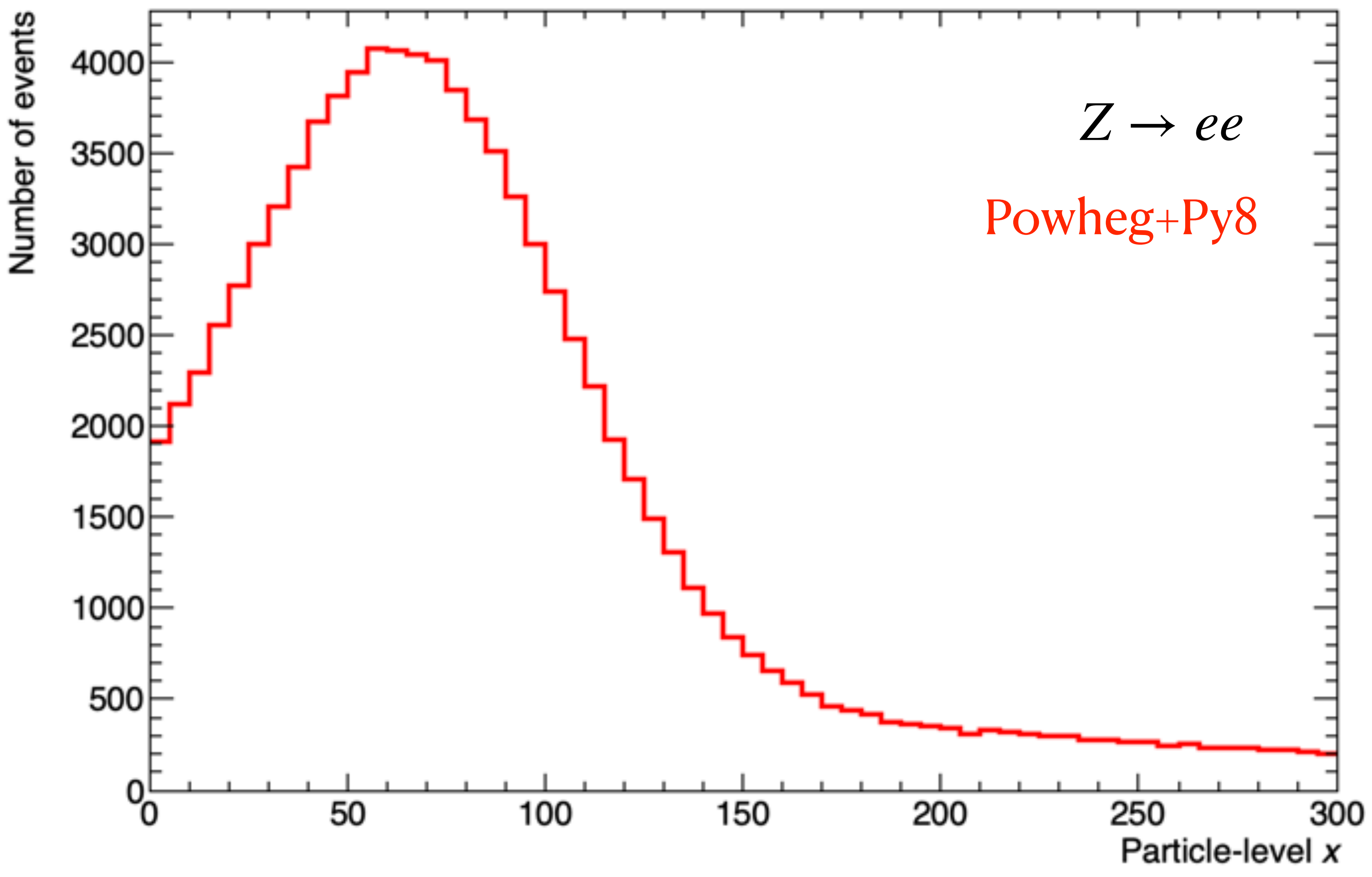


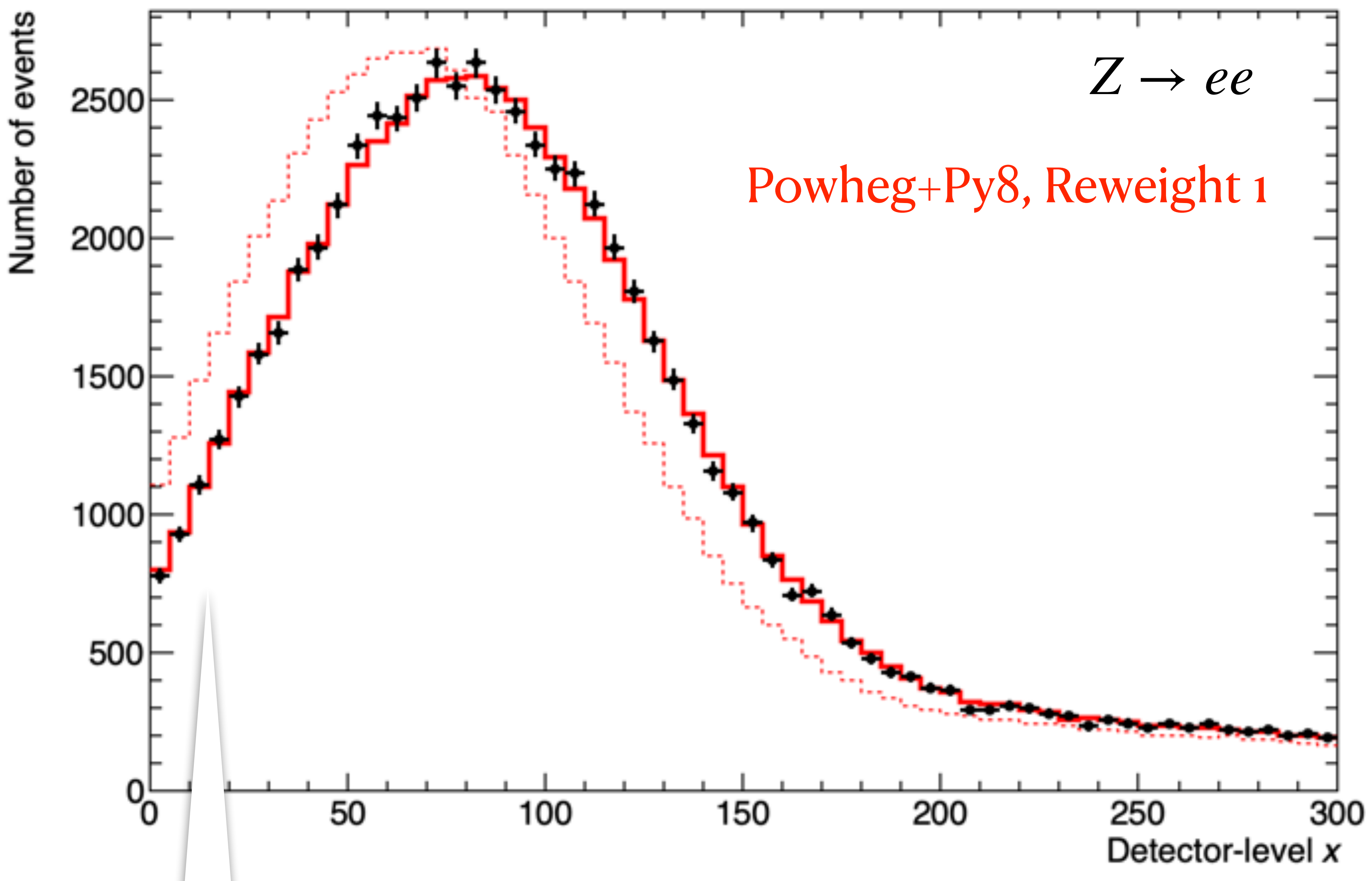
1. Train NN with data as signal, MC bkg
Use to reweight!

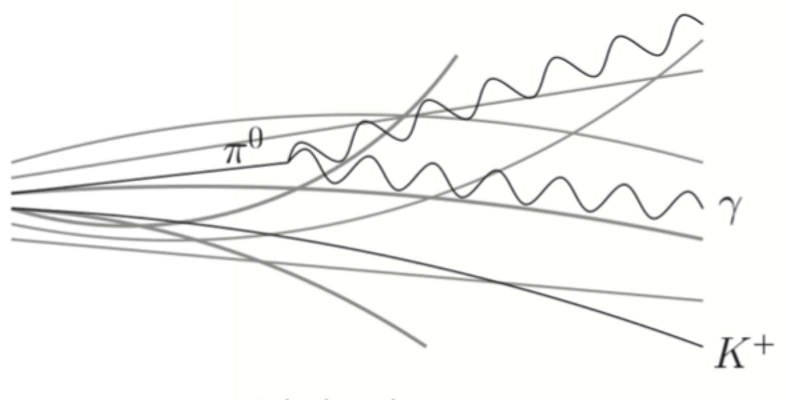
The Omnifold method

- The Omnifold method uses ML to perform unbinned, high-dimensional measurements
- This includes unfolding to the particle-level

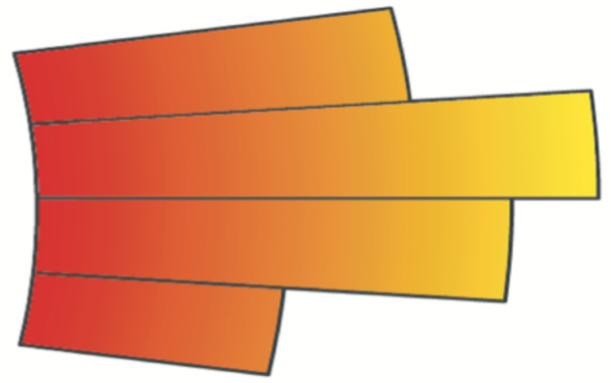
OmniFold: A Method to Simultaneously Unfold All Observables
Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler
Phys. Rev. Lett. **124**, 182001 – Published 7 May 2020







