Precision measurements and open data at future colliders

Do recent developments in AI change the picture?

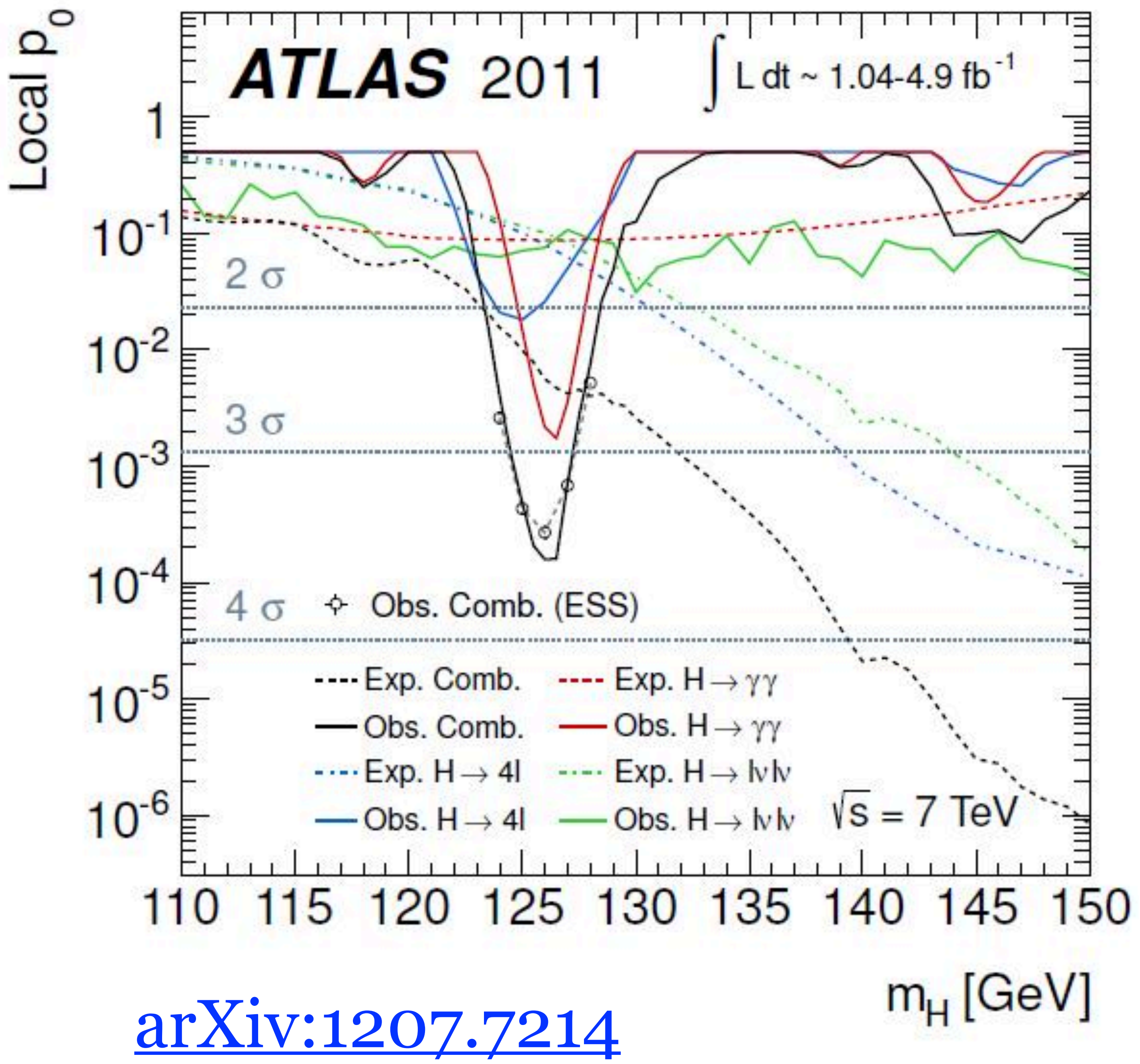
Dag Gillberg, Carleton
TRIUMF Future Collider Conference

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
  - A new possible approach to future precision measurements
    - Factorization
    - Increased collaboration
  - Challenges and open questions
- Backup
  - Underlying mechanism

Particle physics measurements

- Two main classes of experimental analyses
  - Searches
  - Measurements - ***focus of this talk***

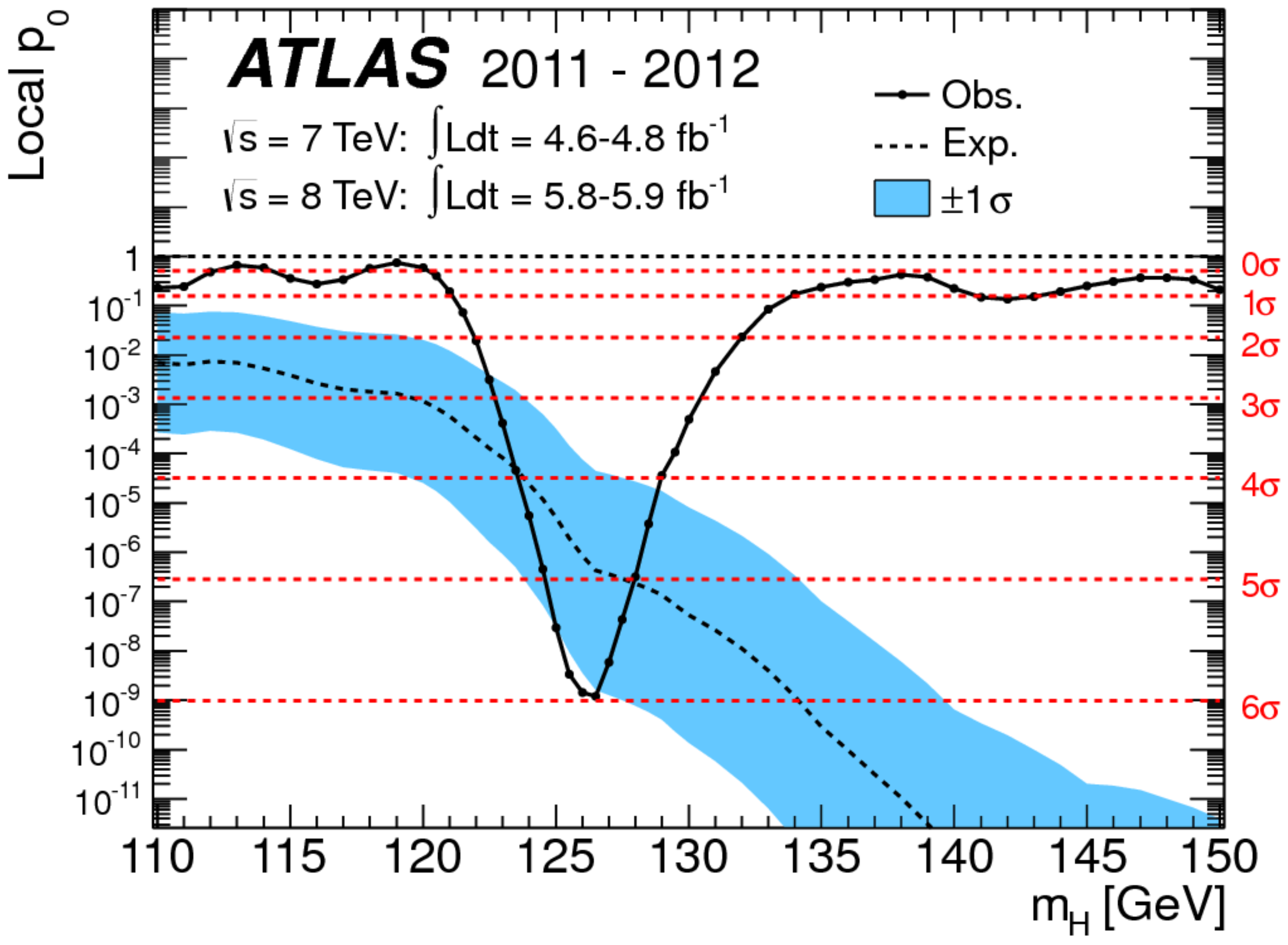


arXiv:1207.7214

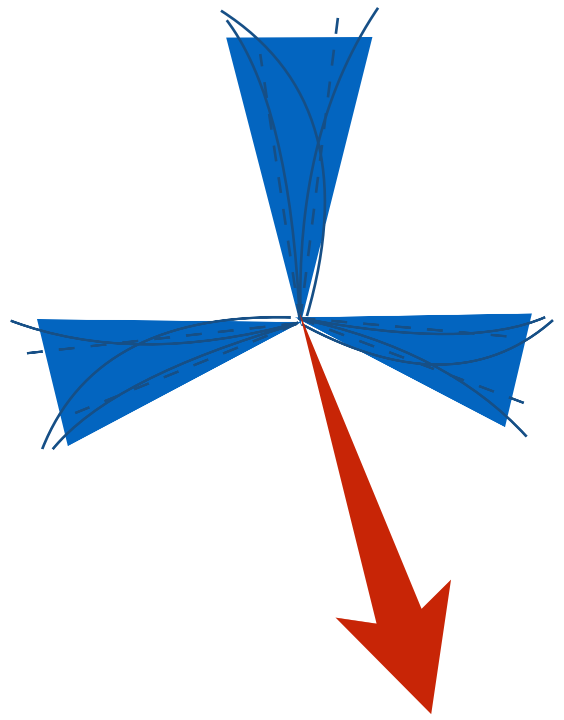
ATLAS-CONF-2016-067

Particle physics measurements

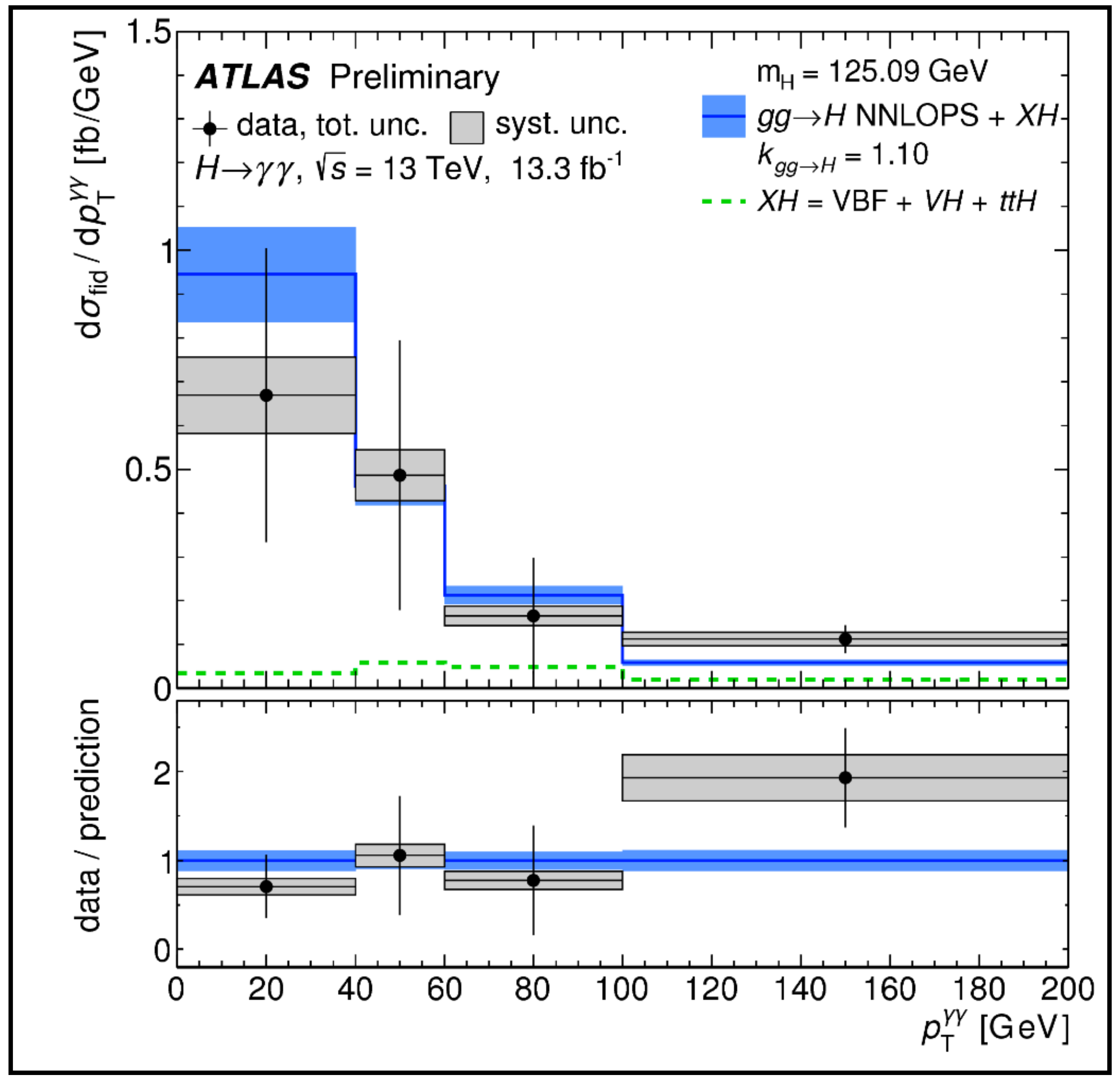
- Two main classes of experimental analyses
  - Searches
  - Measurements - ***focus of this talk***



arXiv:1207.7214



Higgs boson transverse momentum



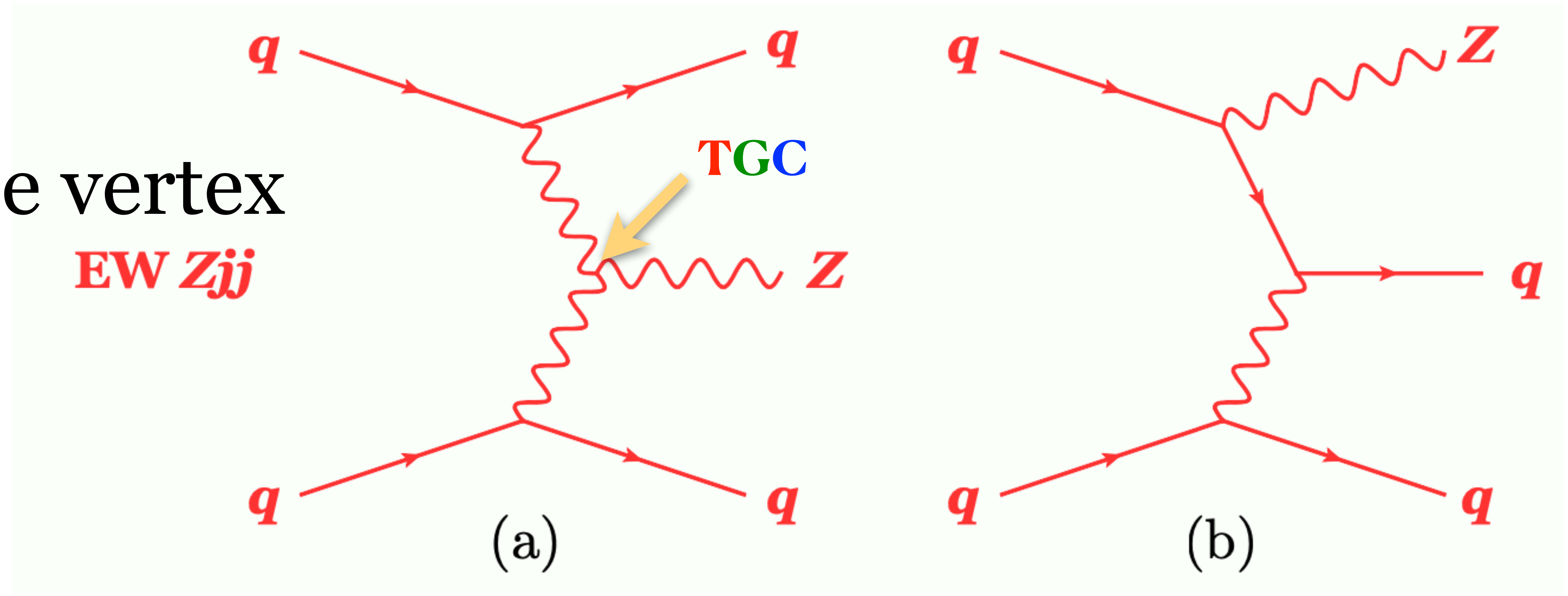
Early Higgs boson transverse momentum measurement

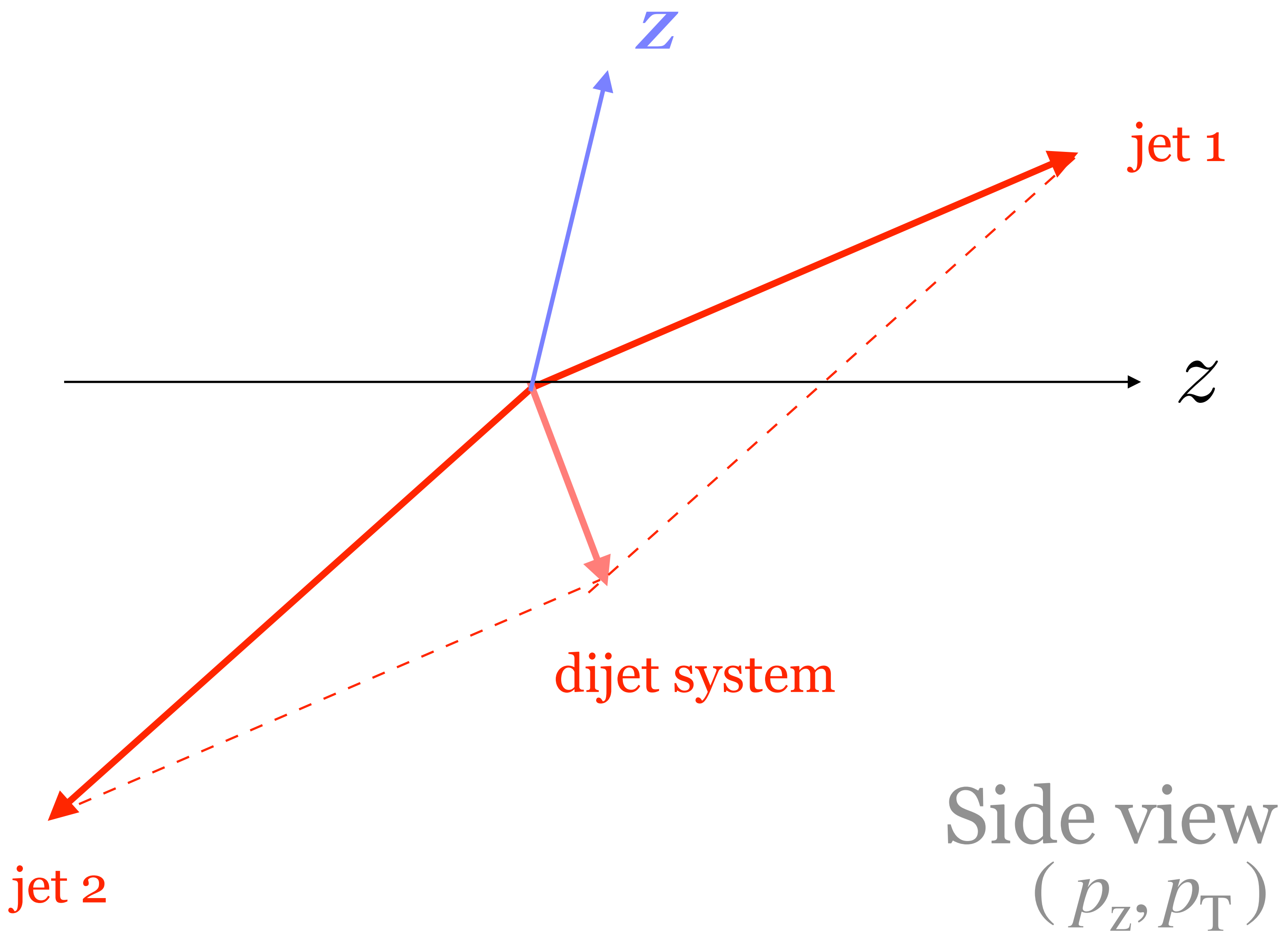
ATLAS-CONF-2016-067

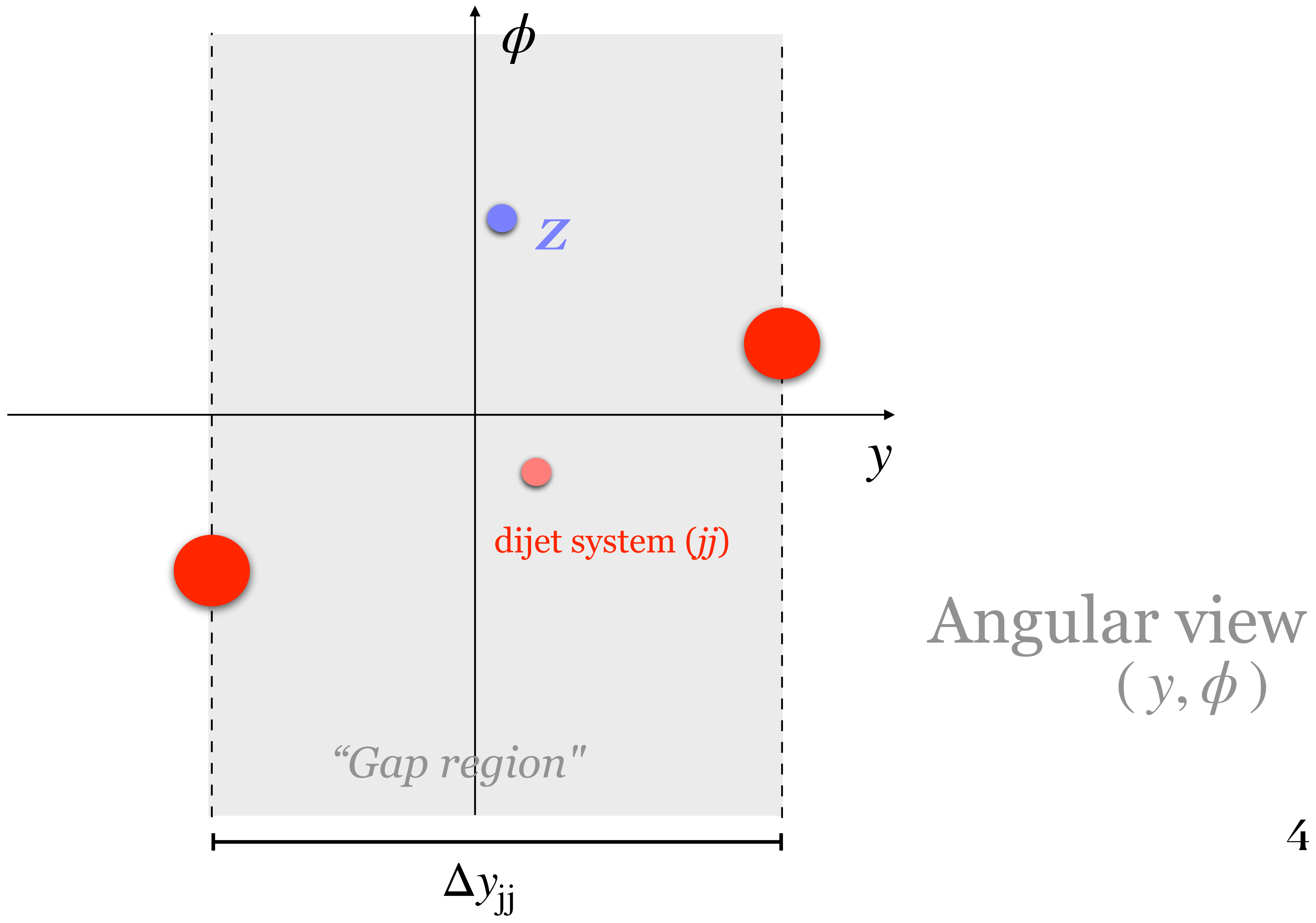
arXiv:2006.15458 (2020)

Example of a ‘present day’ measurement

- 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



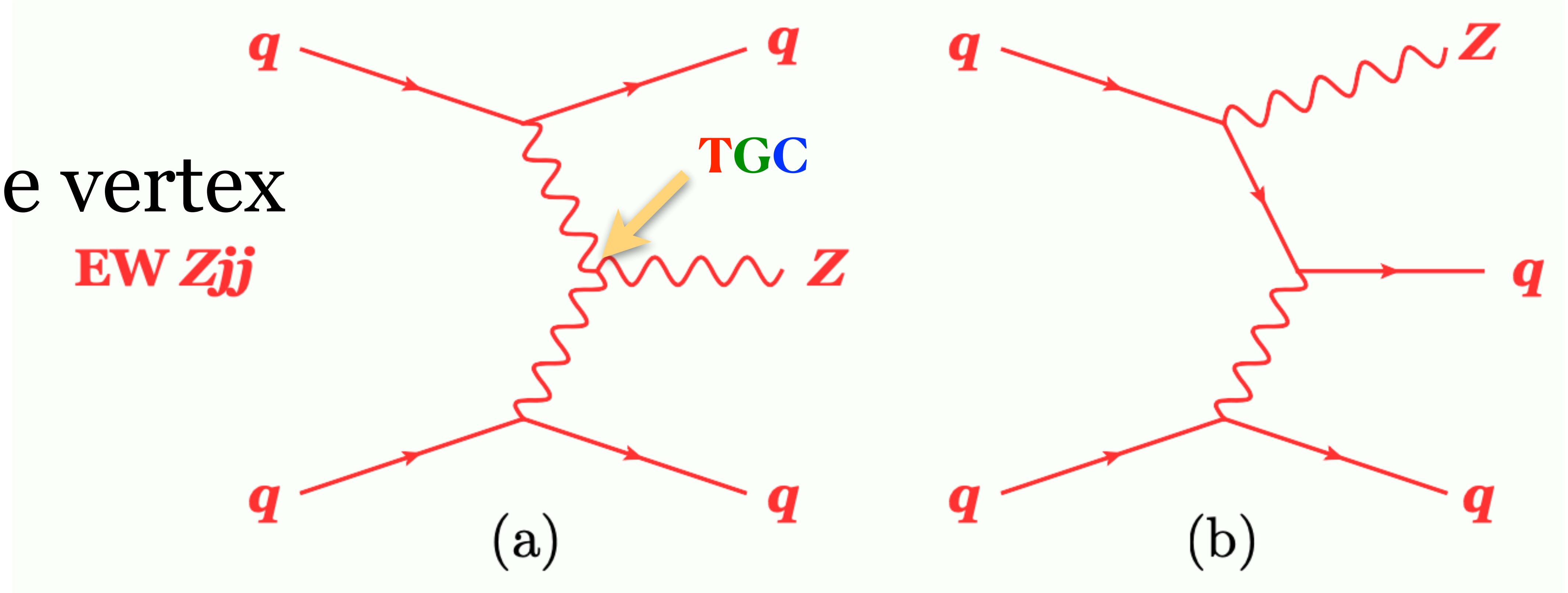


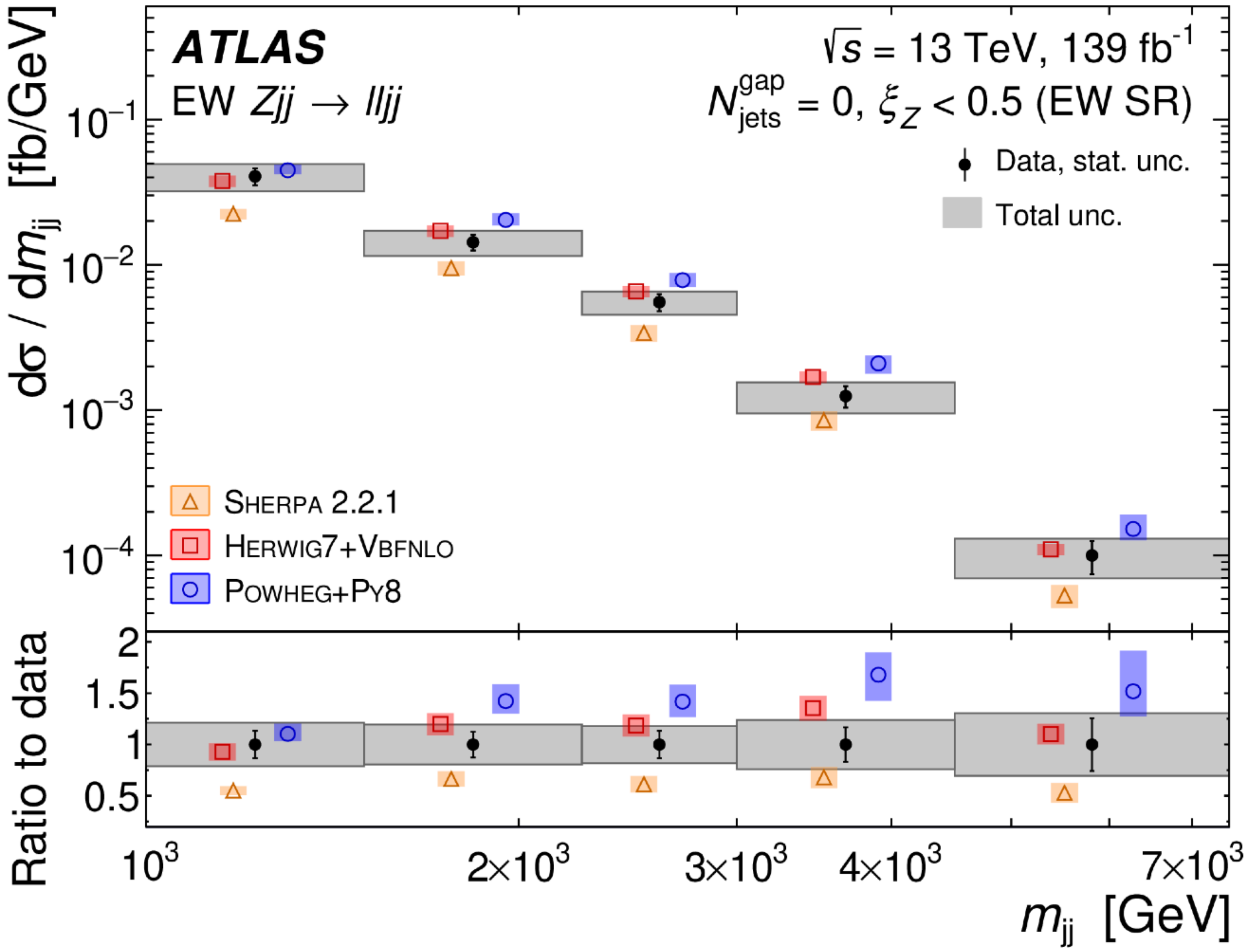


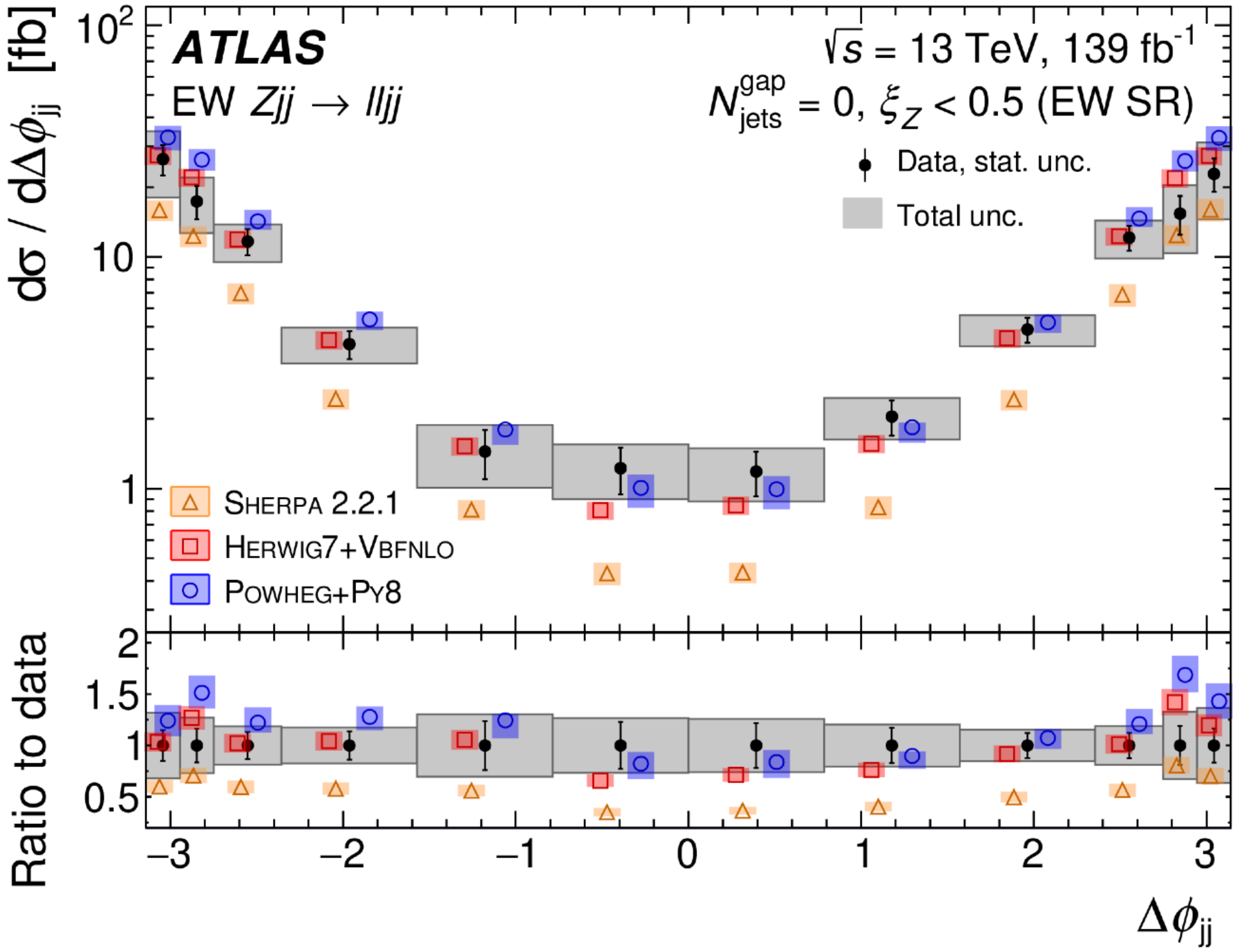
arXiv:2006.15458 (2020)