2. Each simulated event has obtained a weight. Propagate this to the partilcle level distribution

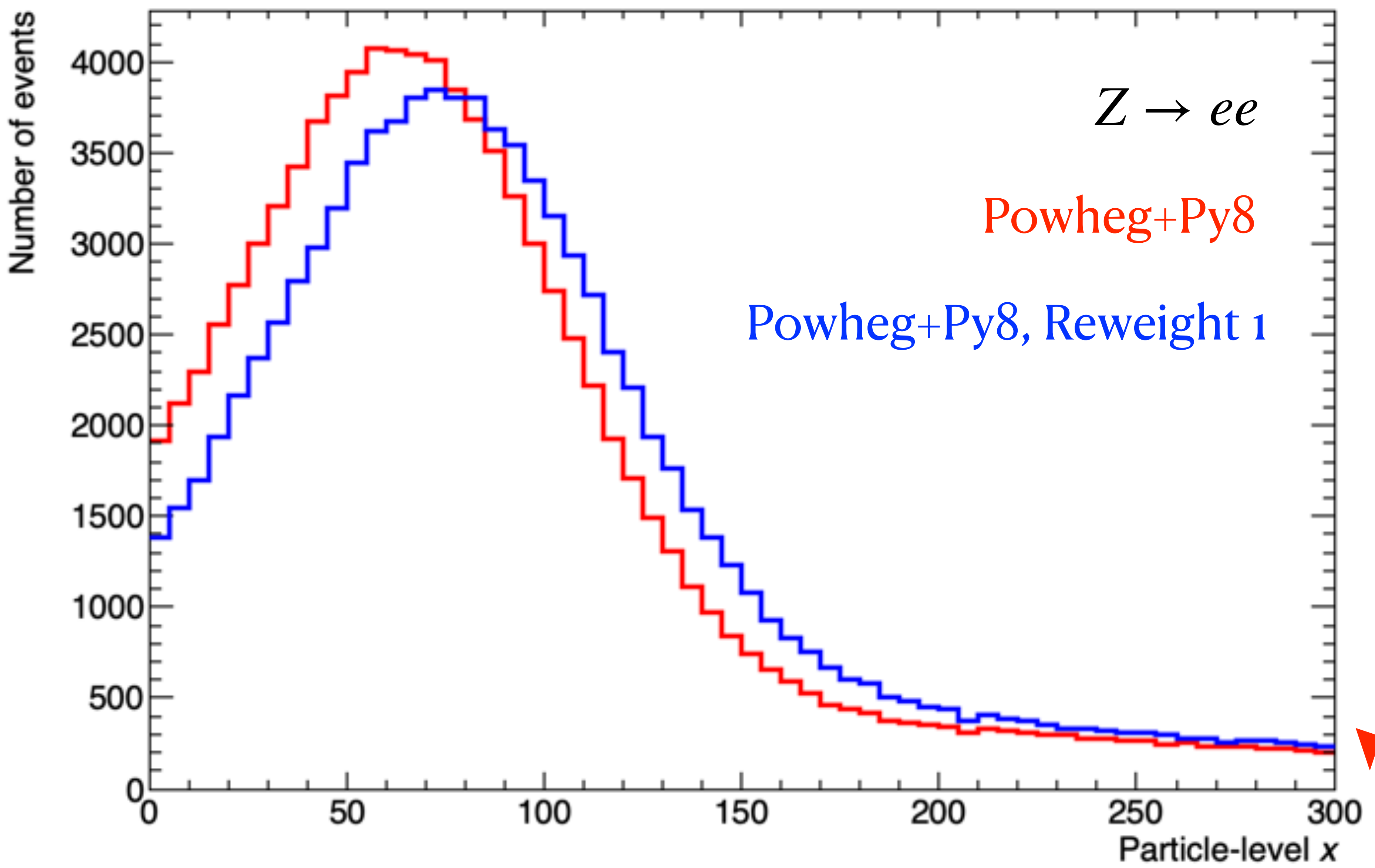


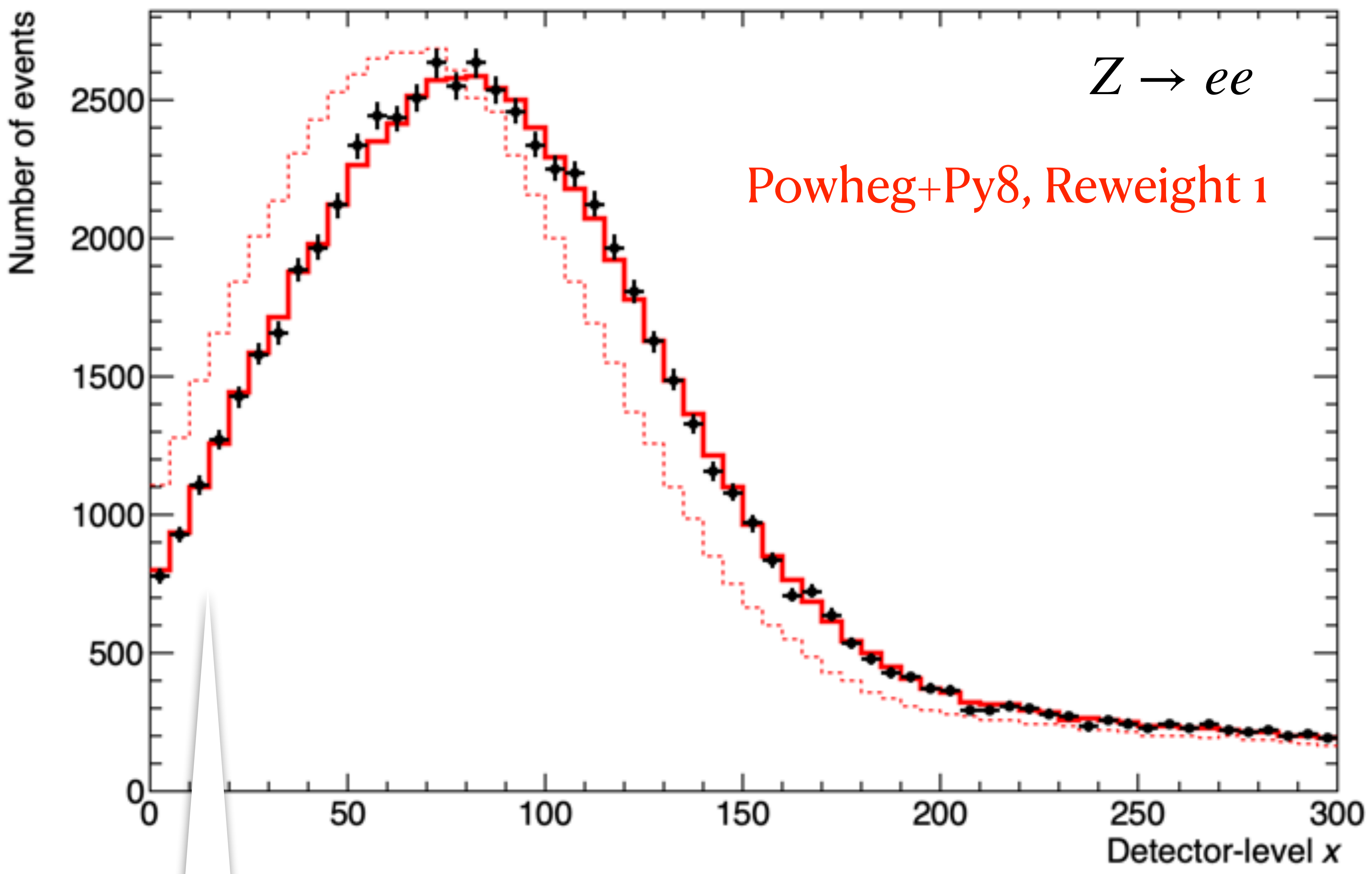
1. Train NN with data as signal, MC bkg
Use to reweight!

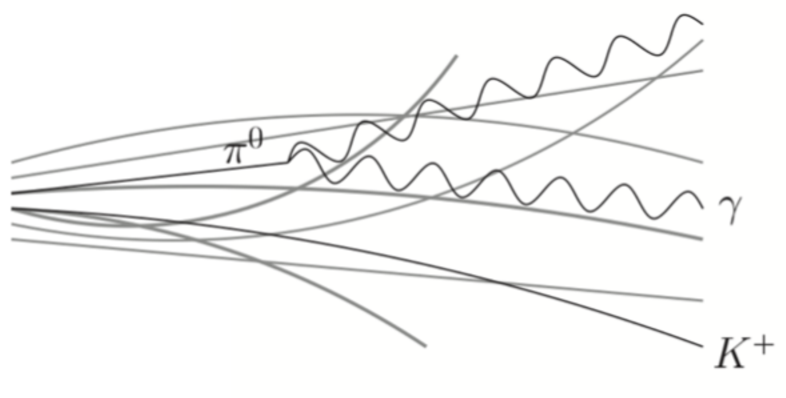
The Omnifold method

- The Omnifold method uses ML to perform unbinned, high-dimensional measurements
- This includes unfolding to the particle-level

OmniFold: A Method to Simultaneously Unfold All Observables
Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler
Phys. Rev. Lett. **124**, 182001 – Published 7 May 2020







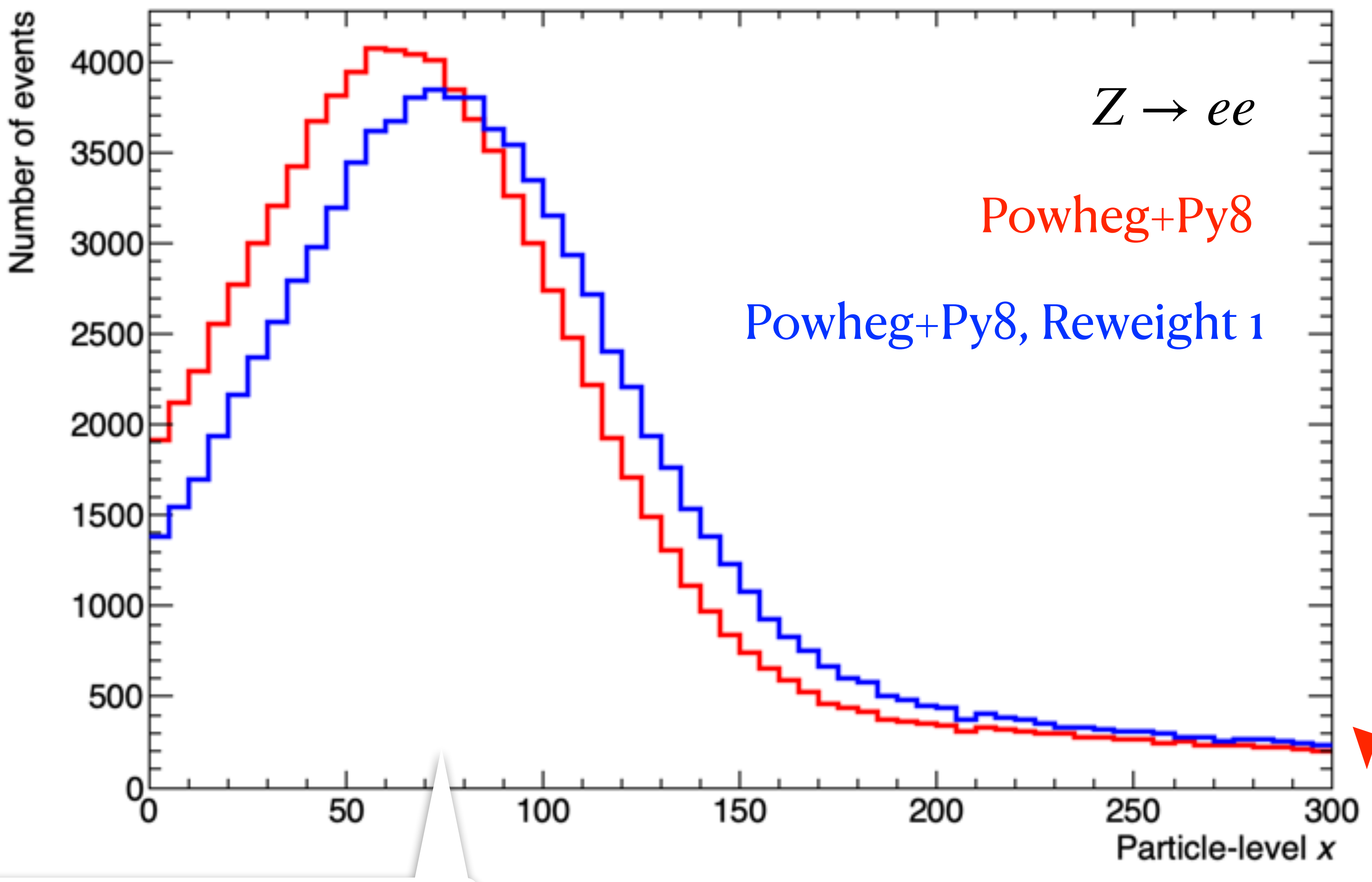
2. Each simulated event has obtained a weight. Propagate this to the partilcle level distribution

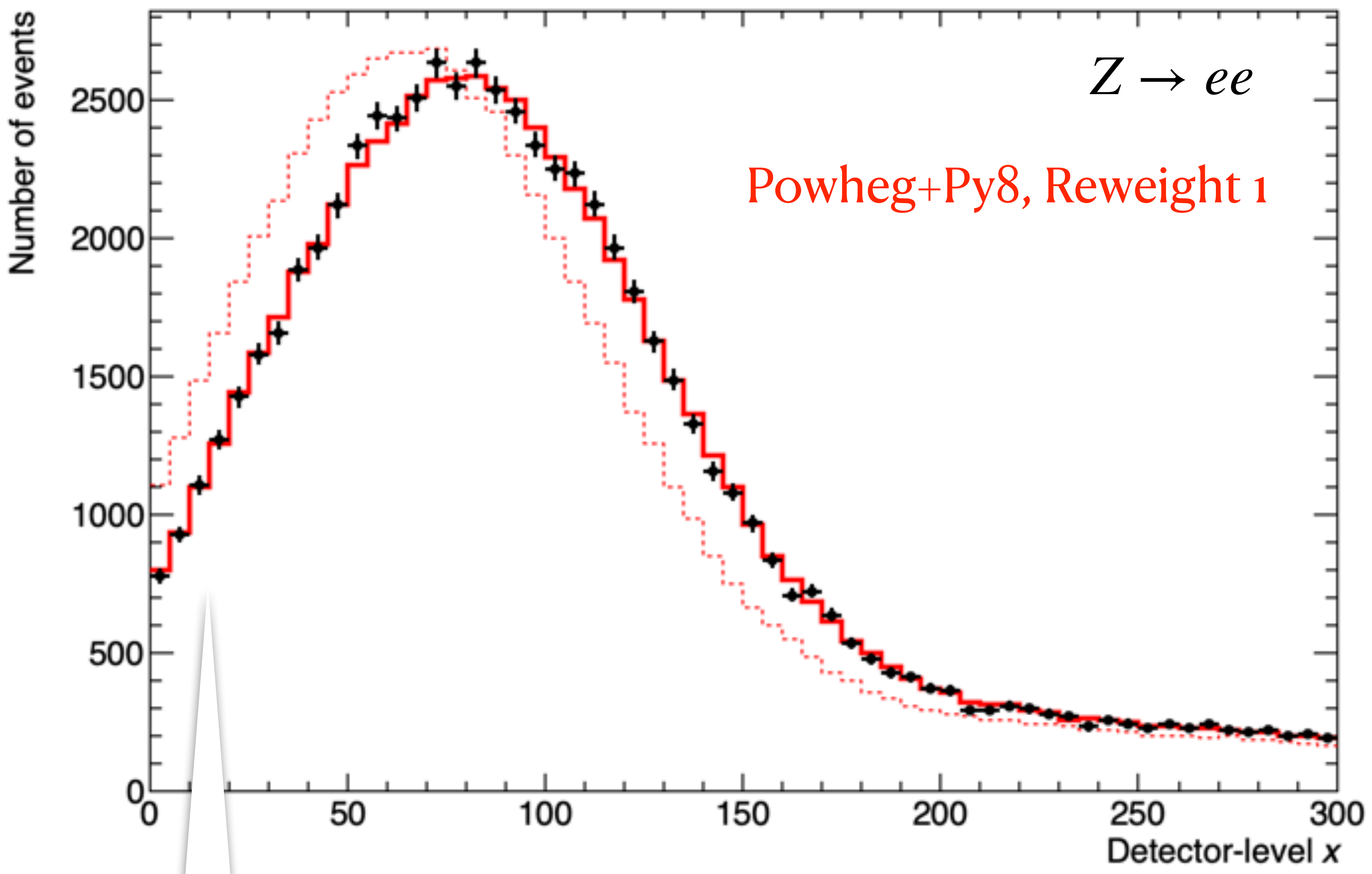
1. Train NN with data as signal, MC bkg
Use to reweight!