Example of a 'present day' measurement

- 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



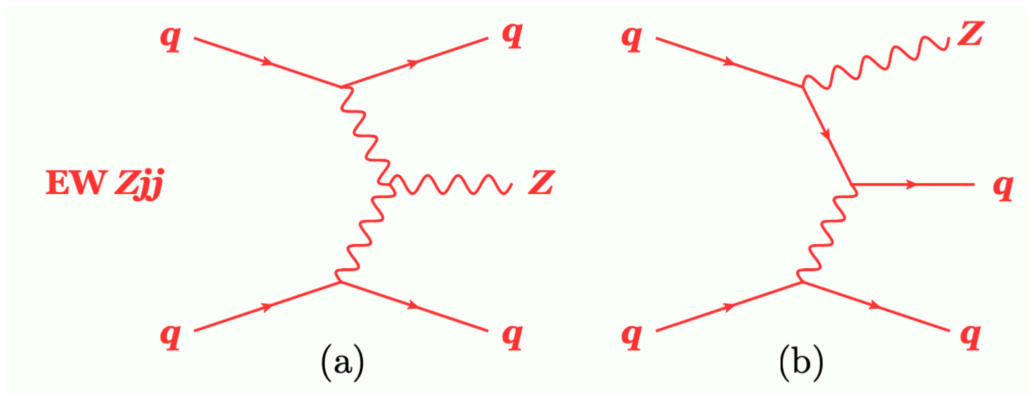




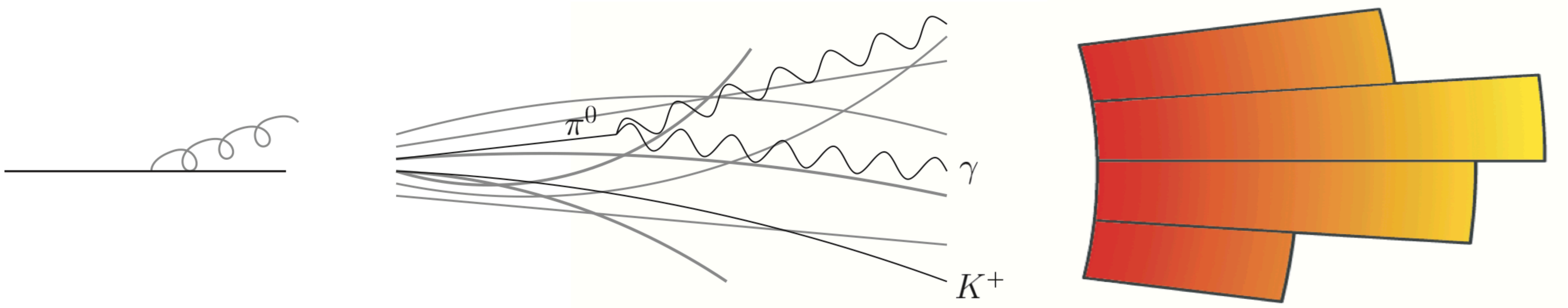
gular view
(y, ϕ)

Key output of many precision measurements
Binned differential spectra at particle level

Precision measurements



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



parton level jet

particle level jet

calorimeter level jet

Quarks/gluons don't exist as free particles

Cannot be observed

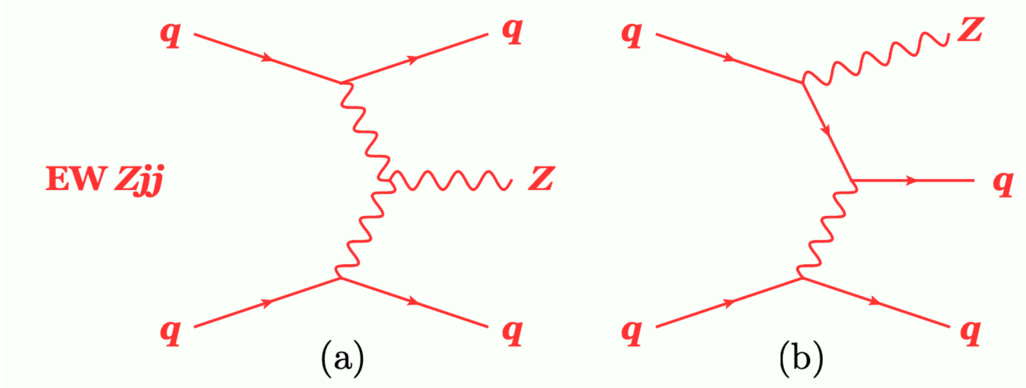
*Final state! Observable in **nature***

"What a perfect detector would see"

Reconstructed level

What we measure in the detector

Precision measurements



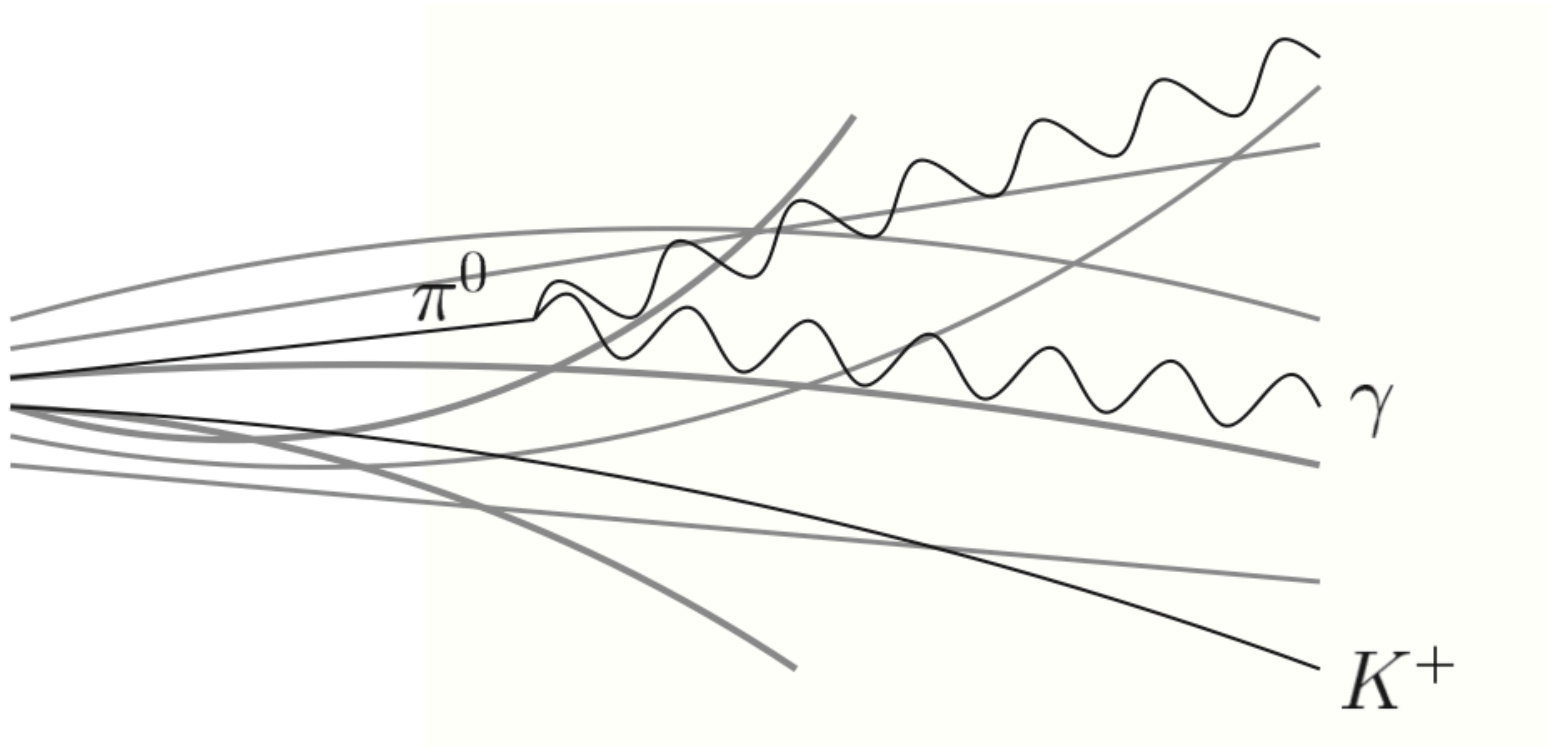
- For a measurement to be useful, it needs a **precise definition**
- Standard: define measurement at the stable particle level
  - Real particles with life time $c\,\tau_0 > 10$ mm ($\pi^{\pm}, p, n, K, e^{-}, e^{+}$ ...)

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, $\lvert\Delta y_{jj}\rvert > 2,\ \xi_Z < 1$

Fiducial definition of the *Zjj* measurement
Defined at the *particle level*

Quarks/gluons don't exist as free particles

Cannot be observed



particle level jet

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

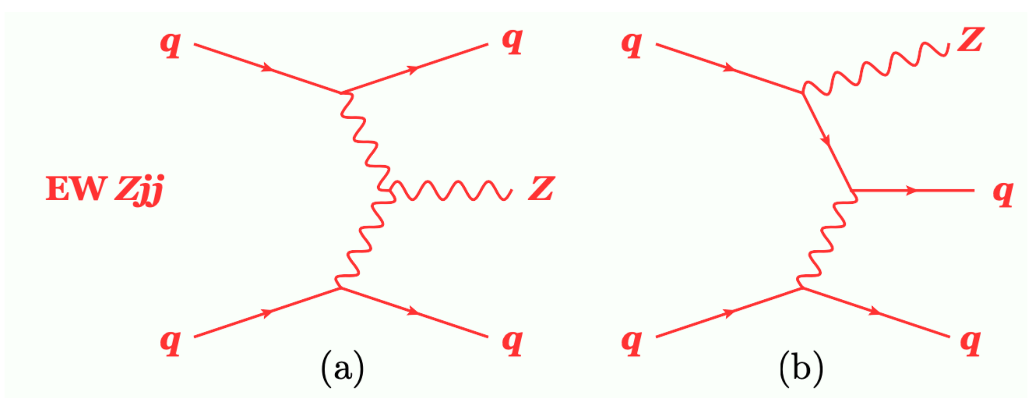
"What a perfect detector would see"

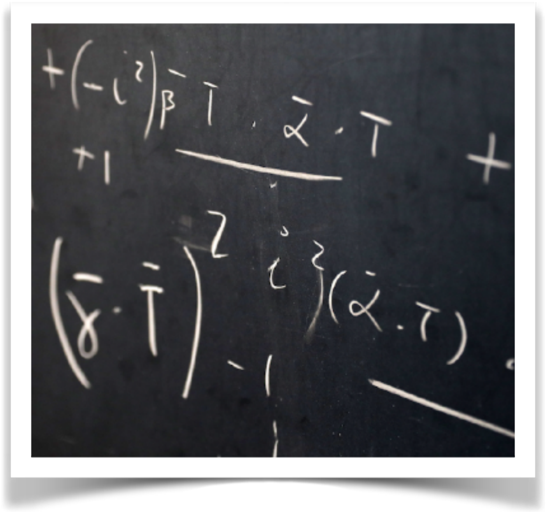
calorimeter level jet

Reconstructed level

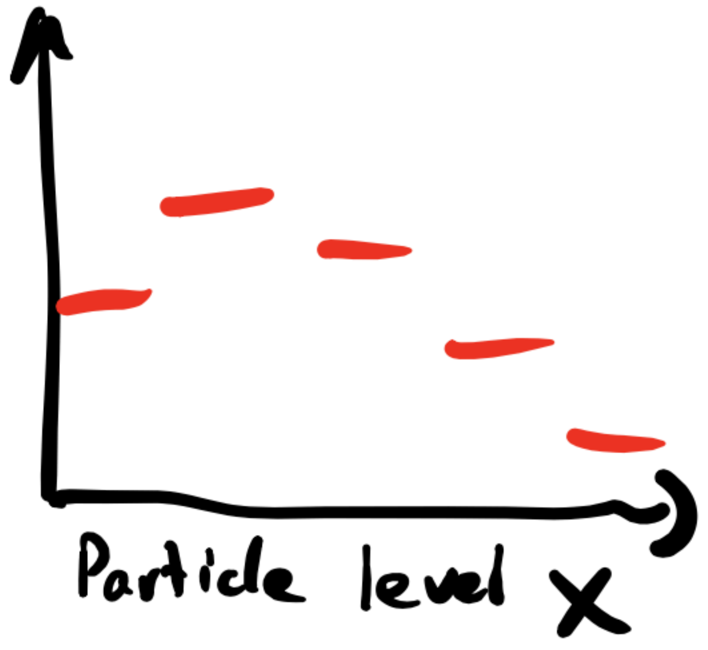
What we measure in the detector

Science at work

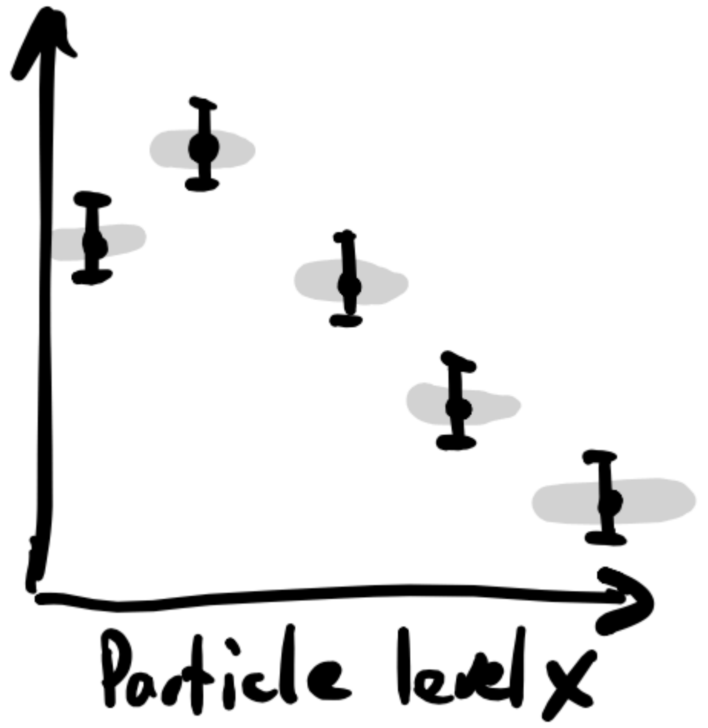


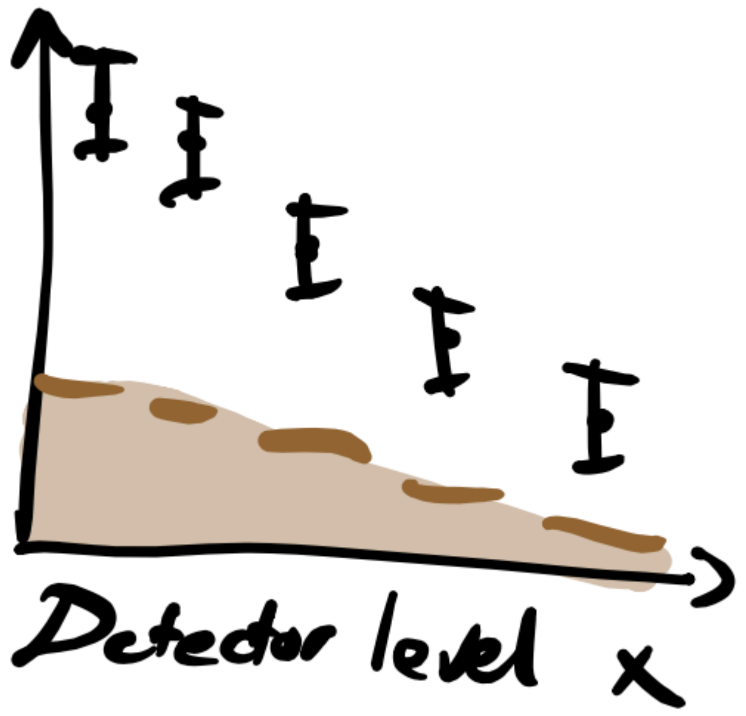


Prediction



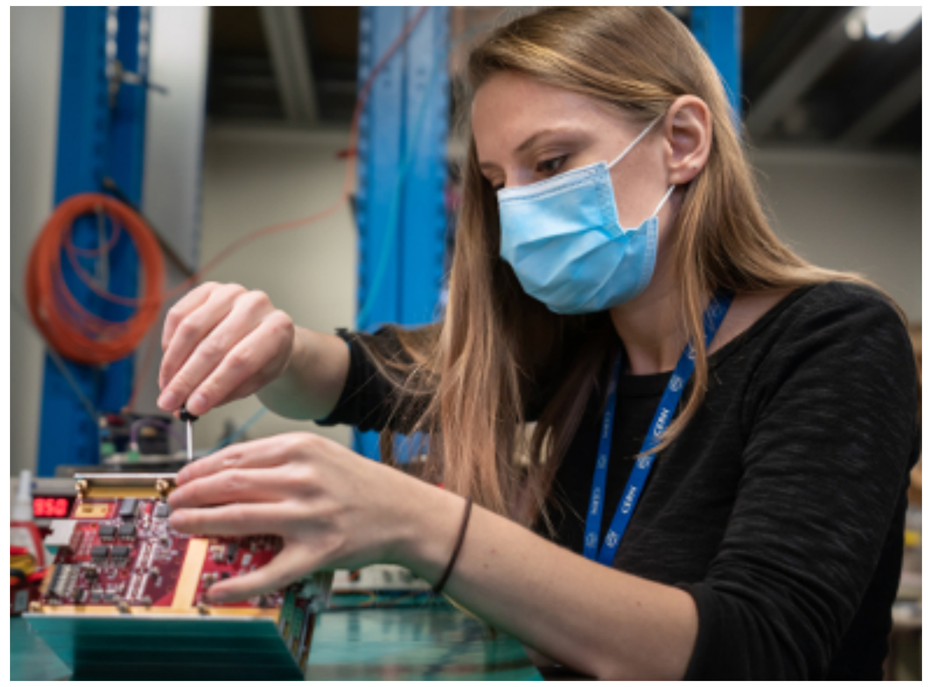
Measurement





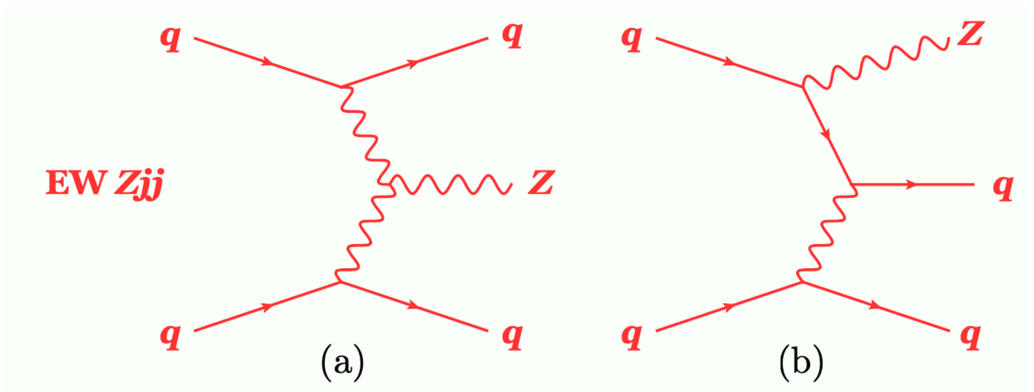
Hypothesis test!

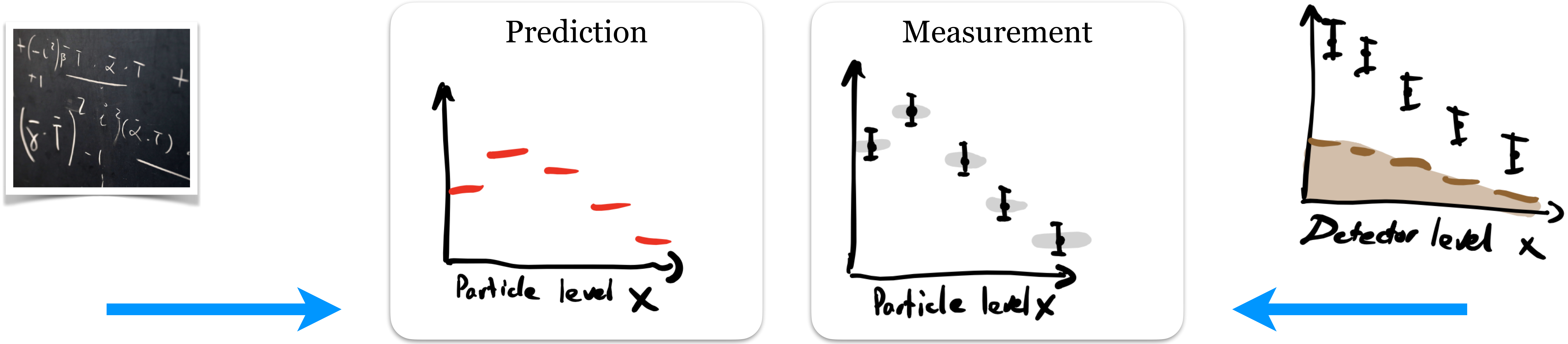
Theorists



Experimentalists

Science at work





Example workflow

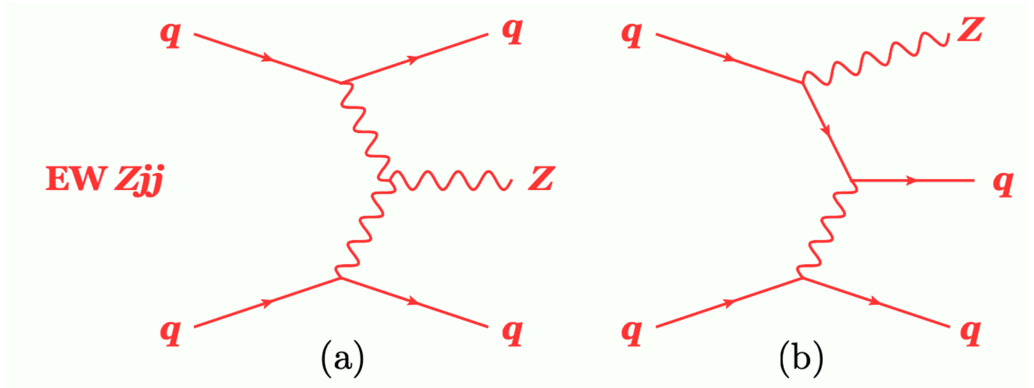
1. UFO module → MadGraph5
2. Generte events with parton shower and hadronization (e.g. MG5+Py8)
3. Feed to Rivet

Theorists

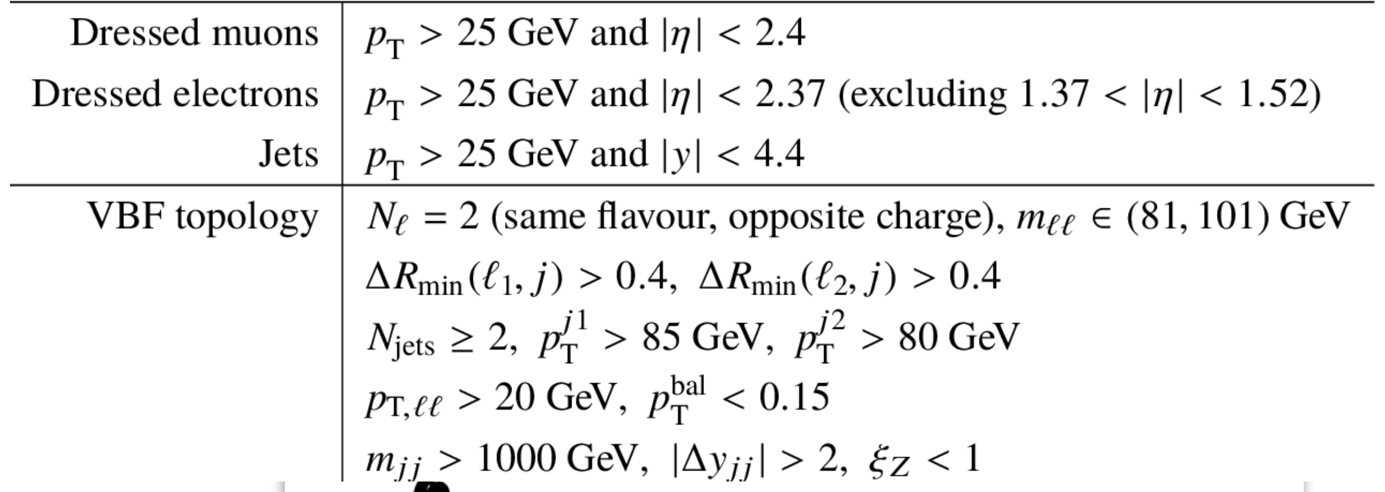
Example workflow

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

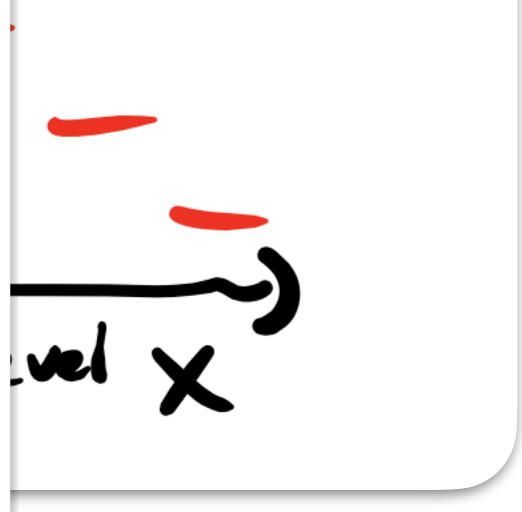
Experimentalists



Science at work



Dressed muons	$p_{\mathrm{T}} > 25$ 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
	$\Delta R_{\min}(\ell_1, j) > 0.4$, $\Delta R_{\min}(\ell_2, j) > 0.4$
	$N_{\mathrm{jets}} \geq 2$, $p_{\mathrm{T}}^{j1} > 85$ 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, $\lvert\Delta y_{jj}\rvert > 2$, $\xi_Z < 1$

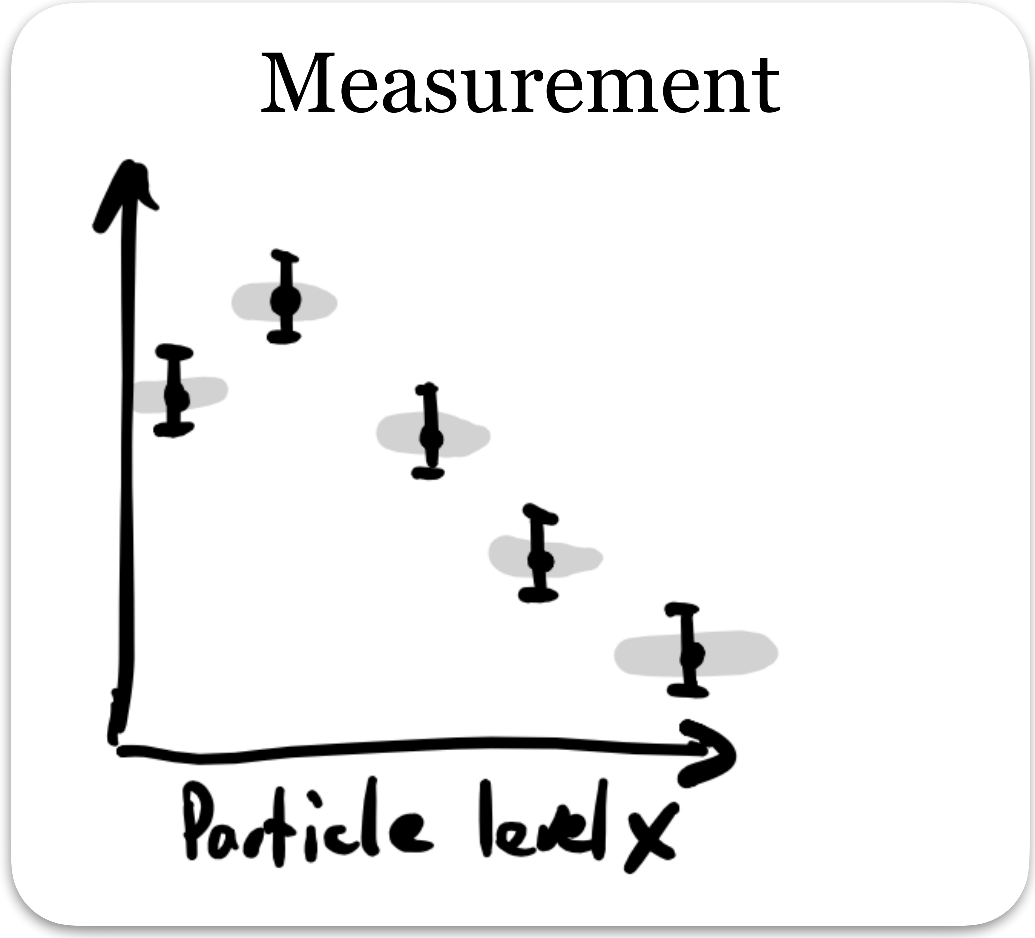


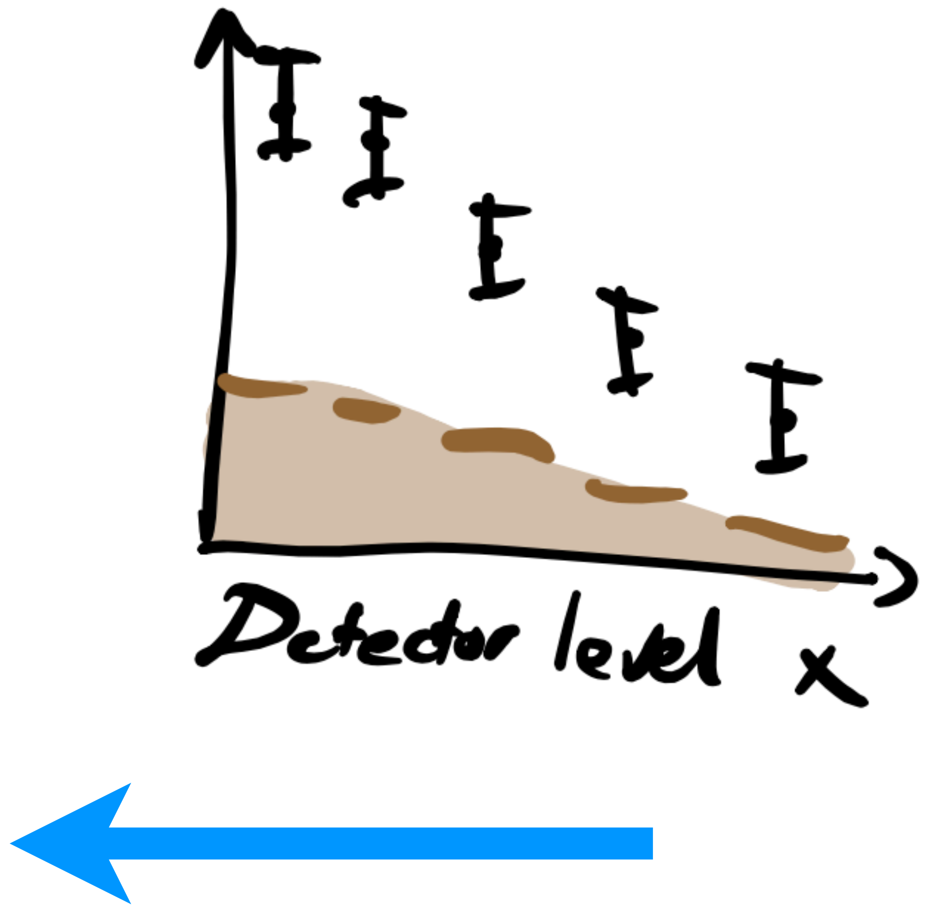
```
    /// Perform the per-event analysis
    void analyze(const Event& event) {

      // 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;
      }
```

r

)

3. Feed to Rivet





Example workflow

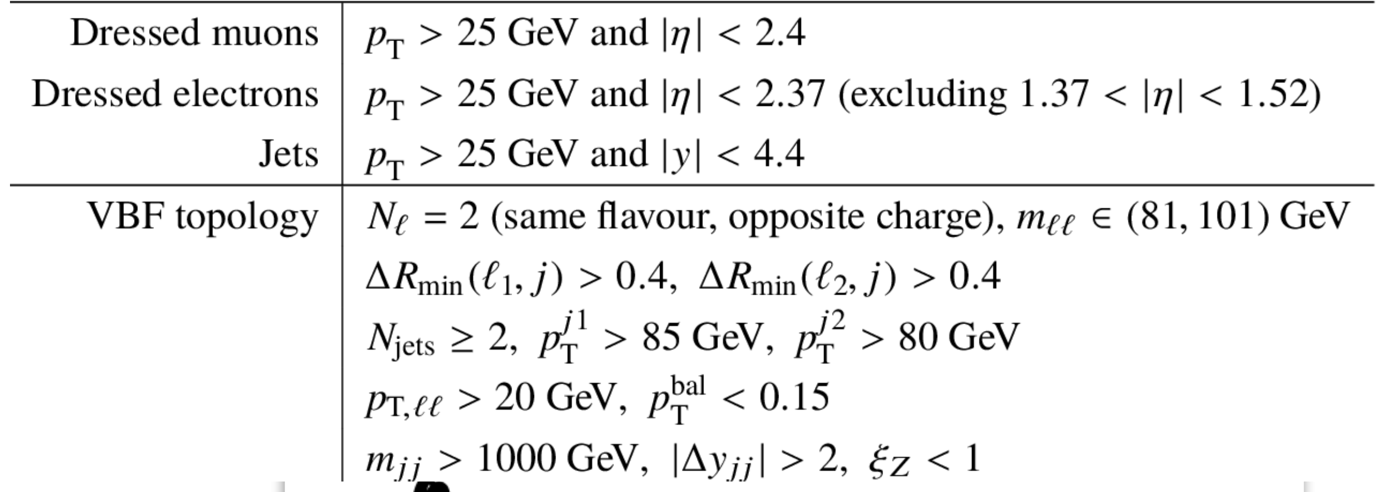
0. (Build detector, operate, calibrate)
1. Event reconstruction+analysis
2. Correct for detector effects
3. Make data public