The Omnifold method

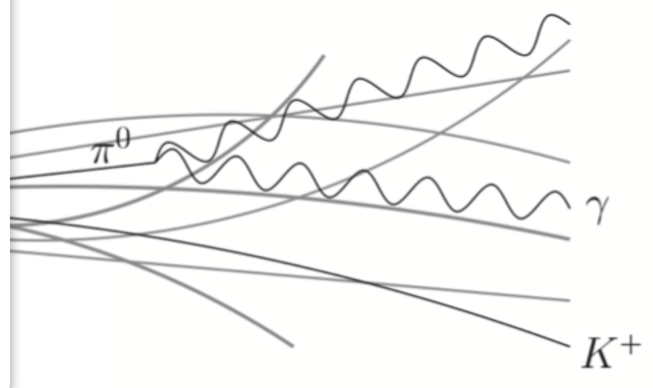
- The Omnifold method uses ML to perform unbinned, high-dimensional measurements
- This includes unfolding to the particle-level

OmniFold: A Method to Simultaneously Unfold All Observables
Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler
Phys. Rev. Lett. **124**, 182001 – Published 7 May 2020

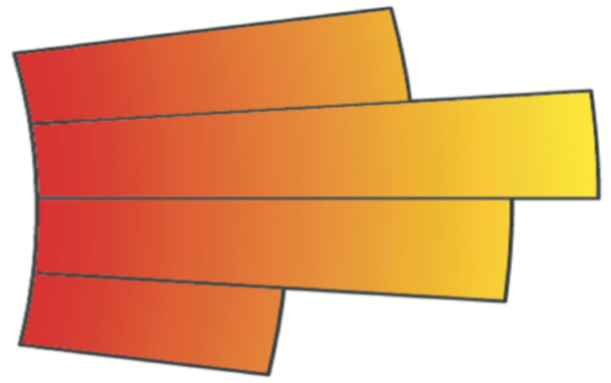




3. Train a new network using using blue as signal, red background (MC-MC). Use to reweight red!



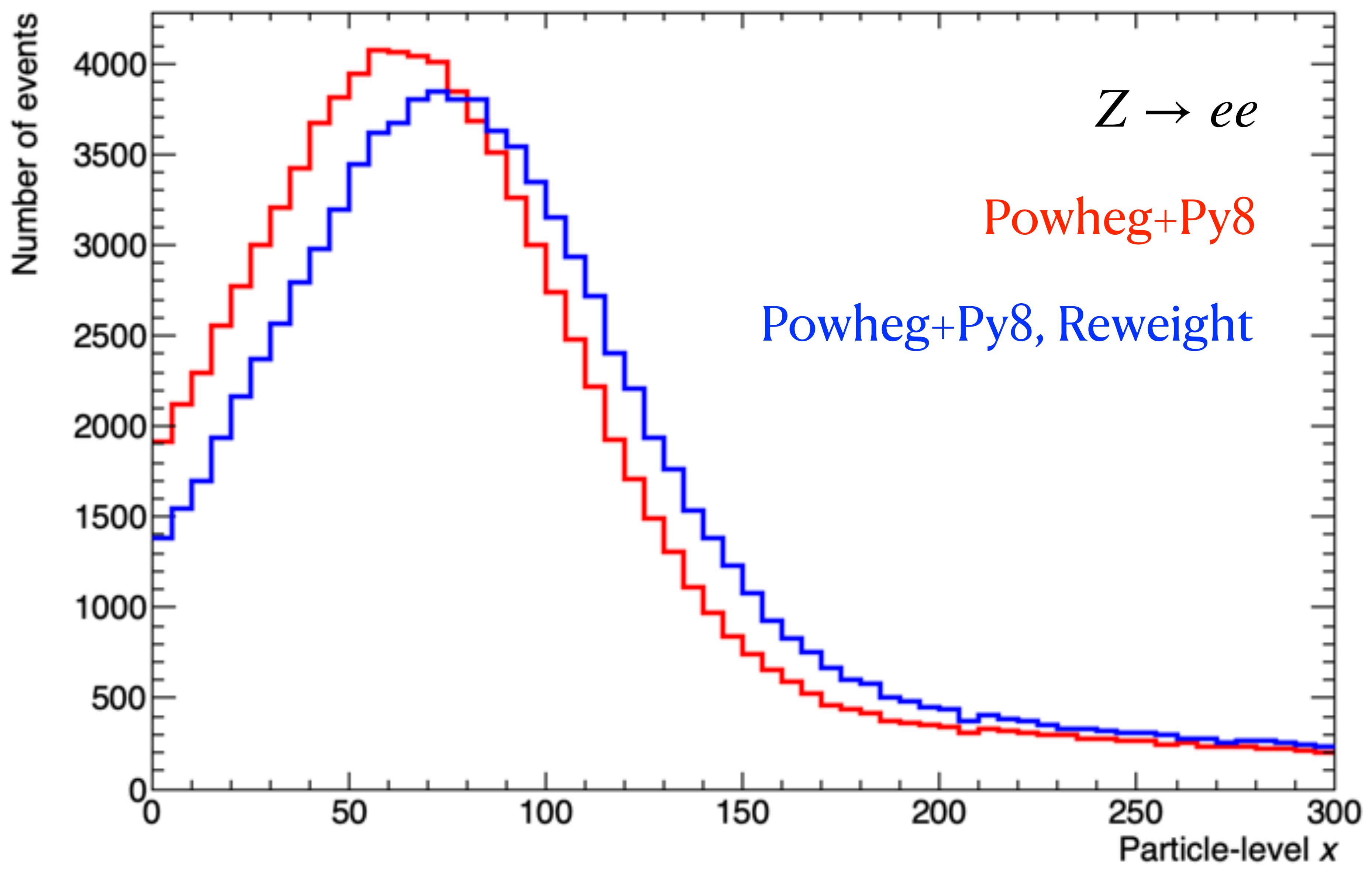
2. Each simulated event has obtained a weight. Propagate this to the partilcle level distribution

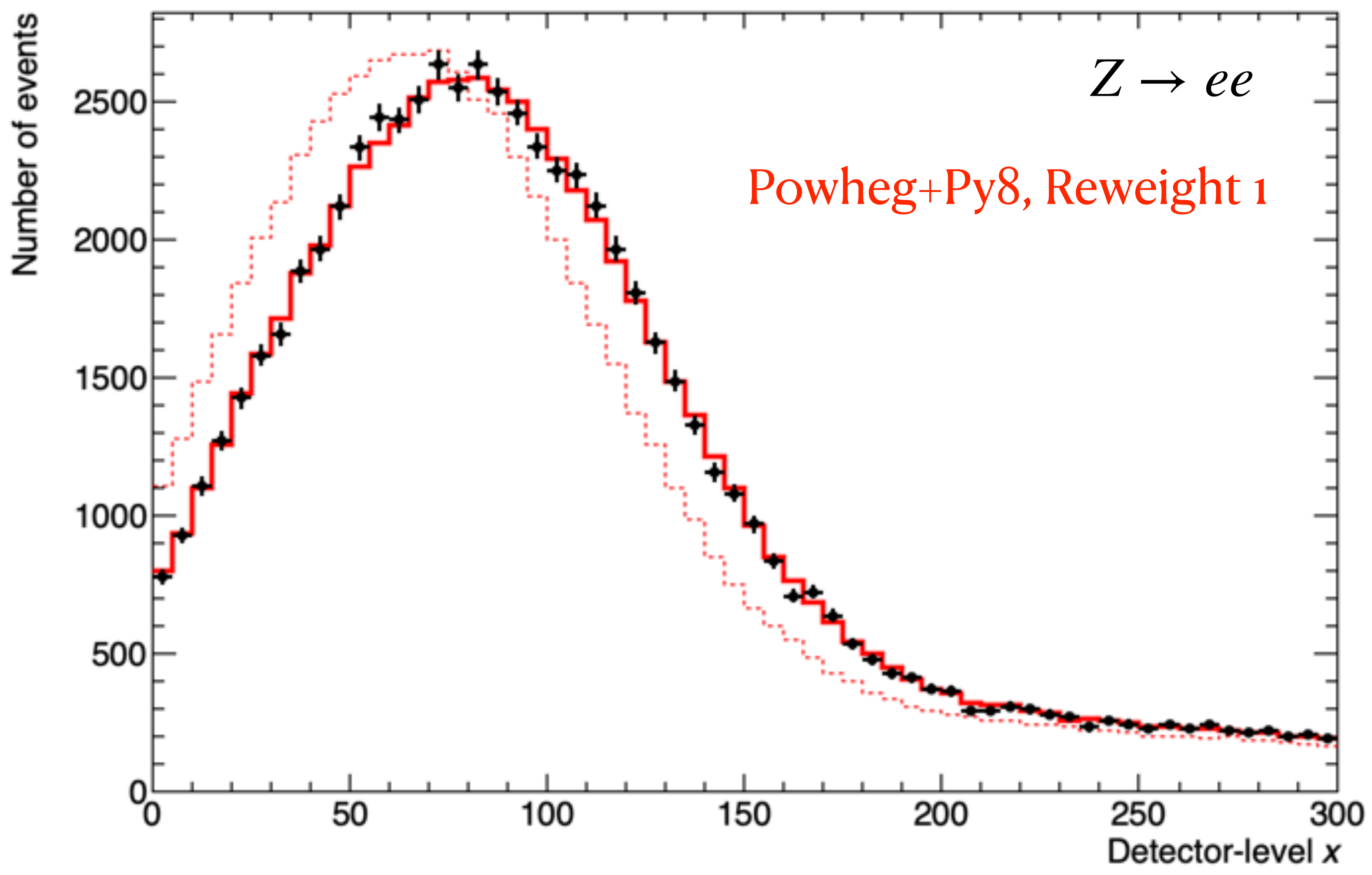


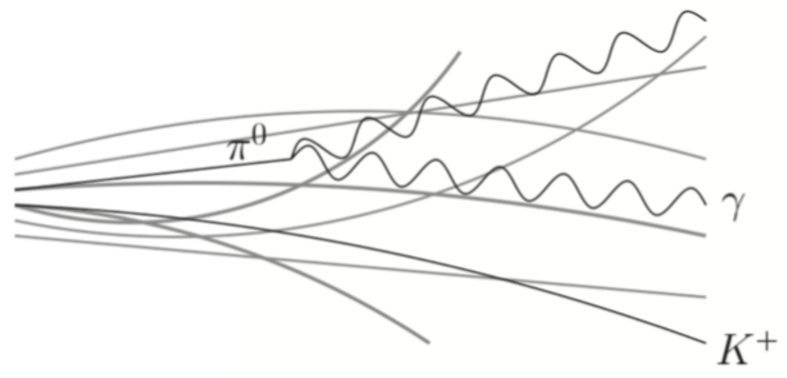
1. Train NN with data as signal, MC bkg Use to reweight!

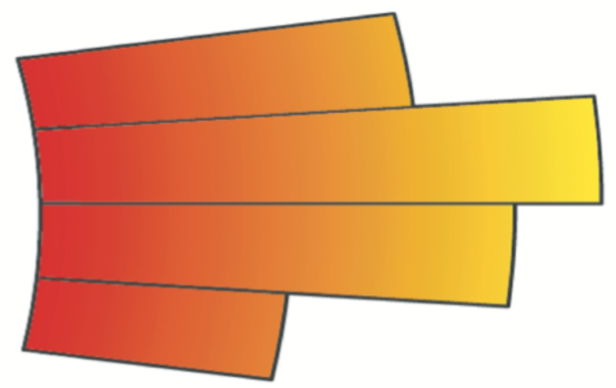
The Omnifold method

- This method interactively reweighs distributions:
  - Match data, then update prior (particle-level distribution)
- Stable solution found after a few iterations (typically 2-5)
- Identical to Iterative Bayesian Unfolding when binned input is used

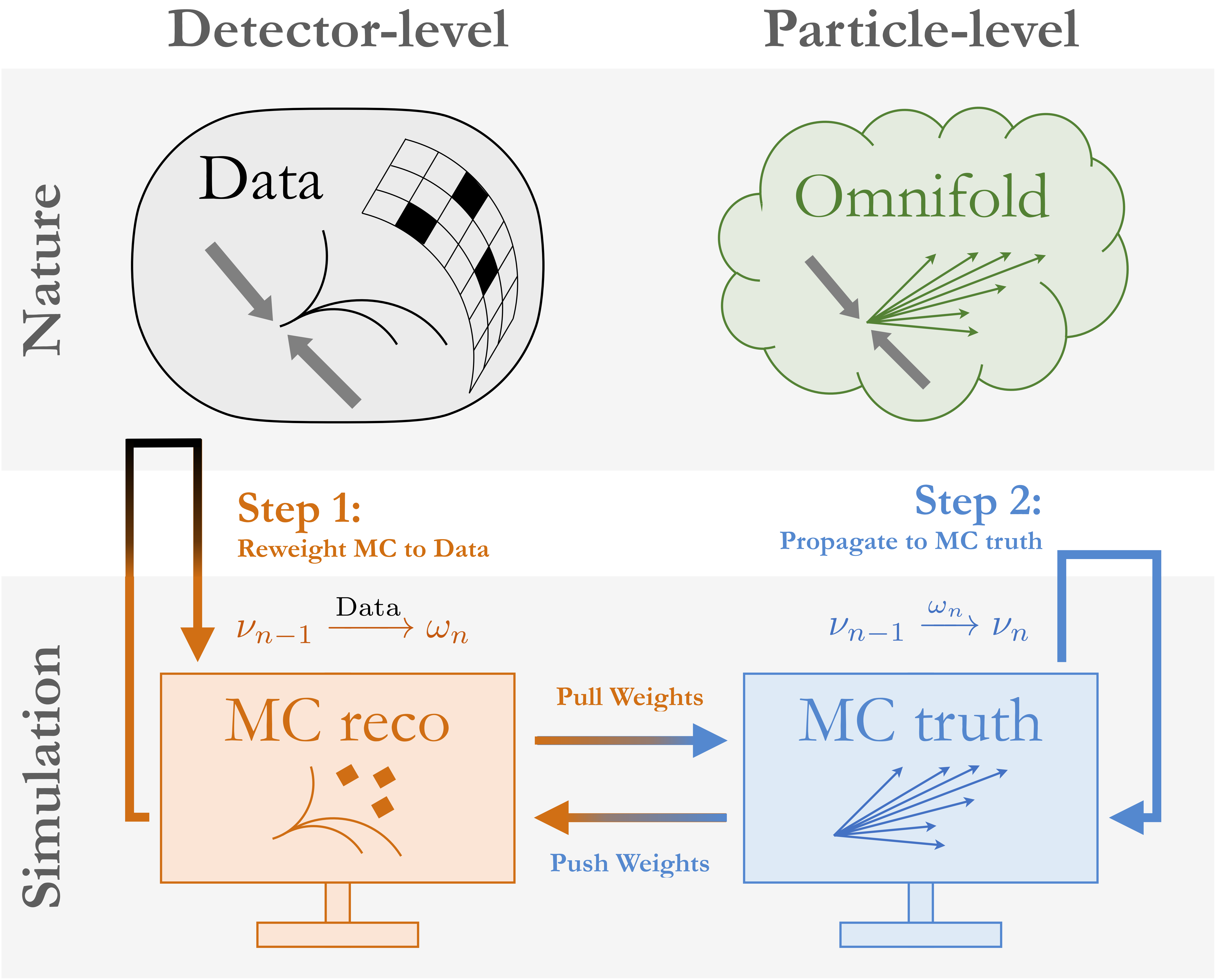








The Omnifold method



Adjust MC reco to match data $\rightarrow \omega_{\text{NN}}(\vec{x}_{\text{reco}})$