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 $\Delta R_{\min}(\ell_1, j) > 0.4,\ \Delta R_{\min}(\ell_2, j) > 0.4$ $N_{\text{jets}} \geq 2,\ p_{\mathrm{T}}^{j1} > 85$ GeV, $p_{\mathrm{T}}^{j2} > 80$ GeV $p_{\mathrm{T},\ell\ell} > 20$ GeV, $p_{\mathrm{T}}^{\text{bal}} < 0.15$ $m_{jj} > 1000$ GeV, $\lvert\Delta y_{jj}\rvert > 2,\ \xi_Z < 1$

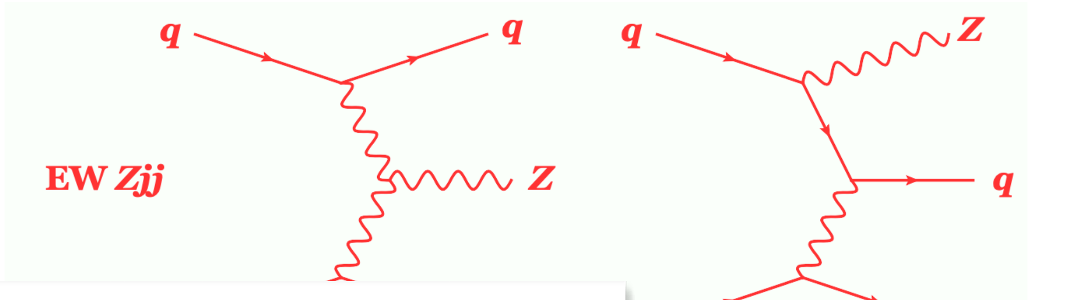
```
    /// Perform the per-event analysis
    void analyze(const Event& event) {

      // 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;
      }
```

r

)

3. Feed to Rivet

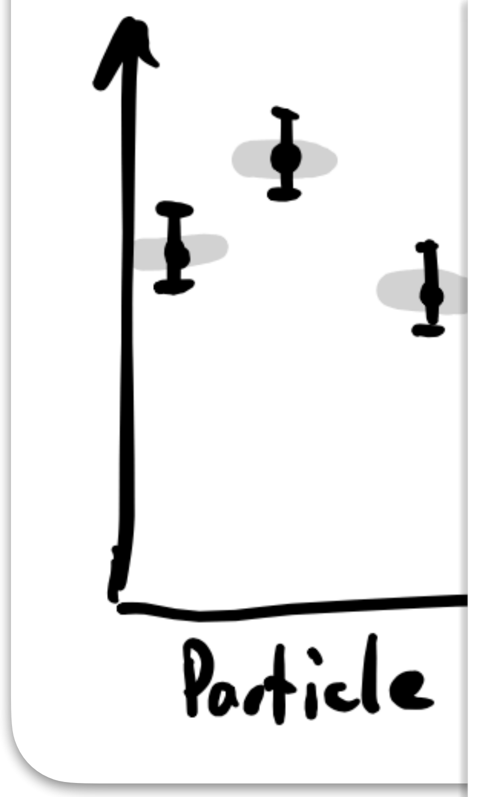
Theorists

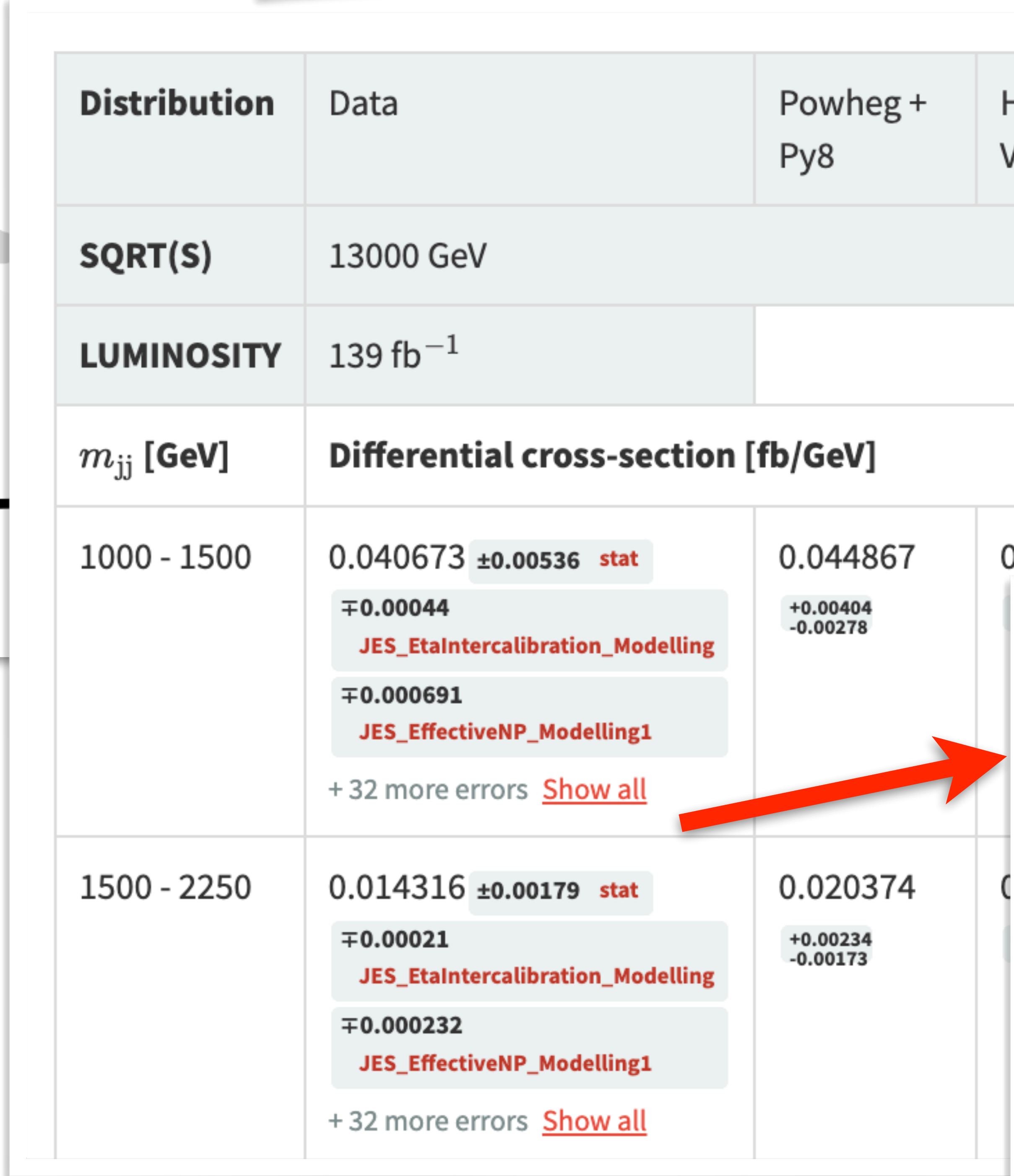


HEPData

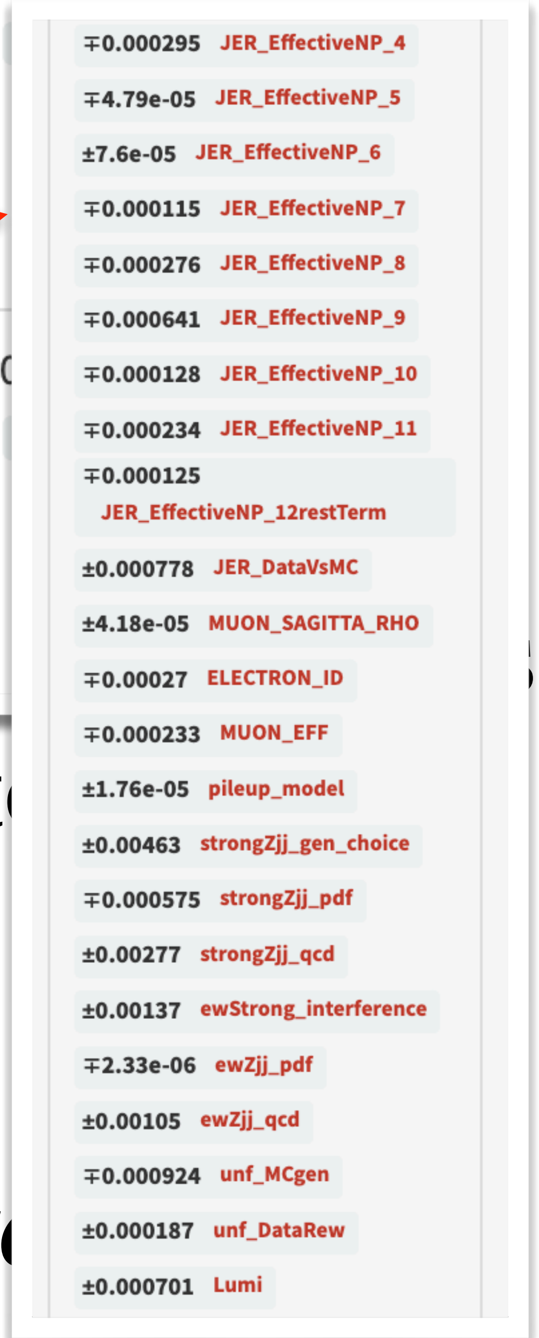
Repository for publication-related High-Energy Physics data

Measurement





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

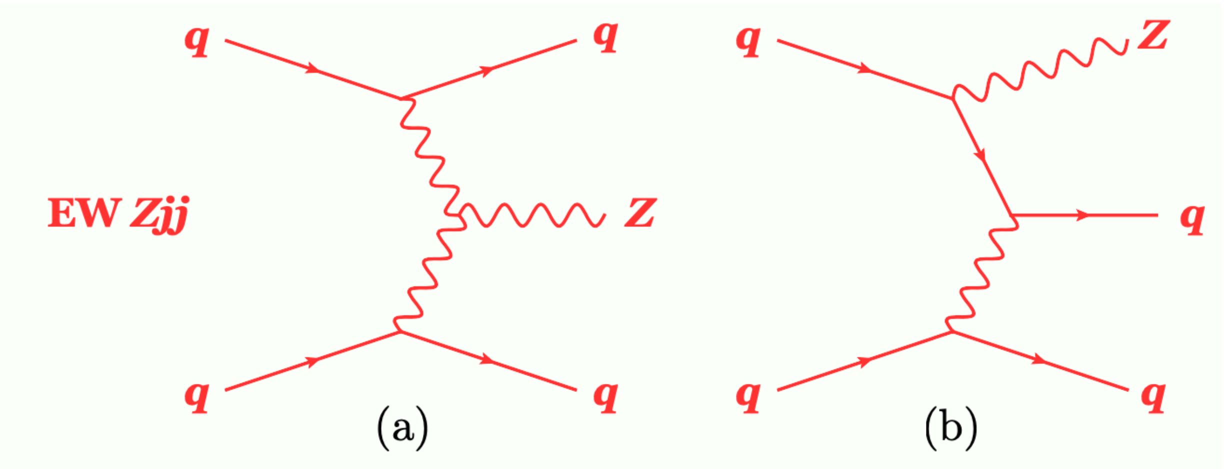
is

2. Correct for detect
3. Make data public

Experiment

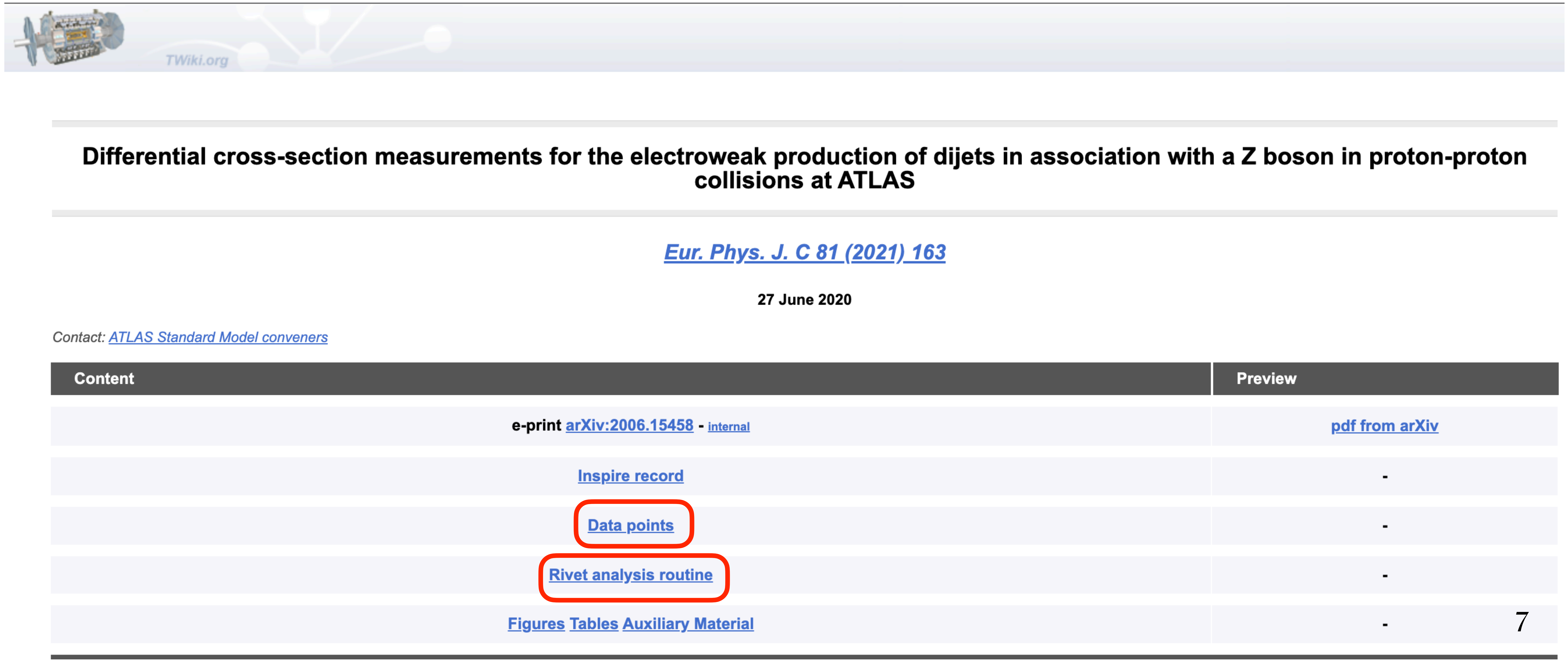
arXiv:2006.15458 (2020)