Propagate to particle-level

Adjust particle-level to match this change $\rightarrow \nu_{\text{NN}}(\vec{x})$

Propagate to reco level

Repeat

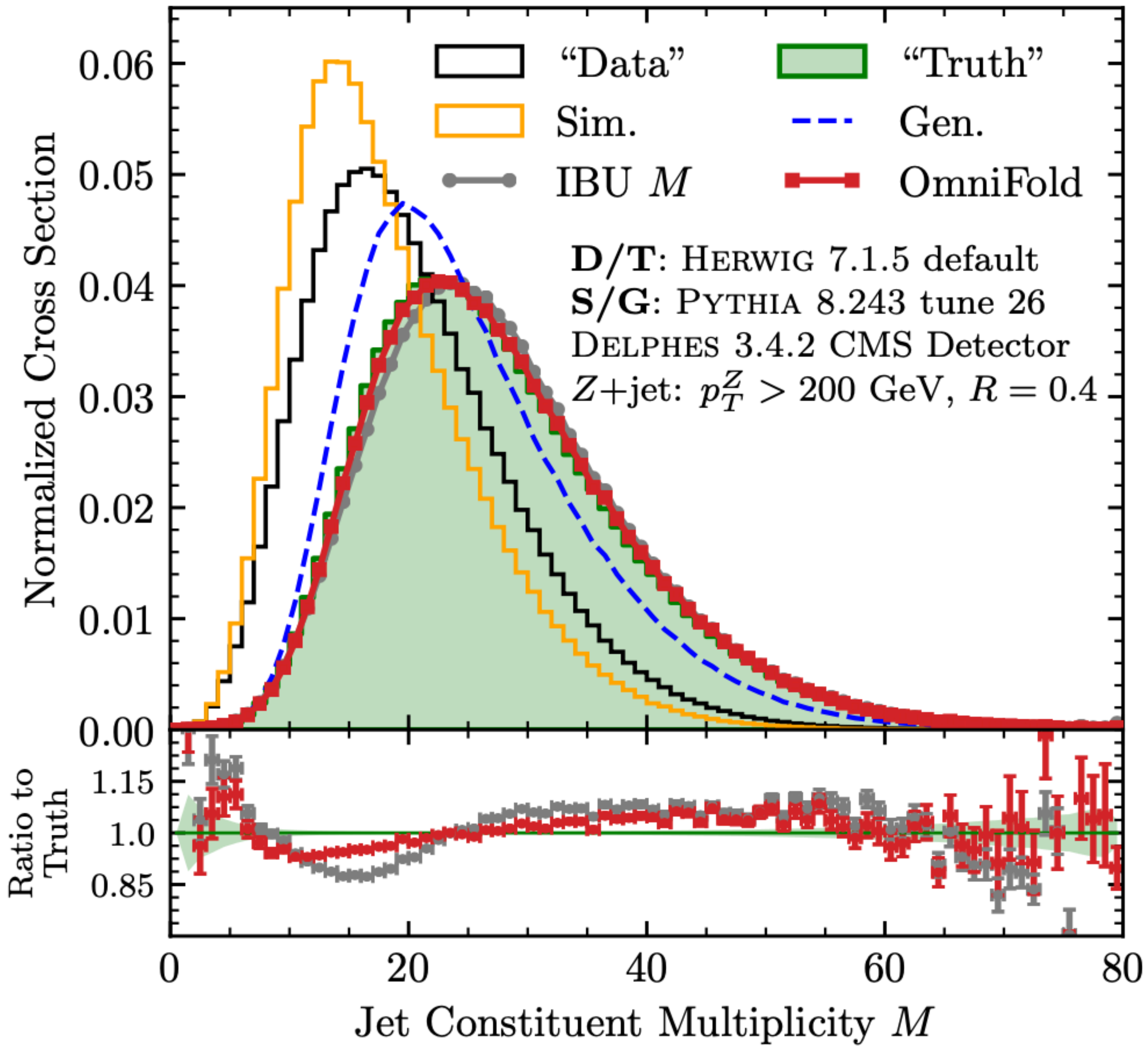
The Omnifold method

- Method announced 2020 with proof-of-principle results based on simulation

OmniFold: A Method to Simultaneously Unfold All Observables

Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler

Phys. Rev. Lett. **124**, 182001 – Published 7 May 2020



- The output is a weighing function that applies to simulated events (e.g. Powheg+Pythia)
- The function takes only particle-level quantities as input (no need for detector simulation)
- Weighing MC events makes them 'become unfolded data'

Output of unbinned measurements

Publishing unbinned differential cross section results

Miguel Arratia[1,2], Anja Butter[3], Mario Campanelli[4], Vincent Croft[5], Dag Gillberg[6], Aishik Ghosh[7,8], Kristin Lohwasser[9], Bogdan Malaescu[10], Vinicius Mikuni[11], Benjamin Nachman[8,12]

- The measured data needs to be made public
- In principle, one could publish just the reweighing functions
  - however, requires that MC events are generated **exactly** in the same way as analysis
- Safer to provide MC events, with the associated weights
- Each event needs to contain
  - All features used in measurement $\vec{x}$
  - Nominal weight that adjust it (to become unfolded data) ν
  - A long list of additional weights corresponding to uncertainties
    - Data statistical uncertainty (we propose ~50)
    - MC statistics uncertainty (we propose ~25)
    - Systematic uncertainties ($\mathcal{O}(100)$)
- Alternative MC sample with all variables and MC stat weights

Statistical uncertainties evaluated using bootstrapping

Systematic uncertainties evaluated using established methods (perturbation of input sample)

Output of unbinned measurements

Publishing unbinned differential cross section results

Miguel Arratia[1,2], Anja Butter[3], Mario Campanelli[4], Vincent Croft[5], Dag Gillberg[6], Aishik Ghosh[7,8], Kristin Lohwasser[9], Bogdan Malaescu[10], Vinicius Mikuni[11], Benjamin Nachman[8,12]

- Plan: produce large n-tuple (several GB) with all weights and variables
  - Need to make it **public** on some site. Zendo is an option.
  - Should keep link from HepData (best if HepData would have capability)
  - Also need to keep providing associated Rivet routine
- Further need to provide **user guide**
  - Idea is to provide a Python notebook that shows how to produce results
  - E.g. how to loop over events to create histograms with all associated uncertainties → unfolded measurements
  - Basic stat. guidelines, e.g. choice of binning (not too narrow, can get empty bins)
  - Caveats and validity: need to be clear with which observables and applications have been validated

Flavours of Omnifold

- UNIFOLD
  - Measure only one variable at the time.
  - Unbinned version of Iterative Bayesian Unfolding
- MULTIFOLD
  - Measure a fixed set of variables simultaneously and unbinned
  - E.g. $p_{\mathrm{T}}^{\ell 1}, p_{\mathrm{T}}^{\ell 2}, \eta^{\ell 2}, \eta^{\ell 1}, p_{\mathrm{T}}^{j1}, \ p_{\mathrm{T}}^{j2}$
  - Note that you can construct measurements of other observables afterwards. E.g. $\Delta\eta_{\ell\ell} = \eta^{\ell 1} - \eta^{\ell 2}$
- (Full) OMNIFOLD
  - Measure a variable-length set of variables (simultaneously and unbinned)
  - For example, the momenta (p_{T}, η, ϕ) of all charged particles in an event (One event might have 50 charged particles, another 150)

Full Omnifold

- Procedure is the same, i.e. reweight by $f(\vec{x}) / (1 - f(\vec{x}))$, just the length of $\vec{x}$ varies from event to event
- Possible with particle flow networks

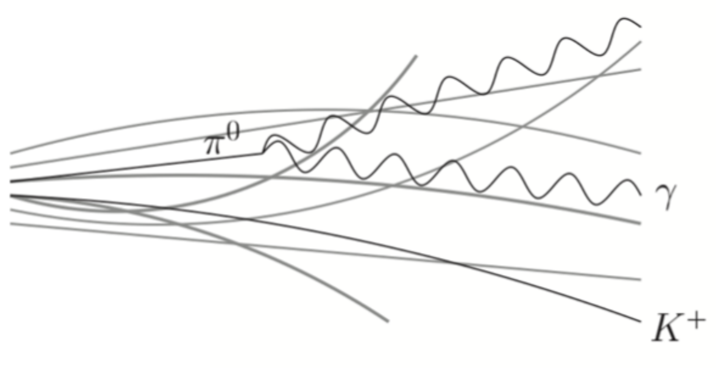
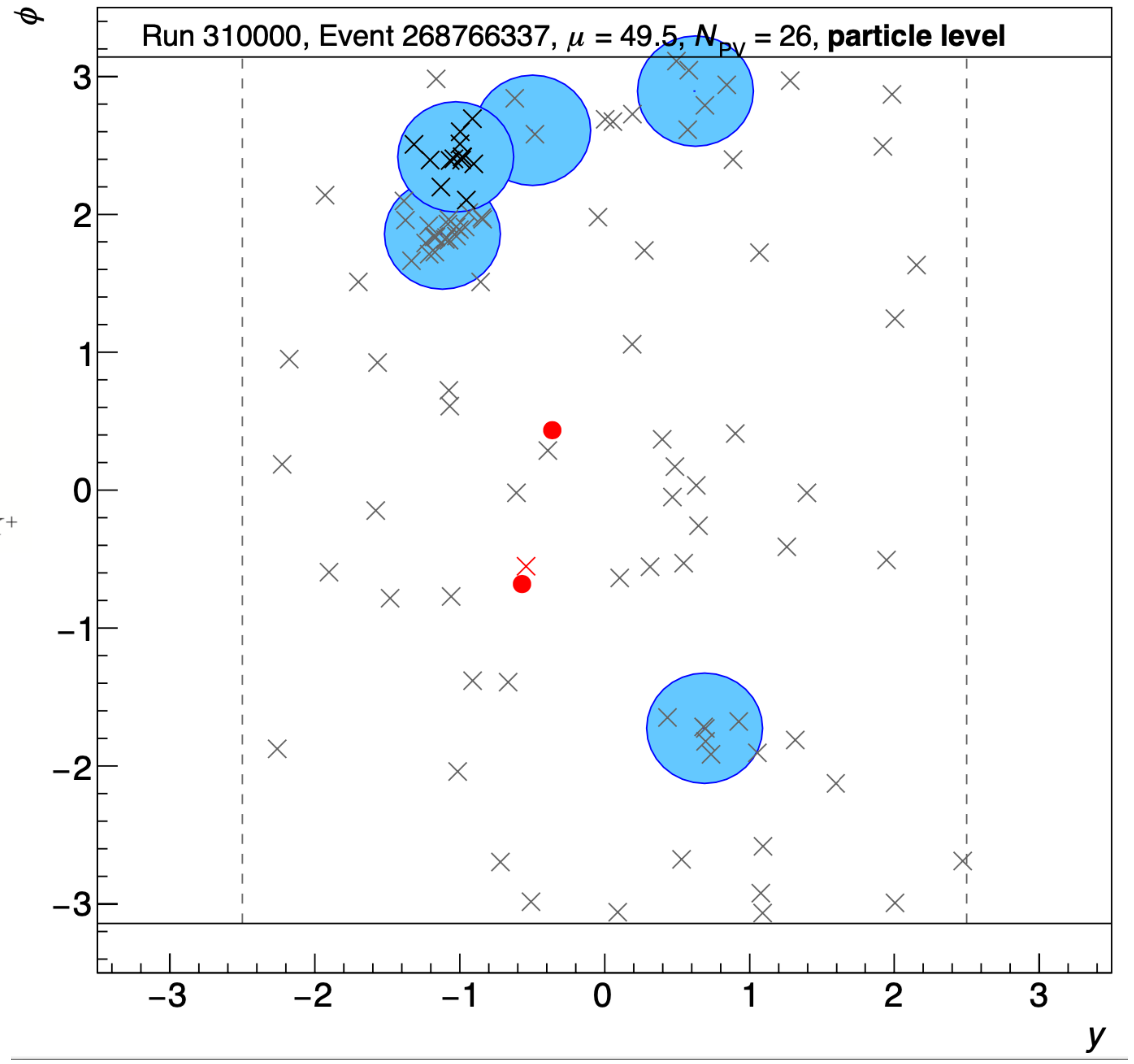
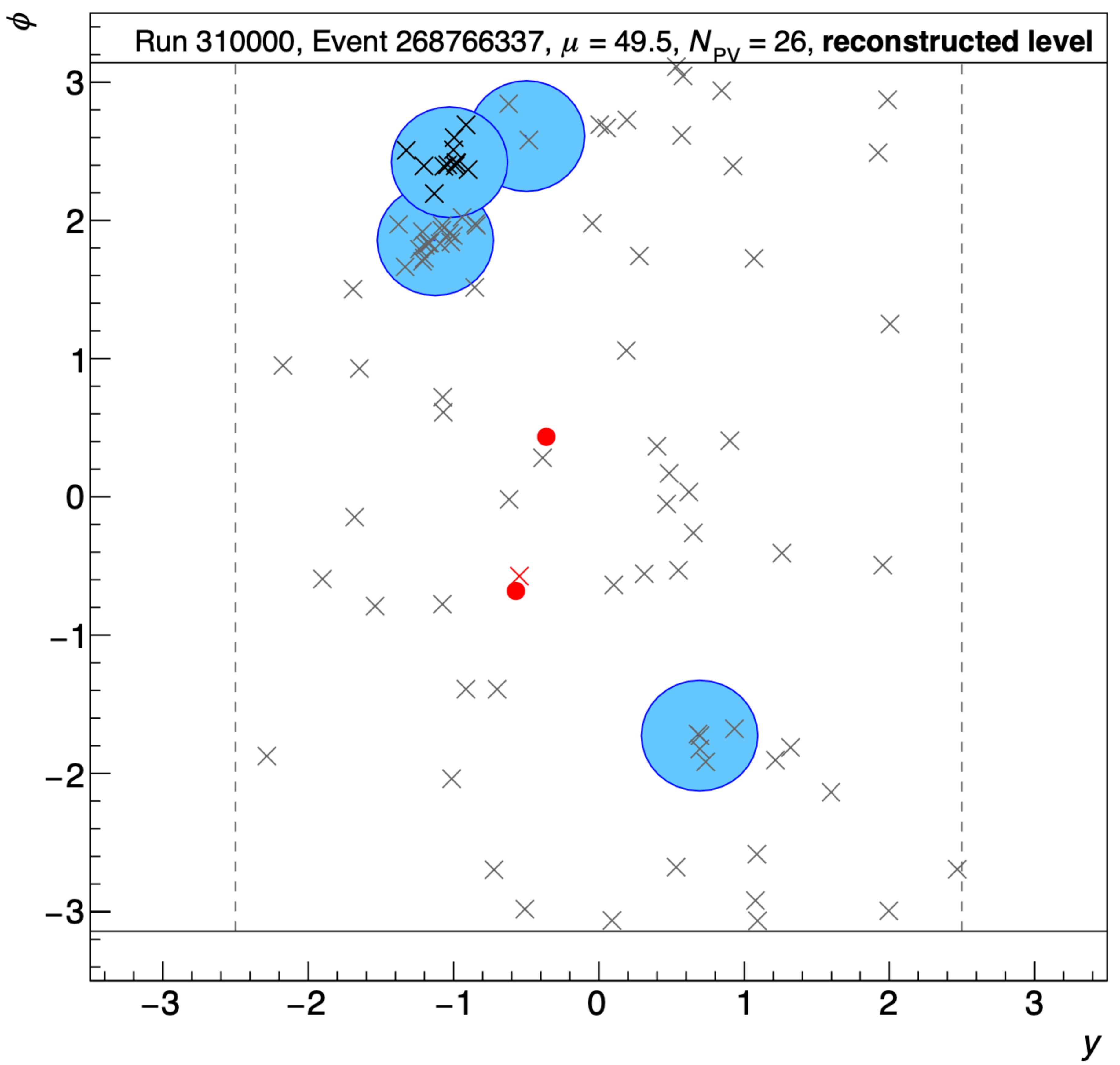


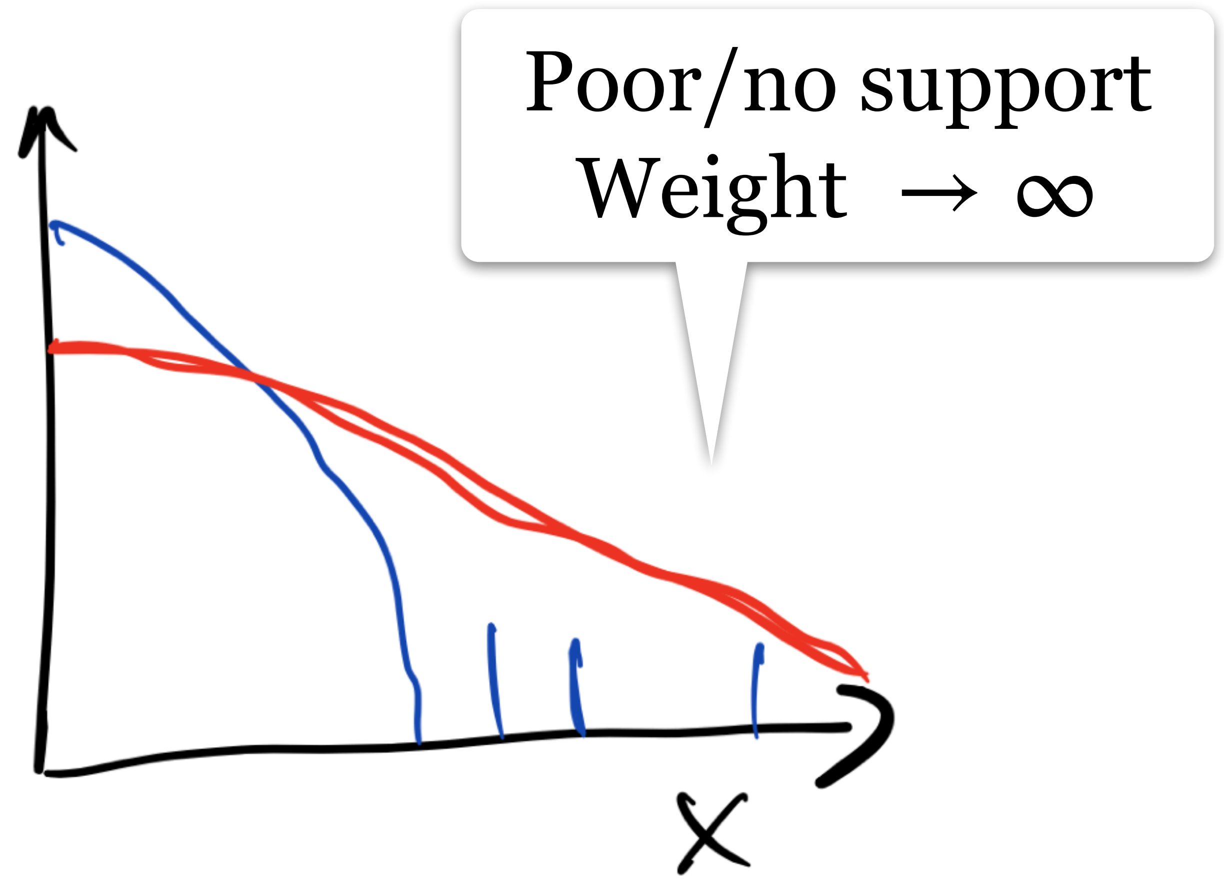
Illustration of
All changed particles
Produced in high pT
$Z \to \mu\mu$ events



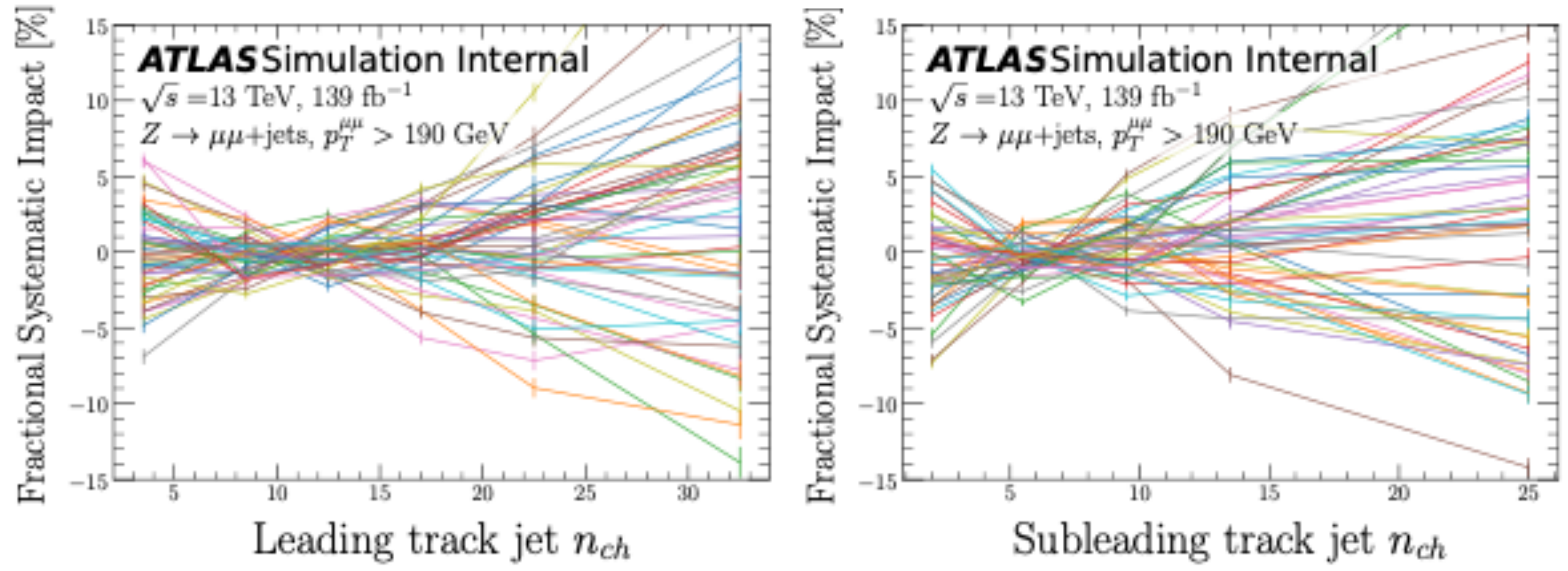


Shortcomings and challenges

- Wrap-around effect
  - Network gets confused by discontinuity in ϕ. It assumes smooth functions. Solved by letting network used $\sin(\phi)$ and $\cos(\phi)$.
- Insufficient support across full phase space
  - If we have reigns of phase space with too few initial MC events, the reweight will be too large (purity $f(\vec{x})$ too low, therefore $w(\vec{x}) = f/(1-f)$ unstable



- Instabilities of the network
  - Networks (Keras Tensorflow) initialized with random seed. Quickly finds solution. But different dep. on seed $\rightarrow$per-event instabilities
  - Hyperparameter otimization, and ensembling

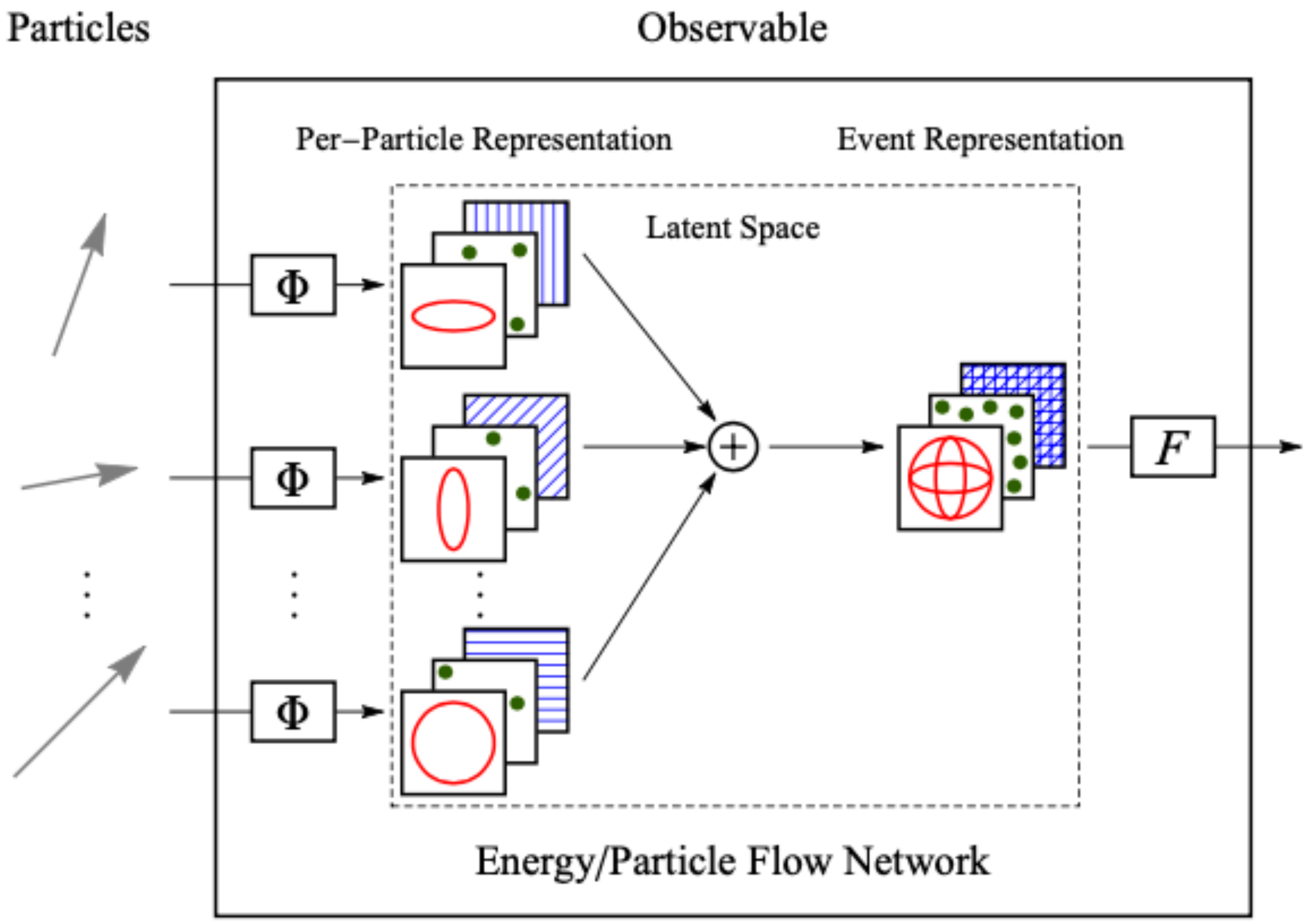


Summary

- Rapid development in machine learning opens up for new possibilities in particle physics
- One such development presented here: simultaneous unfolding of many variables at once
- This means output of measurement will be a large set of events rather than a binned spectrum as we have now
  - Significant more information provided
  - Clear applications to e.g. MC tuning, searches for BSM effects, anomaly detection
- A lot of potential and rapid development
- Challenges and details around validation and guidelines still being worked out
- Exciting times ahead