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

ATLAS public page, EPJC 81 (2021) 163



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

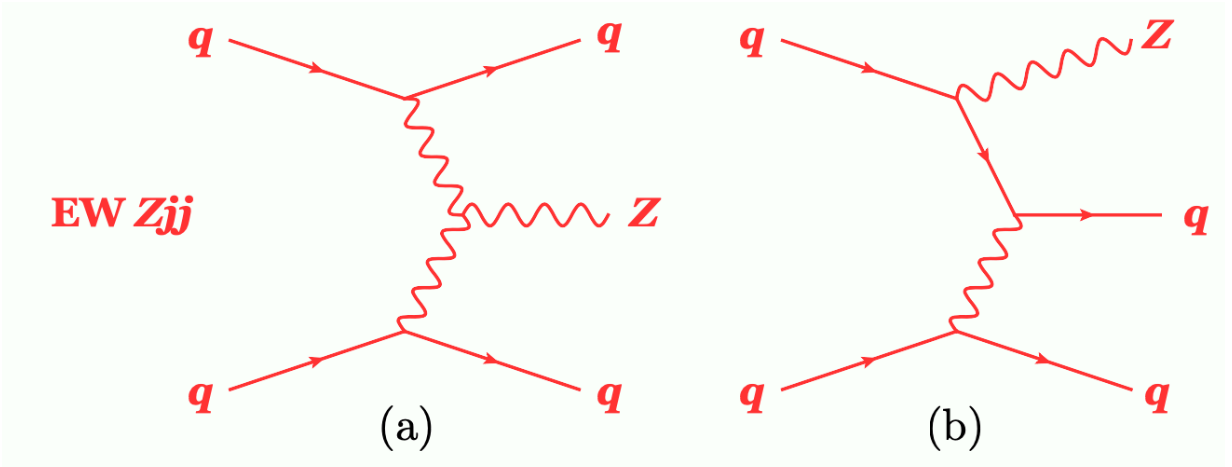
27 June 2020

Contact: ATLAS Standard Model conveners

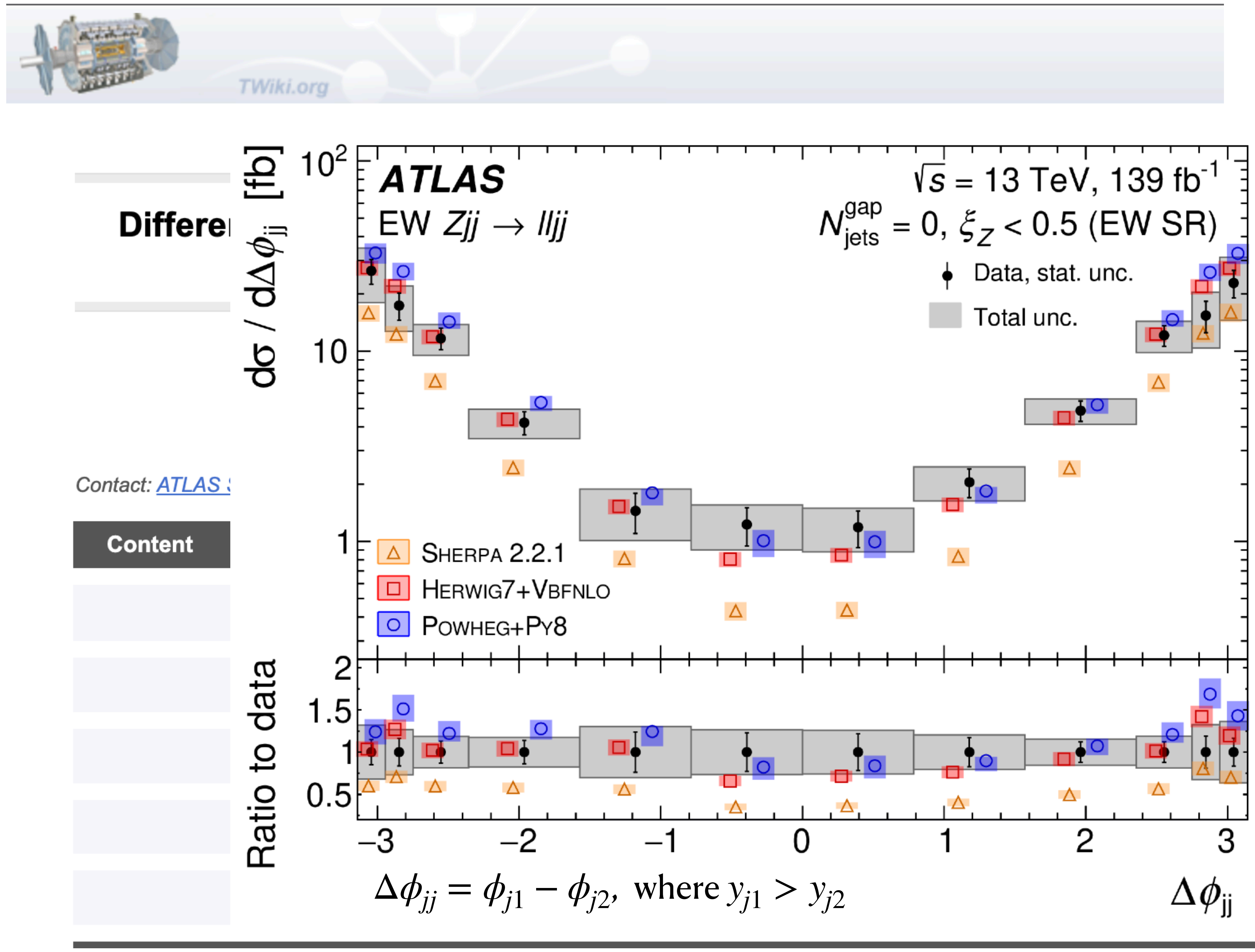
Content	Preview
e-print arXiv:2006.15458 - internal	pdf from arXiv
Inspire record	-
Data points	-
Rivet analysis routine	-
Figures Tables Auxiliary Material	-

[arXiv:2006.15458 (2020)](https://arxiv.org/abs/2006.15458)

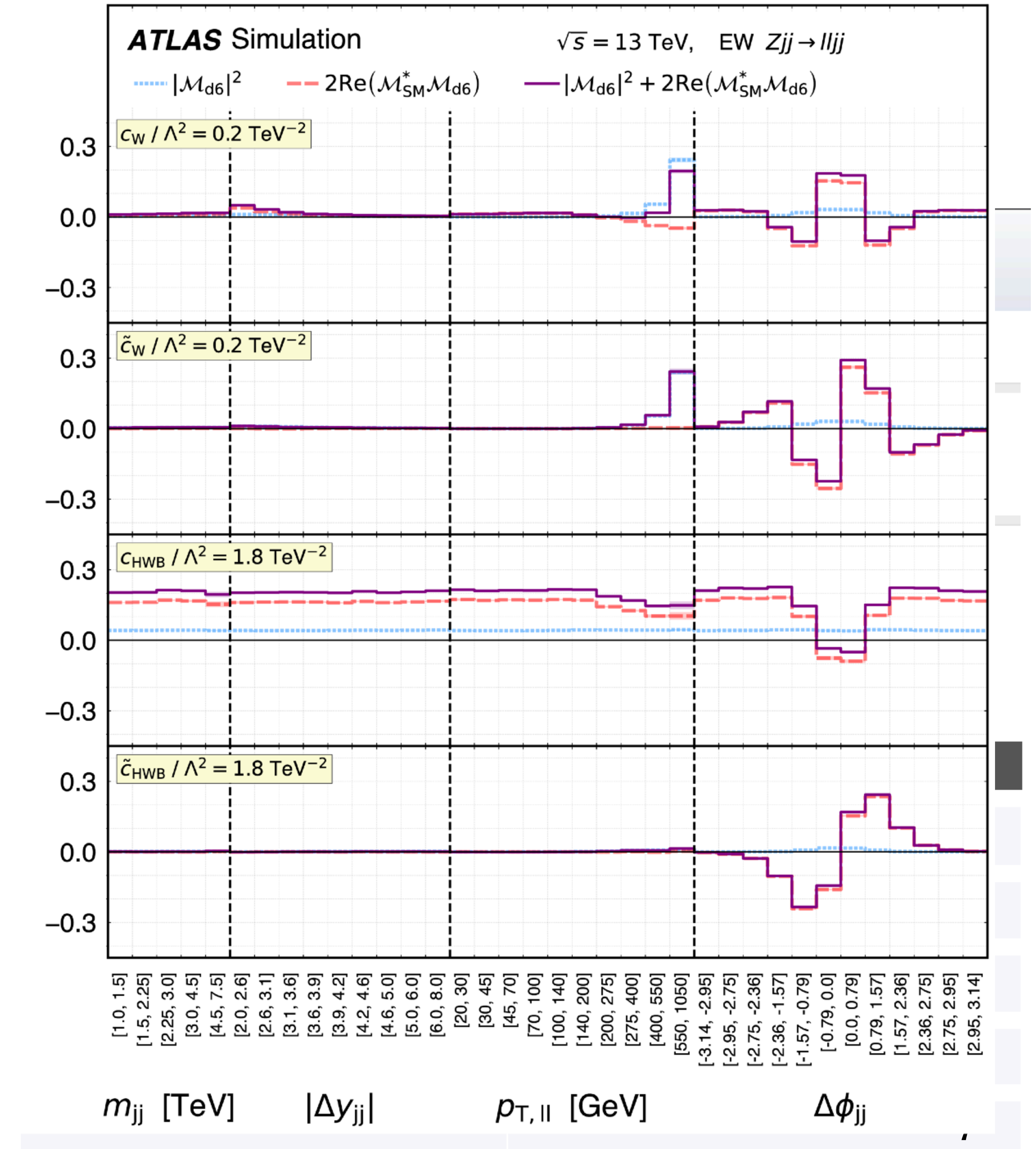
Useful tools at hand



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

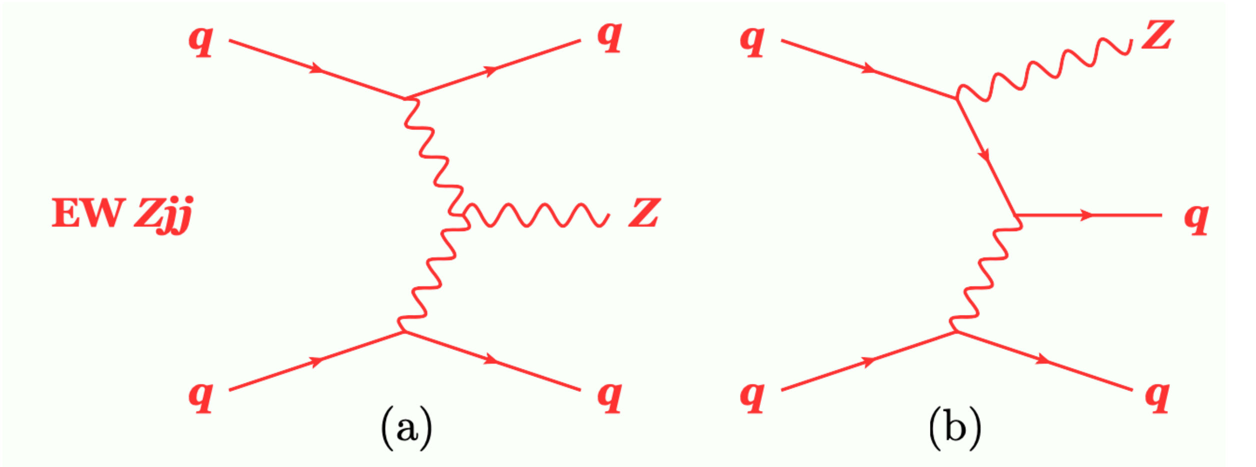


Impact from BSM modifications on the measured EW Zjj differential cross sections



arXiv:2006.15458 (2020)

Useful tools at hand



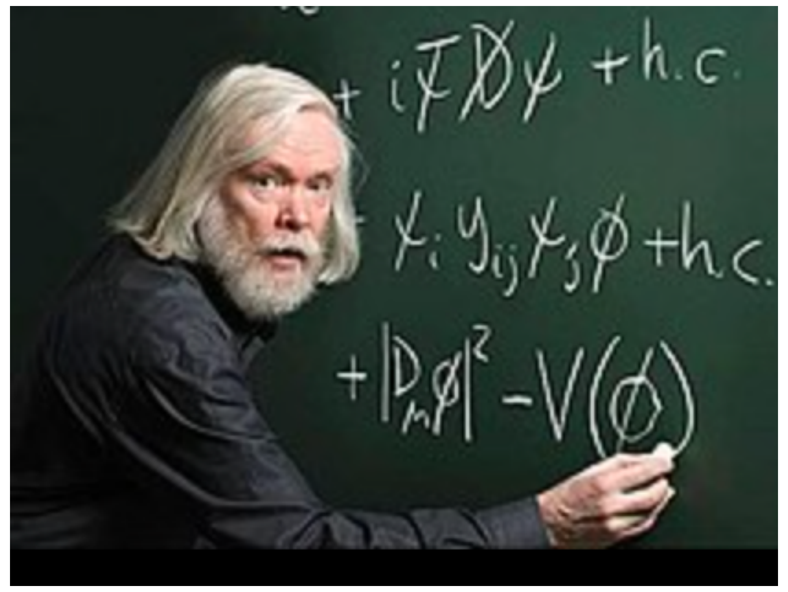
- HepData stores the measurements with associated uncertainties
  - hepdata.net
- Rivet is synchronized with the HepData entry
  - Ensures predictions defined in accordance with the data

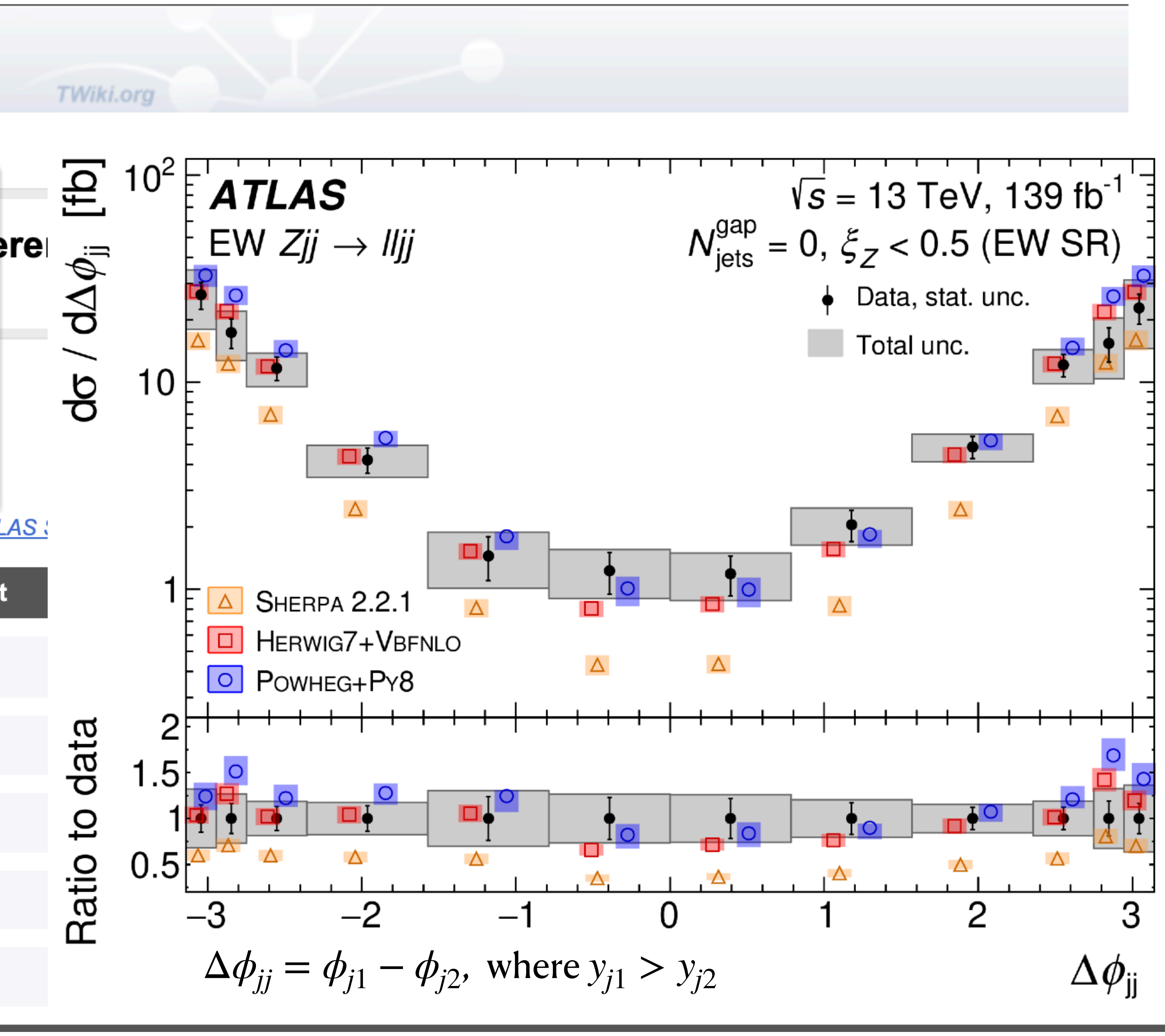
Impact from BSM modifications on the measured EW Zjj differential cross sections

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]