Backup

Particle flow networks



$$\mathcal{O}(\{p_1,\ldots,p_M\}) = F\left(\sum_{i=1}^{M}\Phi(p_i)\right)$$

Symbol	Name	Short Description
PFN-ID	Particle Flow Network w. ID	PFN with full particle ID
PFN-Ex	Particle Flow Network w. PF ID	PFN with realistic particle ID
PFN-Ch	Particle Flow Network w. charge	PFN with charge information
PFN	Particle Flow Network	Using three-momentum information
EFN	Energy Flow Network	Using IRC-safe information
RNN-ID	Recurrent Neural Network w. ID	RNN with full particle ID
RNN	Recurrent Neural Network	Using three-momentum information
EFP	Energy Flow Polynomials	A linear basis for IRC-safe information
DNN	Dense Neural Network	Trained on an N-subjettiness basis
CNN	Convolutional Neural Network	Trained on 33×33 grayscale jet images
M	Constituent Multiplicity	Number of particles in the jet
n_{SD}	Soft Drop Multiplicity	Probes number of perturbative emissions
m	Jet Mass	Mass of the jet

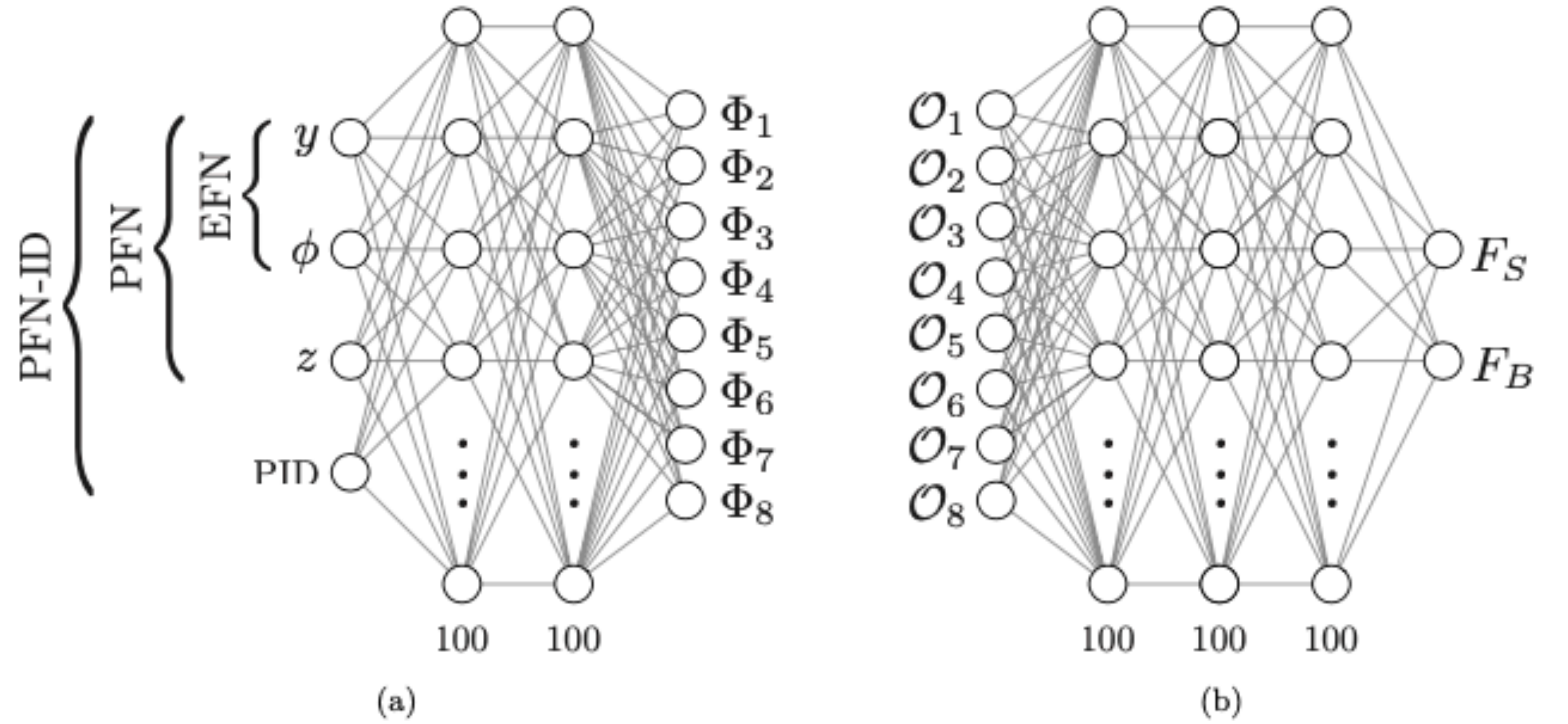
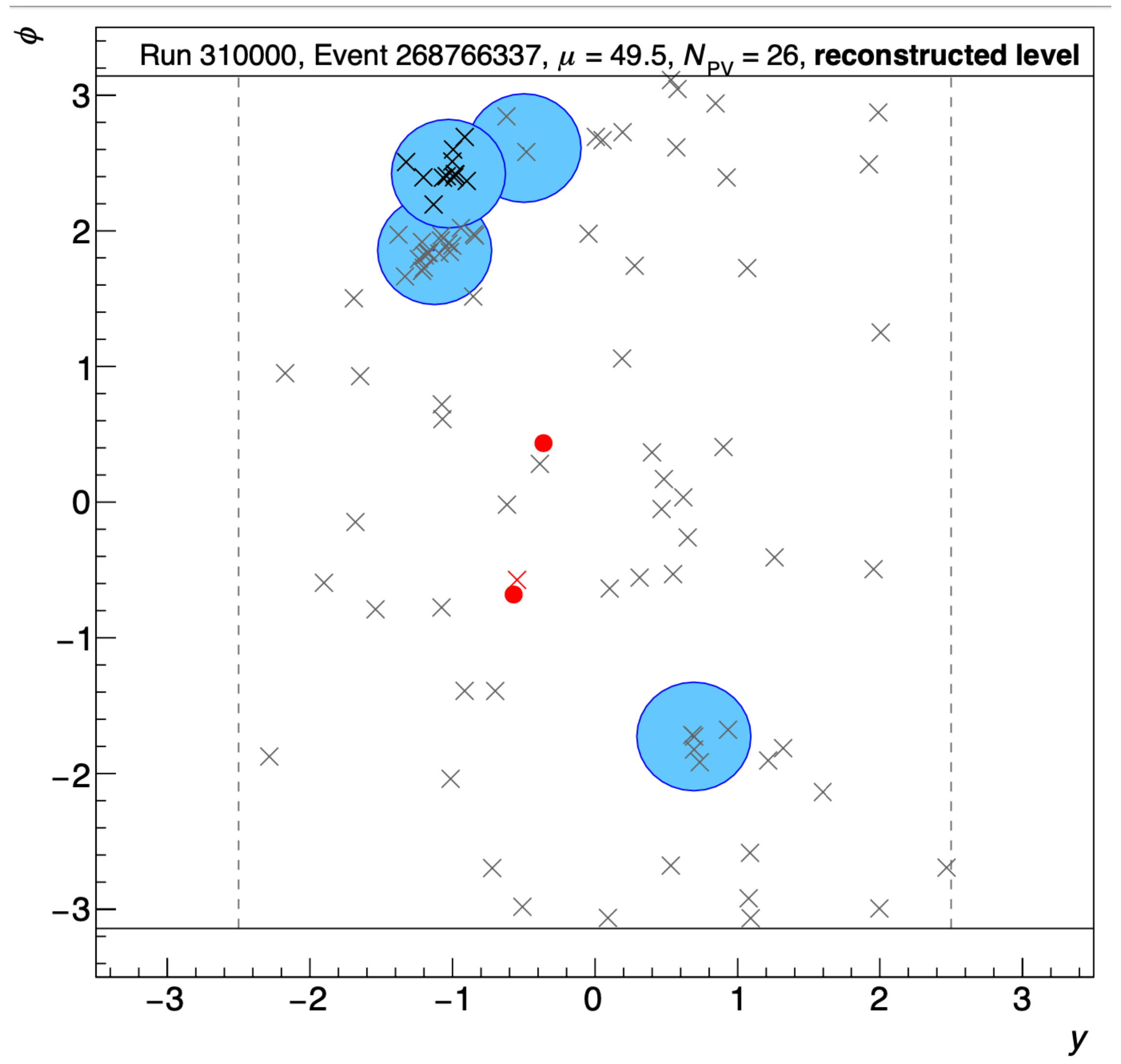
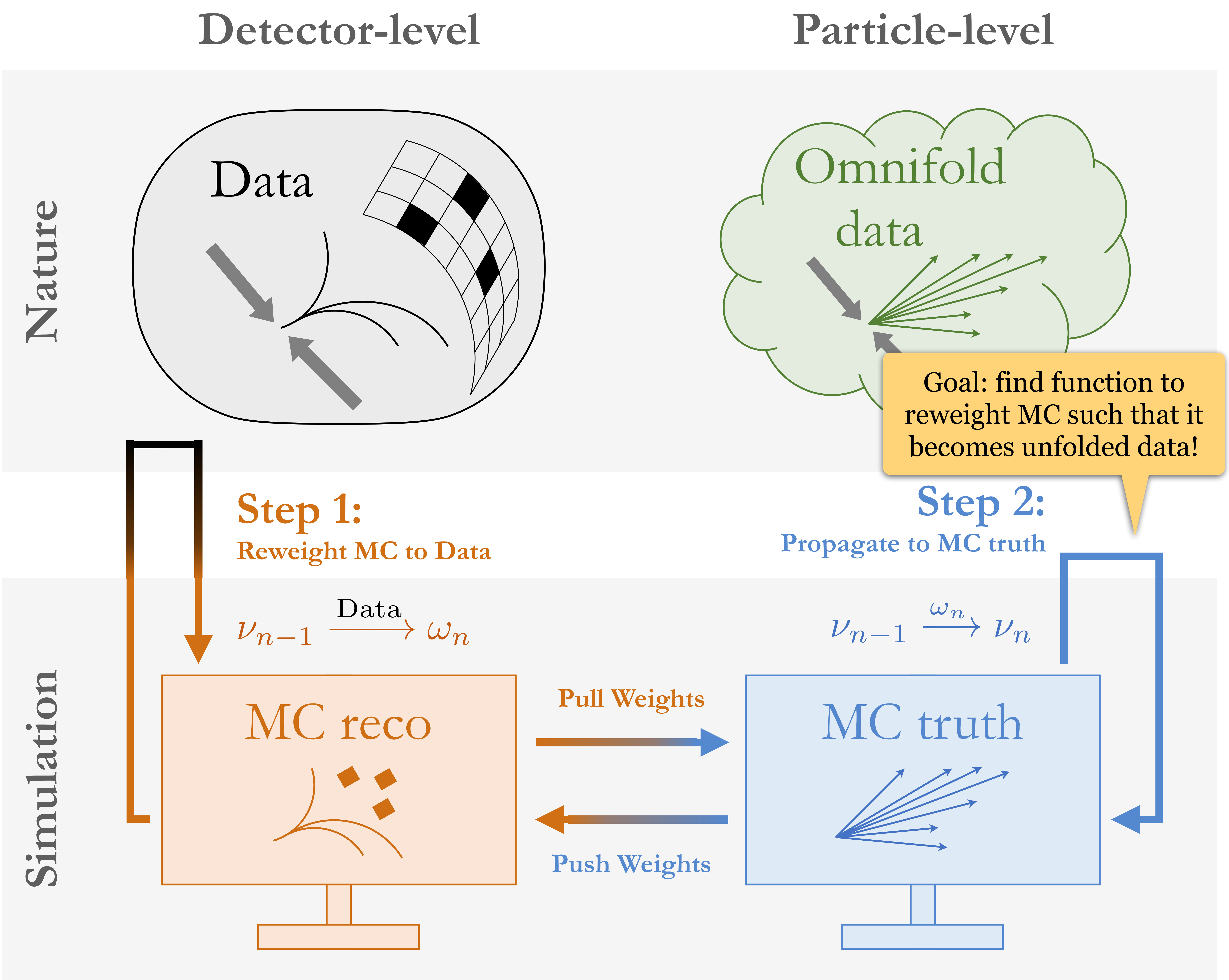


Figure 4: The particular dense networks used here to parametrize (a) the per-particle