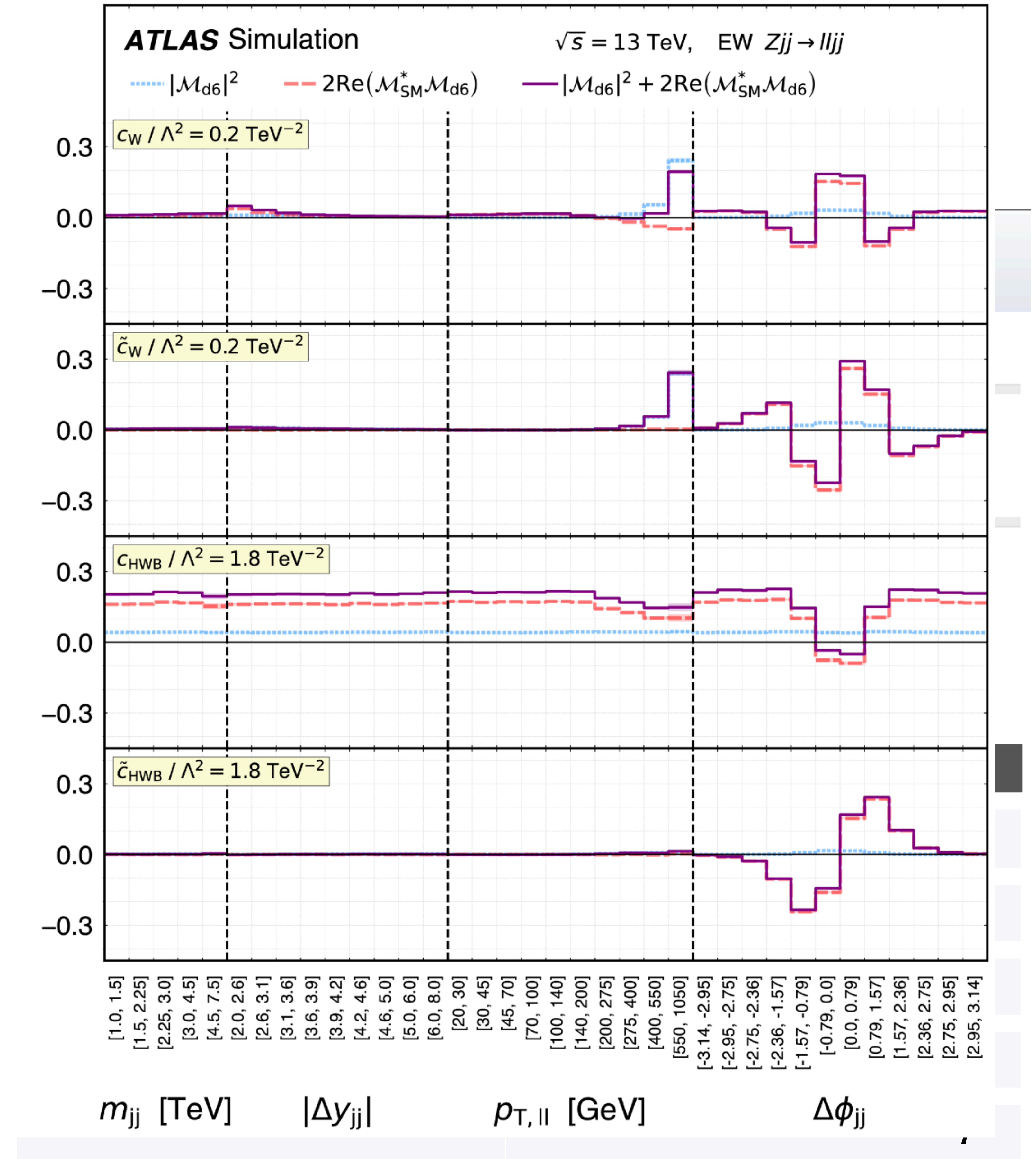
arXiv:2012.02779, JHEP 04 (2021) 279





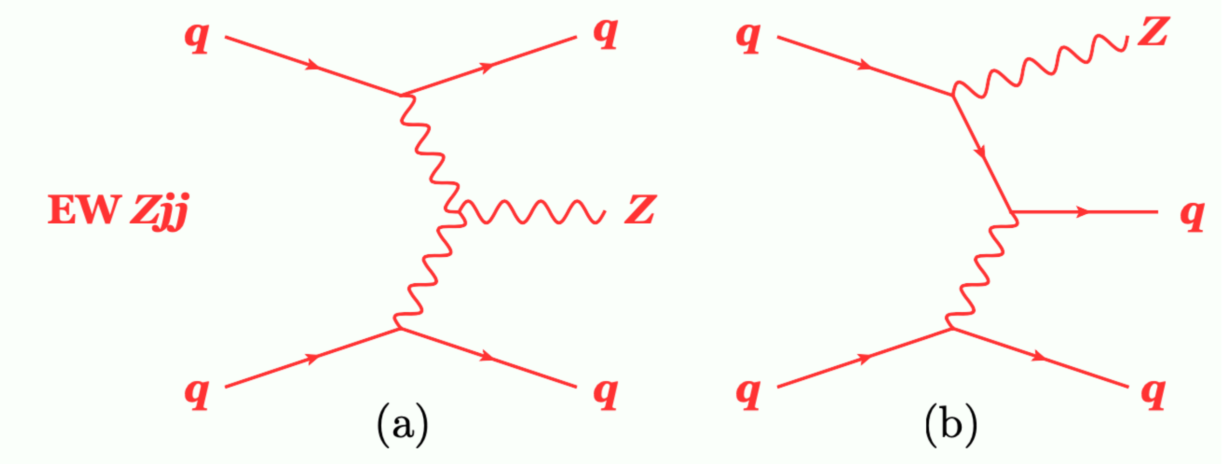
Contact: ATLAS

Content

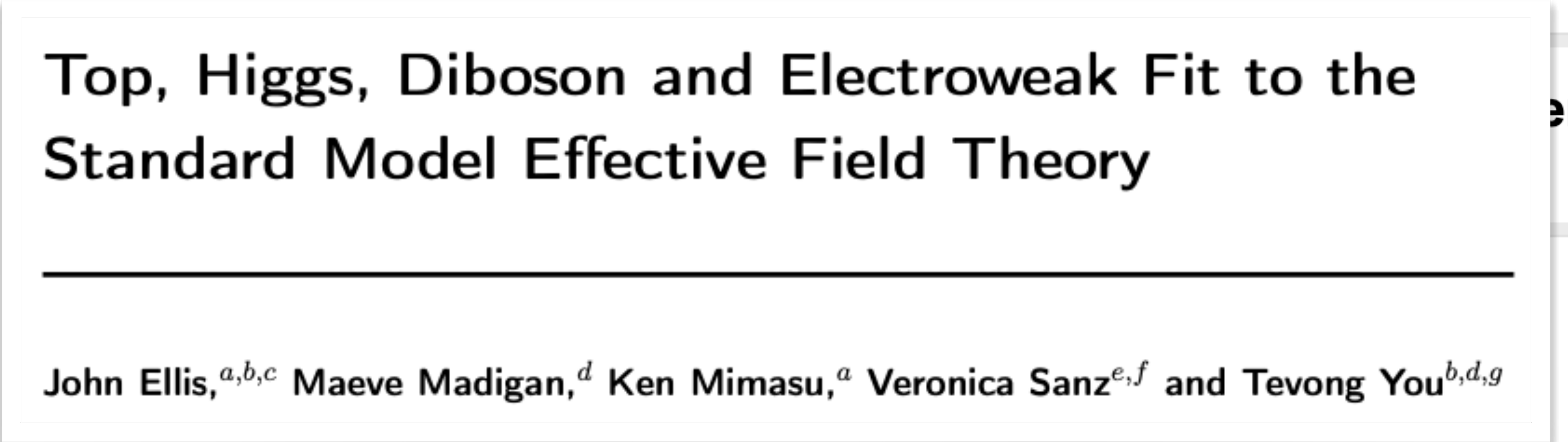


arXiv:2006.15458 (2020)

Useful tools at hand



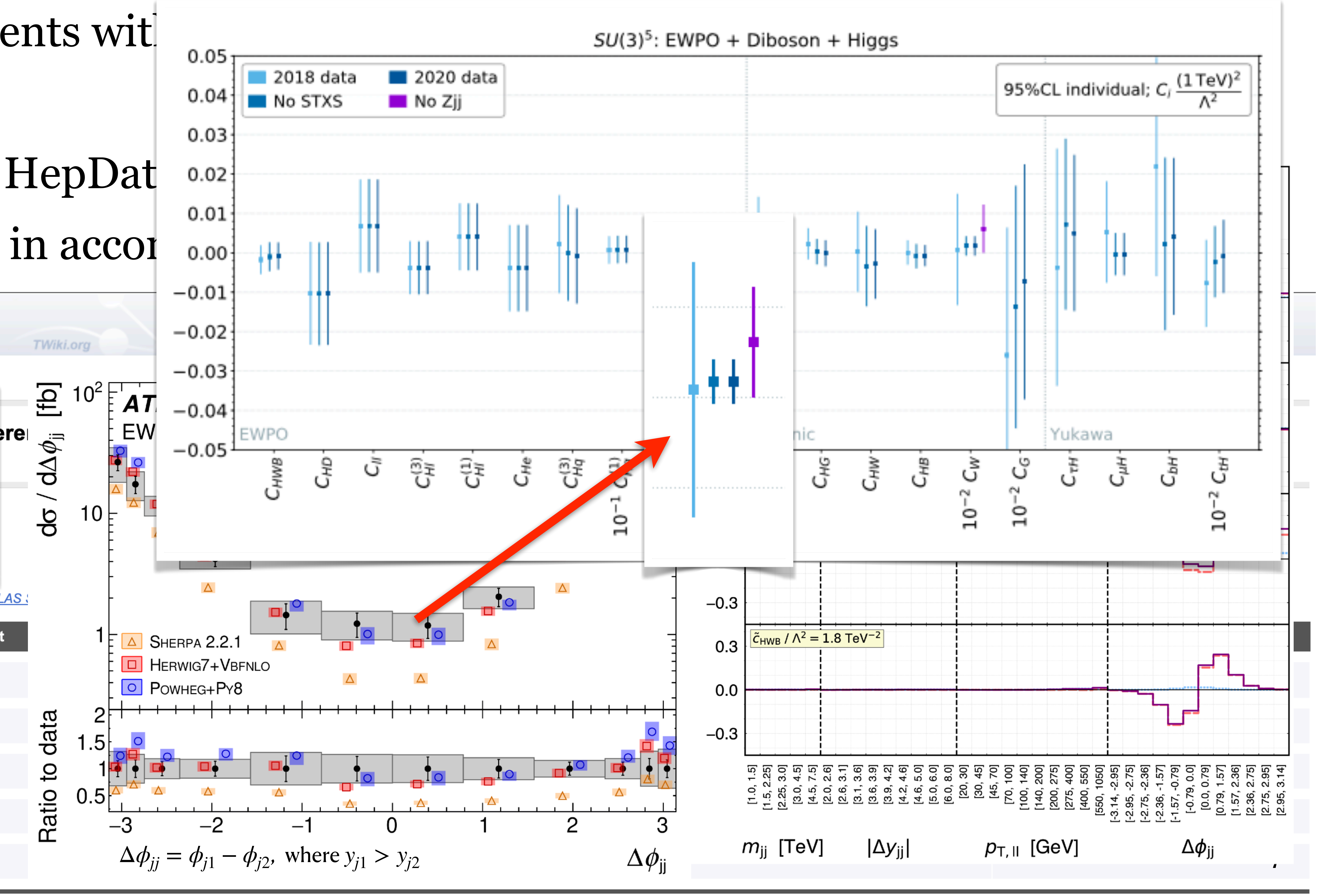
- HepData stores the measurements wit
  - hepdata.net
- Rivet is synchronized with the HepDat
  - Ensures predictions defined in accor



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



Limitations with current approach

- As we have seen, current approach for precision measurements is quite nice
- However there are several 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

  D. Physics analyses are very much internal to the experiment and take very long time; Paper + analysis contain

Recent developments in machine learning opens up new possibilities

Limitations with current approach

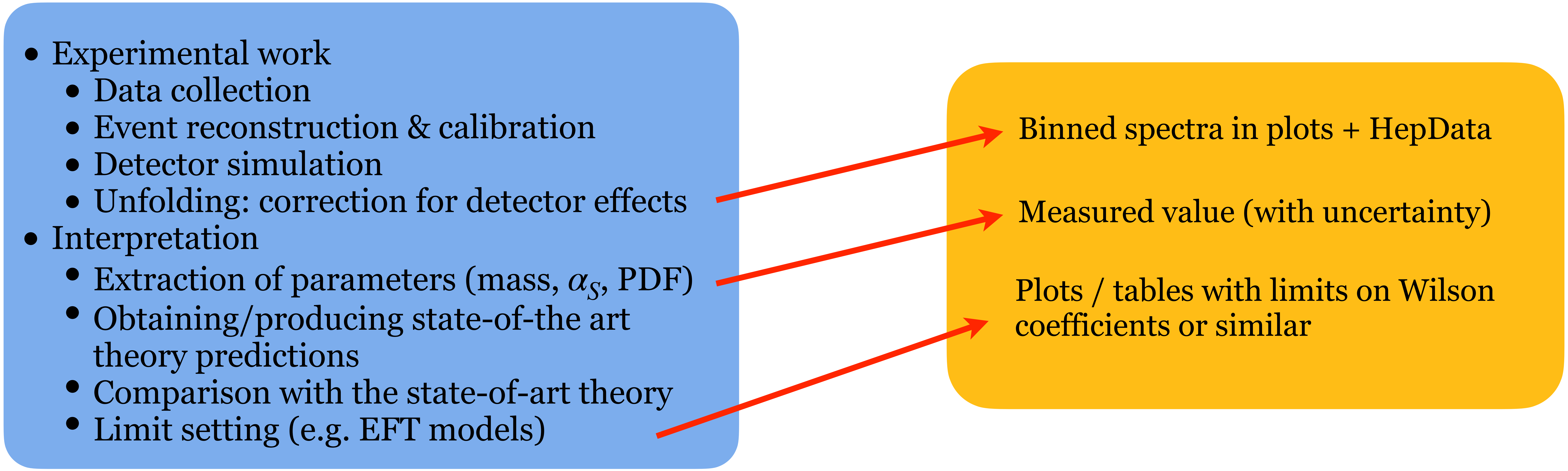
- As we have seen, current approach for precision measurements is quite nice
- However there are several short-comings
- When designing our measurement, we need to a-priori settle on
  A. Exact list of observables to measure **User can combine measured variables**
  B. Bin-boundaries for each measurement **Unbinned**
  C. We are limited to measure one (or a few) observables at the time **High dimensionality**
  D. Physics analyses are very much internal to the experiment and take very long time; Paper + analysis contain

Recent developments in machine learning opens up new possibilities

Precision measurements: The current approach

Experiment (ATLAS, CMS, …)

Public results (‘open data’)



- Experimental work
  - Data collection
  - Event reconstruction & calibration
  - Detector simulation
  - Unfolding: correction for detector effects
- Interpretation
  - Extraction of parameters (mass, α_S, PDF)
  - Obtaining/producing state-of-the art theory predictions
  - Comparison with the state-of-art theory
  - Limit setting (e.g. EFT models)

Binned spectra in plots + HepData

Measured value (with uncertainty)

Plots / tables with limits on Wilson coefficients or similar

Precision measurements are ***a lot of work***!
Usually takes ***many years*** to complete → big paper
During these years, analysis is ***internal to experiment*** (‘closed data’)

For scientific community can use and analyze

Open data

- Key results from precision measurements are typically presented in HepData
  - Differential and fiducial cross sections with fixed binning and covariance
  - Other measured quantities with uncertainties

Particle level

Good!
But sparse ‘data’

- Event datasets are rarely made public
- Actual data events are very easy to use for non-expert as they contain:
  - Noise & pileup & sometimes dead detector regions
  - ‘Kinks’ due to bin-edges in calibration functions
  - Mis-reconstructed or mis-identified objects
  - Detector inefficiencies, which are very different depending on the object
- These effects depend heavily on the detector (ATLAS ≠ CMS ≠ LHCb)

Detector level

Not so good …
Error prone

Precision measurements: Future approach(?)