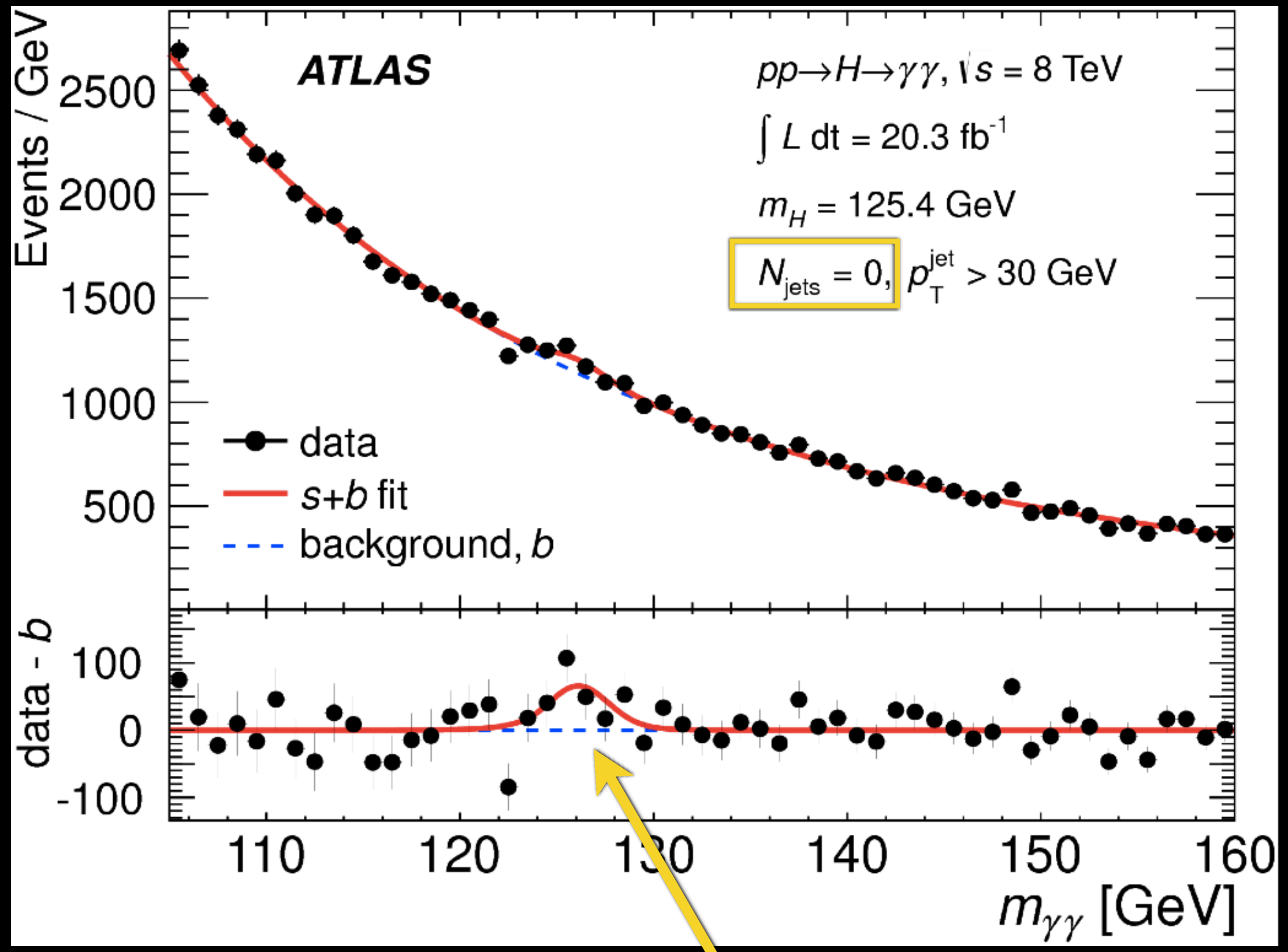
Energy Flow Networks: Deep Sets for Particle Jets #1
Patrick T. Komiske (MIT, Cambridge, CTP), Eric M. Metodiev (Harvard U. and MIT, Cambridge, CTP), Jesse Thaler (Harvard U. and MIT, Cambridge, CTP) (Oct 11, 2018)
Published in: *JHEP* 01 (2019) 121 • e-Print: 1810.05165 [hep-ph]



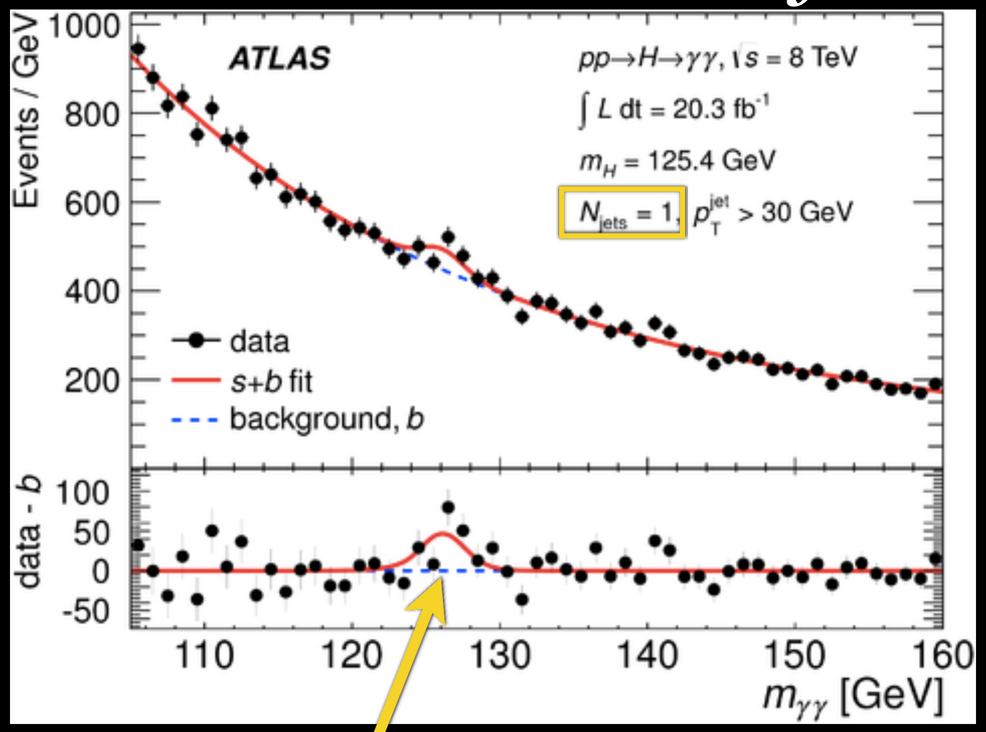
Detector-level
Particle-level
Nature
Simulation
Data
Omnifold data
Goal: find function to reweight MC such that it becomes unfolded data!
Step 1:
Reweight MC to Data
Step 2:
Propagate to MC truth
$\nu_{n-1} \xrightarrow{\text{Data}} \omega_n$
$\nu_{n-1} \xrightarrow{\omega_n} \nu_n$
MC reco
MC truth
Pull Weights
Push Weights

Measuring Higgs boson distributions

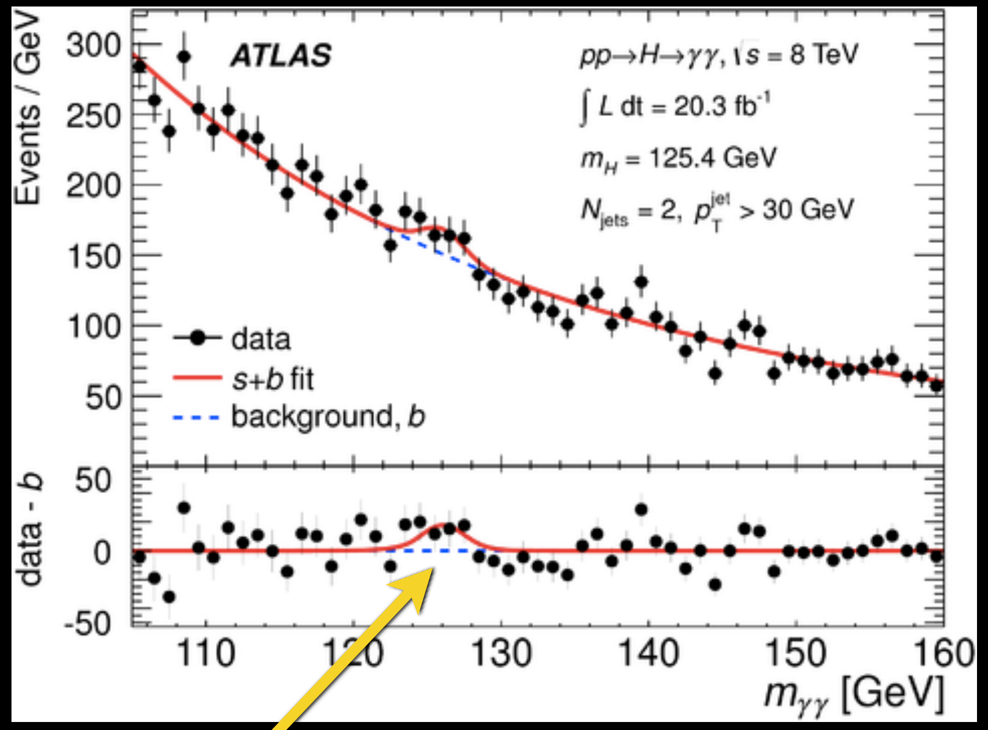
All events with no jets

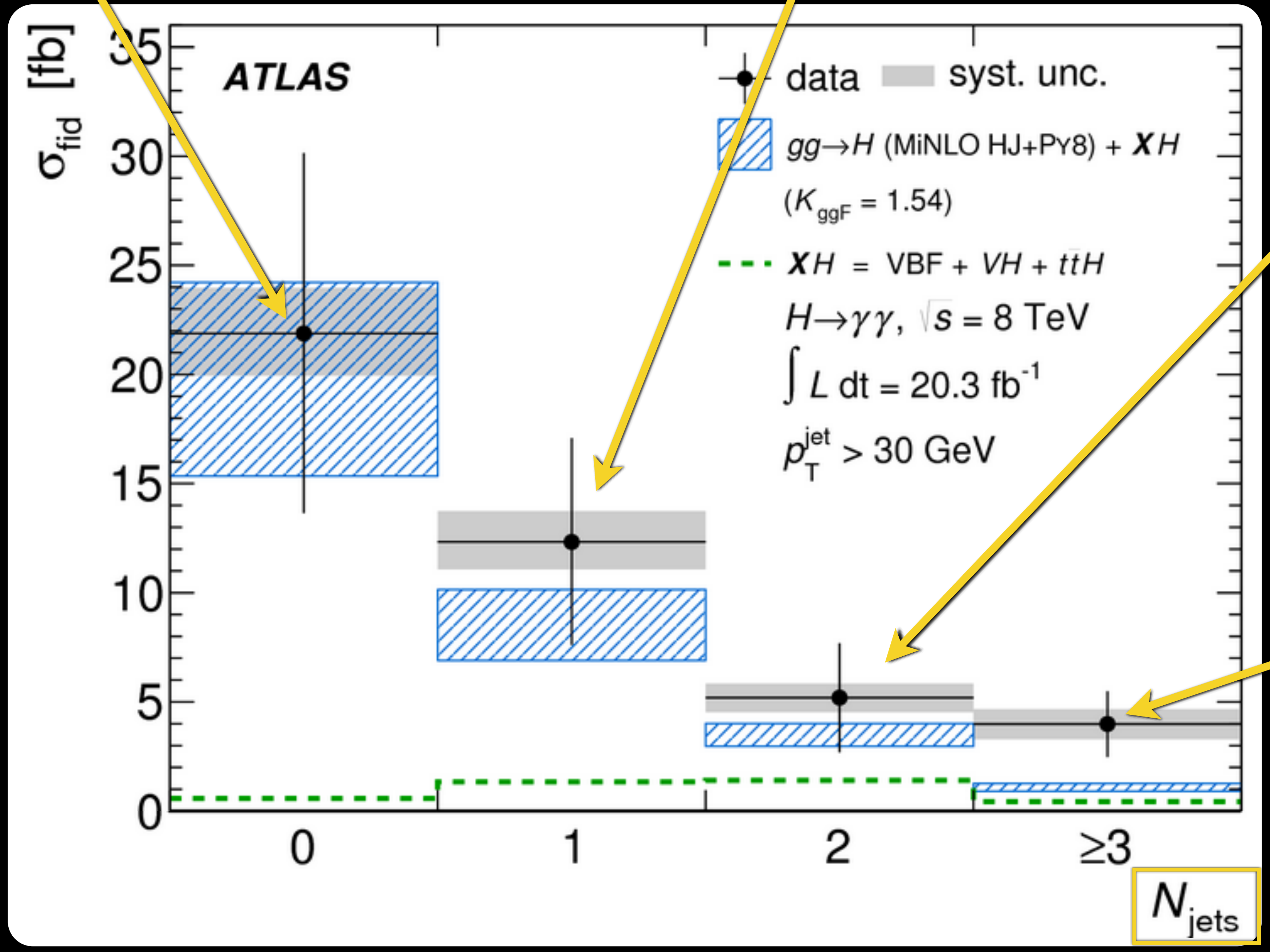


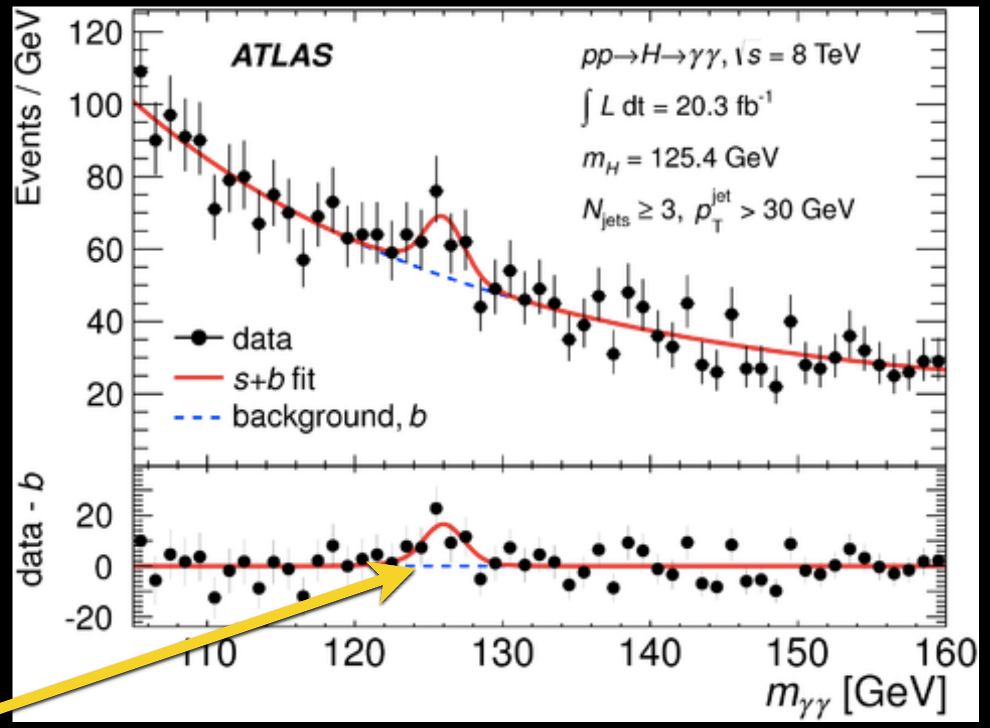
All events with one jet



All events with two jets …



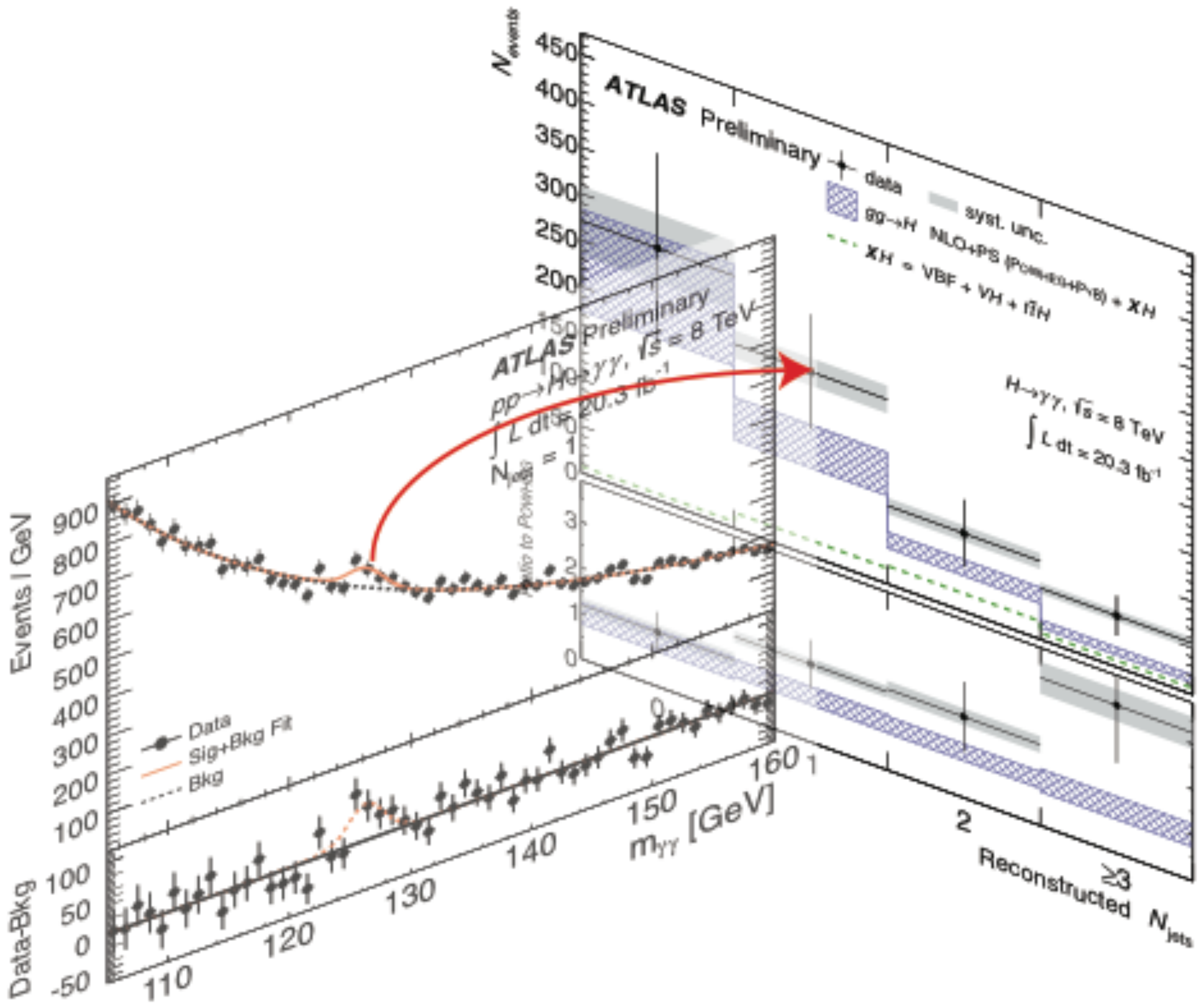




All events with 3 or more jets

Differential cross section measurement overview

1. Signal extraction



a) Spit dataset into bins of variable of interest (here 4 N_{jets} bins)

b) For each bin, extract s from a s+b fit to the $m_{\gamma\gamma}$ spectra

c) Large statistical uncertainty due to small s/b

2. Unfold to particle level

and divide by integrated luminosity and bin-width

$$\sigma_{\text{fid}} = \frac{n_{\text{sig},i}}{c_i \; \mathcal{L}_{\text{int}}}$$

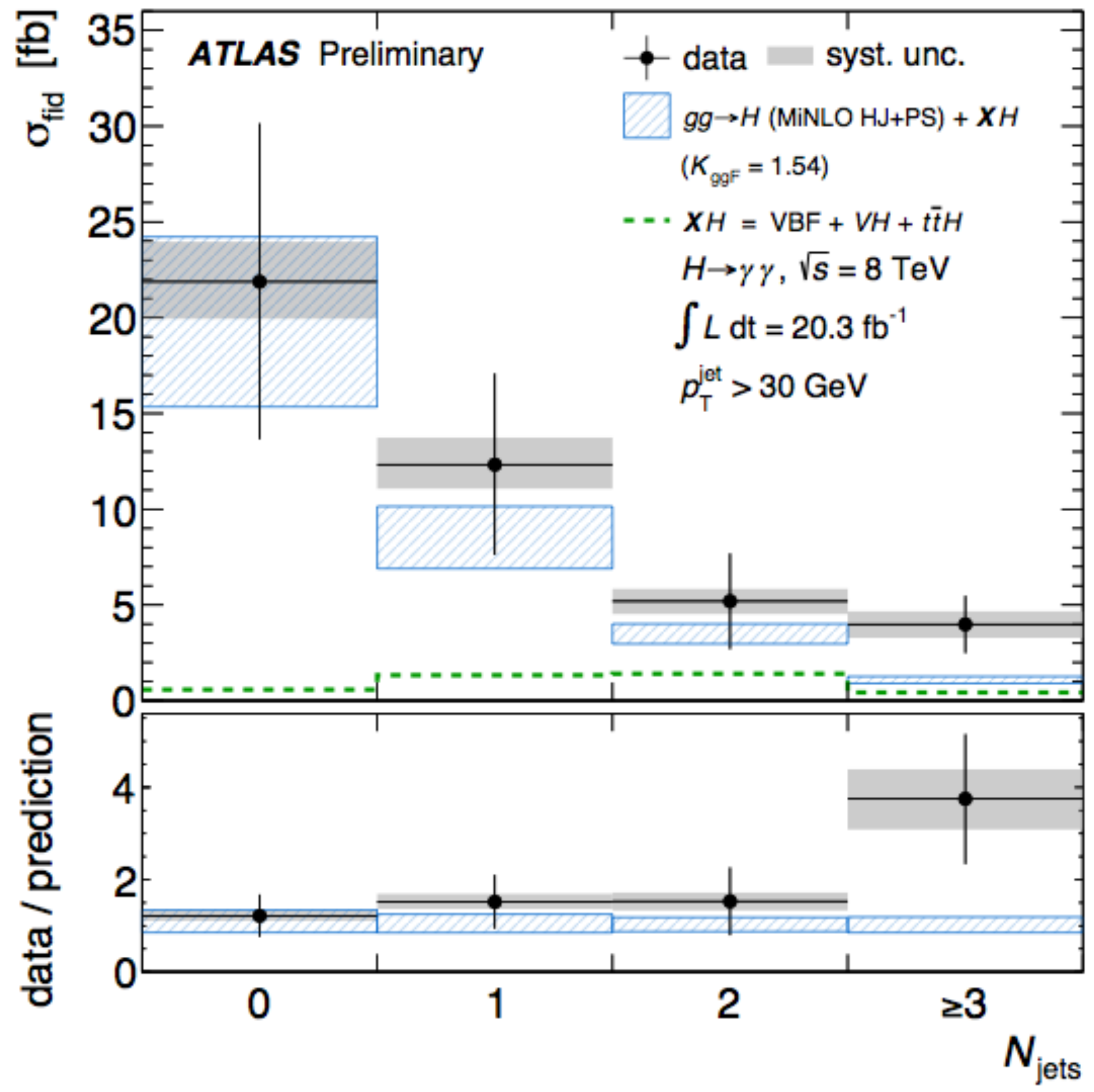
correction factor for detector effects

20.3 fb^{-1} (±2.8%)

a) correction for detector effects with bin-by-bin unfolding

b) convert to ("differential") cross section by dividing by int. lumi (and bin-width)

3. Plot and compare with theory



a) compare to **particle level** prediction - i.e. no need for detector simulation

b) Can also compare with analytical calculations (parton level) but then need small parton→particle level (NP) correction

Likelihood fit for EW *Zjj* signal extraction

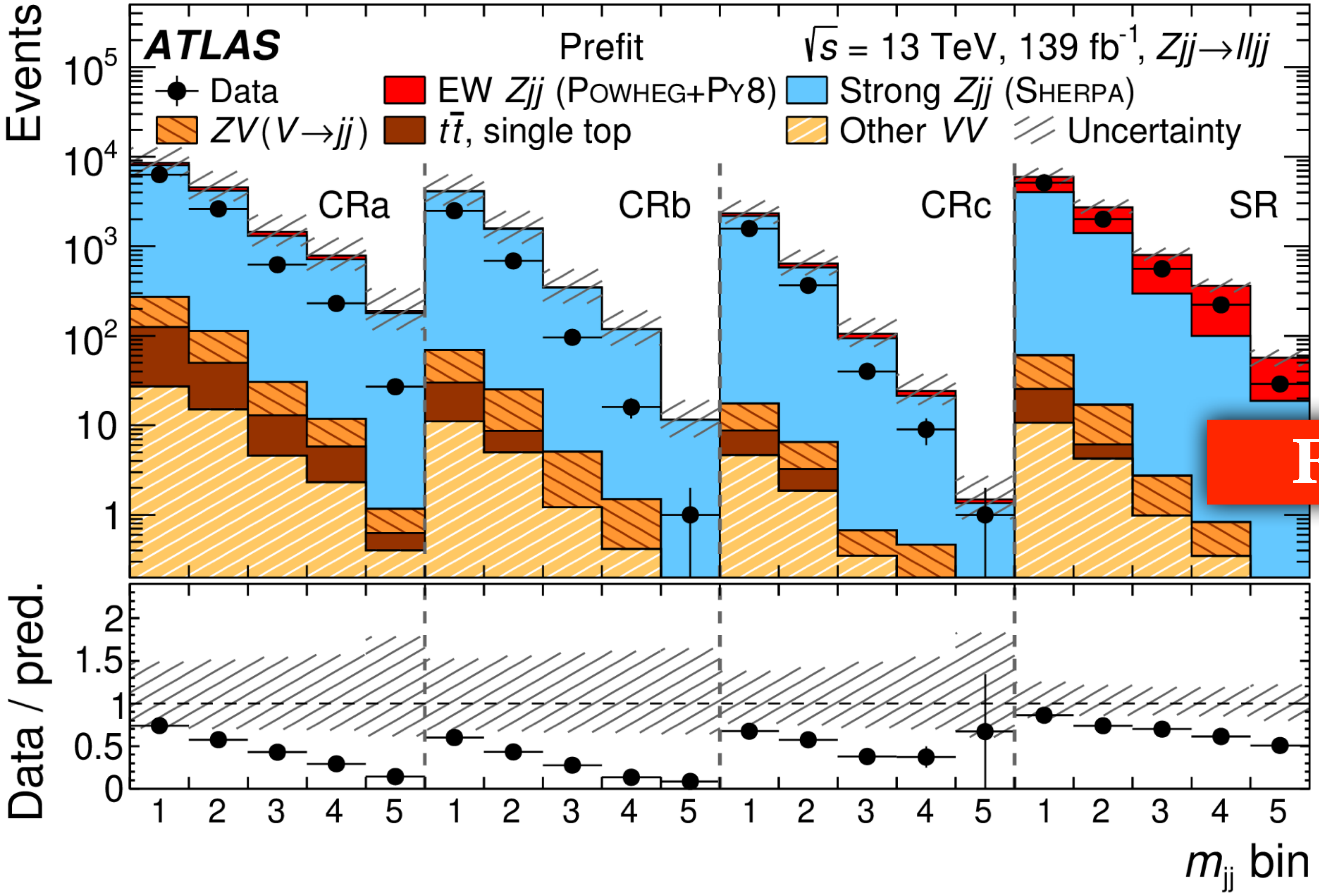
$$\ln \mathcal{L} = -\sum_{r,i} \nu_{ri}(\boldsymbol{\theta}) + \sum_{r,i} N_{ri}^{\text{data}} \ln \nu_{ri}(\boldsymbol{\theta}) - \sum_{s} \frac{\theta_s^2}{2},$$

$$\nu_{ri} = \boxed{\mu_i}\, \nu_{ri}^{\text{EW,MC}} + \nu_{ri}^{\text{strong}} + \nu_{ri}^{\text{other,MC}},$$

$$\nu_{\text{CRa},i}^{\text{strong}} = b_{\text{L},i}\, \nu_{\text{CRa},i}^{\text{strong,MC}}, \qquad \nu_{\text{CRb},i}^{\text{strong}} = b_{\text{H},i}\, \nu_{\text{CRb},i}^{\text{strong,MC}},$$
$$\nu_{\text{SR}i}^{\text{strong}} = b_{\text{L},i}\, f(x_i)\, \nu_{\text{SR},i}^{\text{strong,MC}}, \qquad \nu_{\text{CRc},i}^{\text{strong}} = b_{\text{H},i}\, f(x_i)\, \nu_{\text{CRc},i}^{\text{strong,MC}}$$

20 bins, 5 POIs, 12 free parameter that constrain strong *Zjj* (5+5+2)

Each red bin count (after fitting, in SR) is: $\hat{\nu}_i^{\text{EW}} = \hat{\mu}_i \nu_i^{\text{EW,MC}}$



Fit