Experiment (ATLAS, CMS, …)

- Experimental work
  - Data collection
  - Event reconstruction & calibration
  - Detector simulation
  - ML-based unfolding:
    correction for detector effects

Less work than current approach.
Only experimental → paper
Still ***internal*** to experiment
(Curate data)

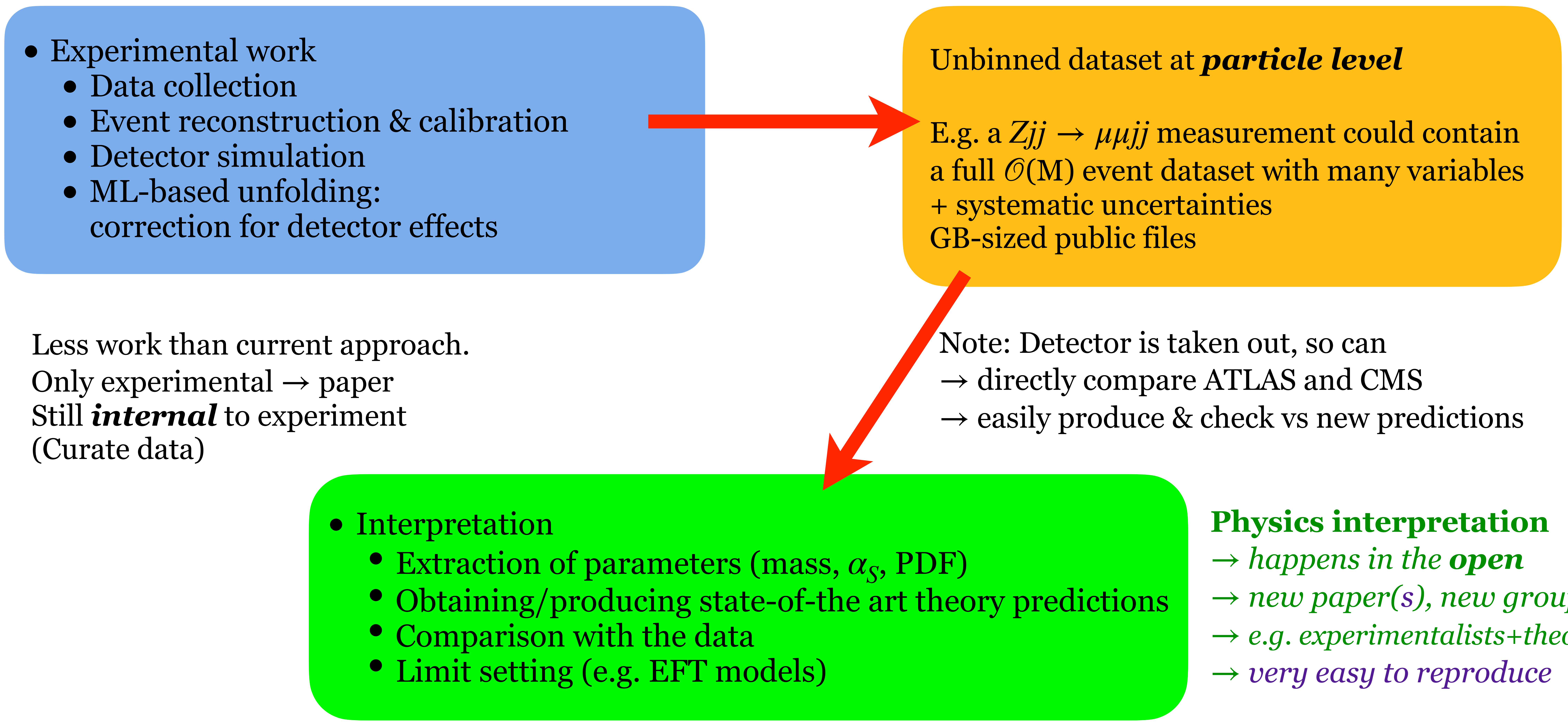
Public results ('open data')

Unbinned dataset at ***particle level***

E.g. a $Zjj \to \mu\mu jj$ measurement could contain
a full $\mathcal{O}$(M) event dataset with many variables
+ systematic uncertainties
GB-sized public files

Note: Detector is taken out, so can
→ directly compare ATLAS and CMS
→ easily produce & check vs new predictions

Precision measurements: Future approach(?)



Experiment (ATLAS, CMS, …)

- Experimental work
  - Data collection
  - Event reconstruction & calibration
  - Detector simulation
  - ML-based unfolding: correction for detector effects

Public results (‘open data’)

Unbinned dataset at ***particle level***

E.g. a $Zjj \to \mu\mu jj$ measurement could contain
a full $\mathcal{O}$(M) event dataset with many variables
+ systematic uncertainties
GB-sized public files

Less work than current approach.
Only experimental → paper
Still ***internal*** to experiment
(Curate data)

Note: Detector is taken out, so can
→ directly compare ATLAS and CMS
→ easily produce & check vs new predictions

- Interpretation
  - Extraction of parameters (mass, α_S, PDF)
  - Obtaining/producing state-of-the art theory predictions
  - Comparison with the data
  - Limit setting (e.g. EFT models)

Physics interpretation
→ *happens in the **open***
→ *new paper(s), new group(s)*
→ *e.g. experimentalists+theorists*
→ *very easy to reproduce*

Modern ML method for unbinned measurements

- New possibilities have opened up with the advanced of ML
- A lot of interest from the particle physics community — also from the precision community
- Several proofs of principle papers released that take ***two main approaches***

Discriminative models
'Density reweighing'

Publish MC events with weights to match data

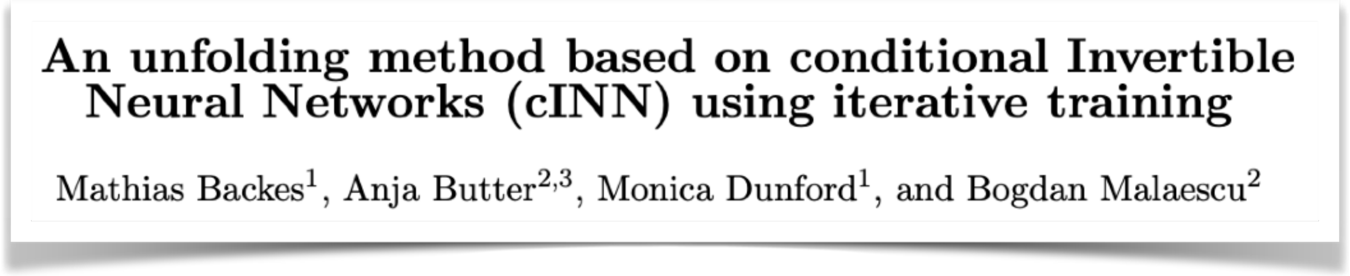
Example the OmniFold method.
Already used for real measurement:
H1
ATLAS
CMS
LHCb

Note: only the ATLAS measurement (Laura Miller's PhD) has released the unpinned data to the public. The other only use method as internal stepping stone.

Generative models
'Density reweighing'

Publish 'data events width widths'

Not used for real data yet (as far as I know)



An unfolding method based on conditional Invertible Neural Networks (cINN) using iterative training

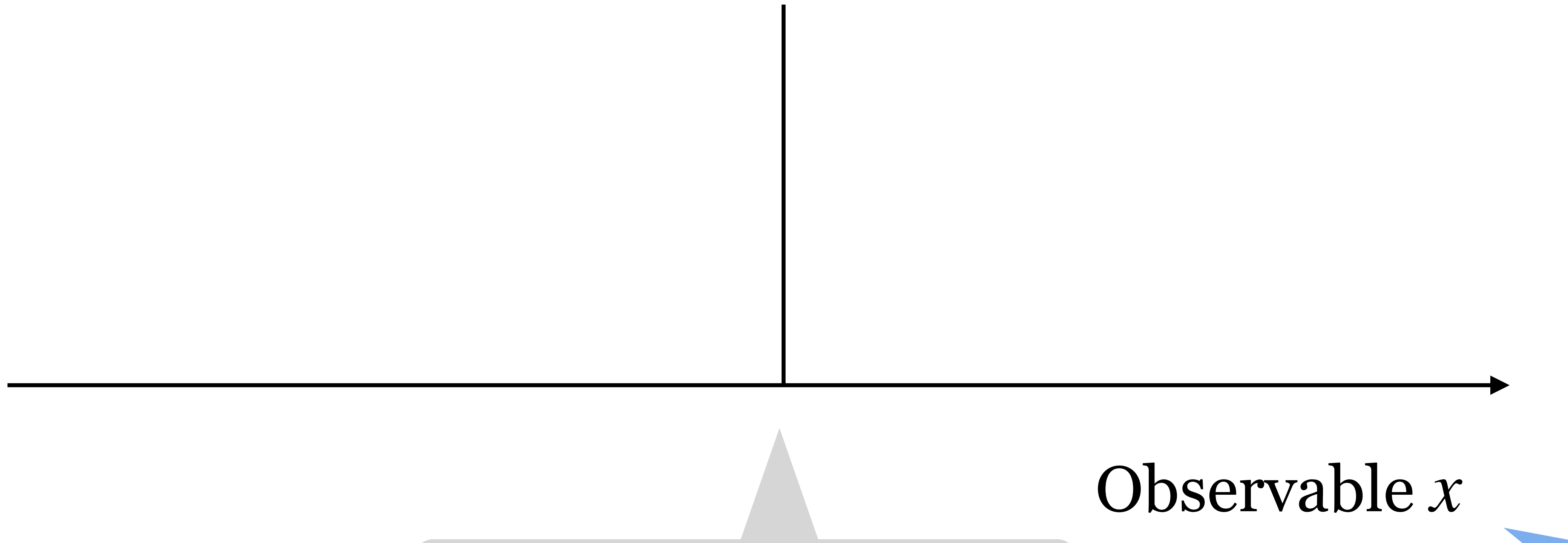
Mathias Backes[1], Anja Butter[2,3], Monica Dunford[1], and Bogdan Malaescu[2]

Principle of generative models

An unfolding method based on conditional Invertible Neural Networks (cINN) using iterative training

Mathias Backes[1], Anja Butter[2,3], Monica Dunford[1], and Bogdan Malaescu[2]

Example paper — proof of principle with MC
[arXiv:2212.08674](https://arxiv.org/abs/2212.08674)

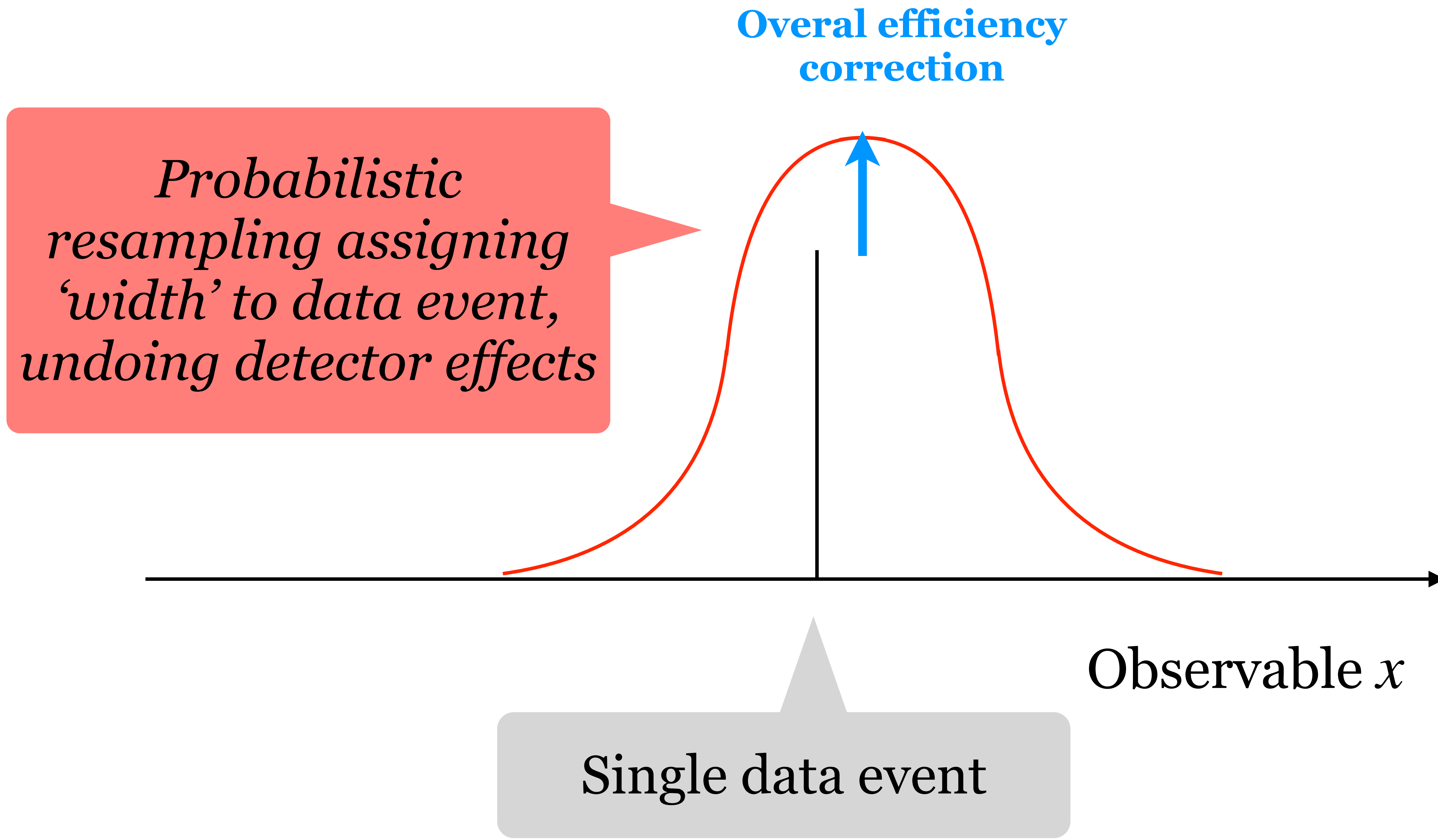


Observable x

Single data event

Only one observables shown, but this happens for many at the same time

Principle of generative models



Overal efficiency correction

Probabilistic resampling assigning 'width' to data event, undoing detector effects

Single data event

Observable x

An unfolding method based on conditional Invertible Neural Networks (cINN) using iterative training

Mathias Backes[1], Anja Butter[2,3], Monica Dunford[1], and Bogdan Malaescu[2]

Example paper — proof of principle with MC
arXiv:2212.08674

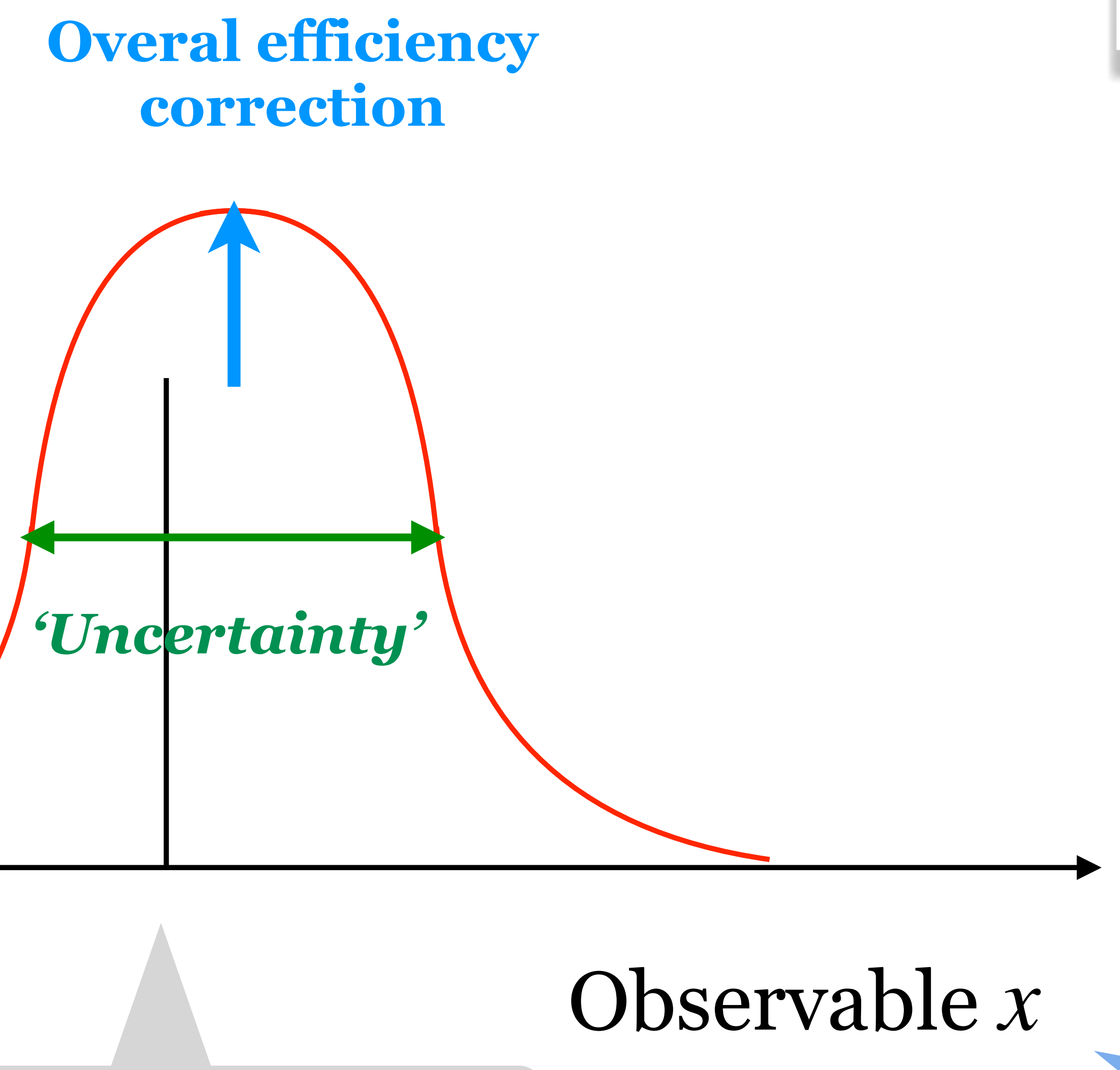
Only one observables shown, but this happens for many at the same time

Principle of generative models

An unfolding method based on conditional Invertible Neural Networks (cINN) using iterative training

Mathias Backes[1], Anja Butter[2,3], Monica Dunford[1], and Bogdan Malaescu[2]

Example paper — proof of principle with MC
[arXiv:2212.08674](https://arxiv.org/abs/2212.08674)



Probabilistic resampling assigning 'width' to data event, undoing detector effects

Single data event

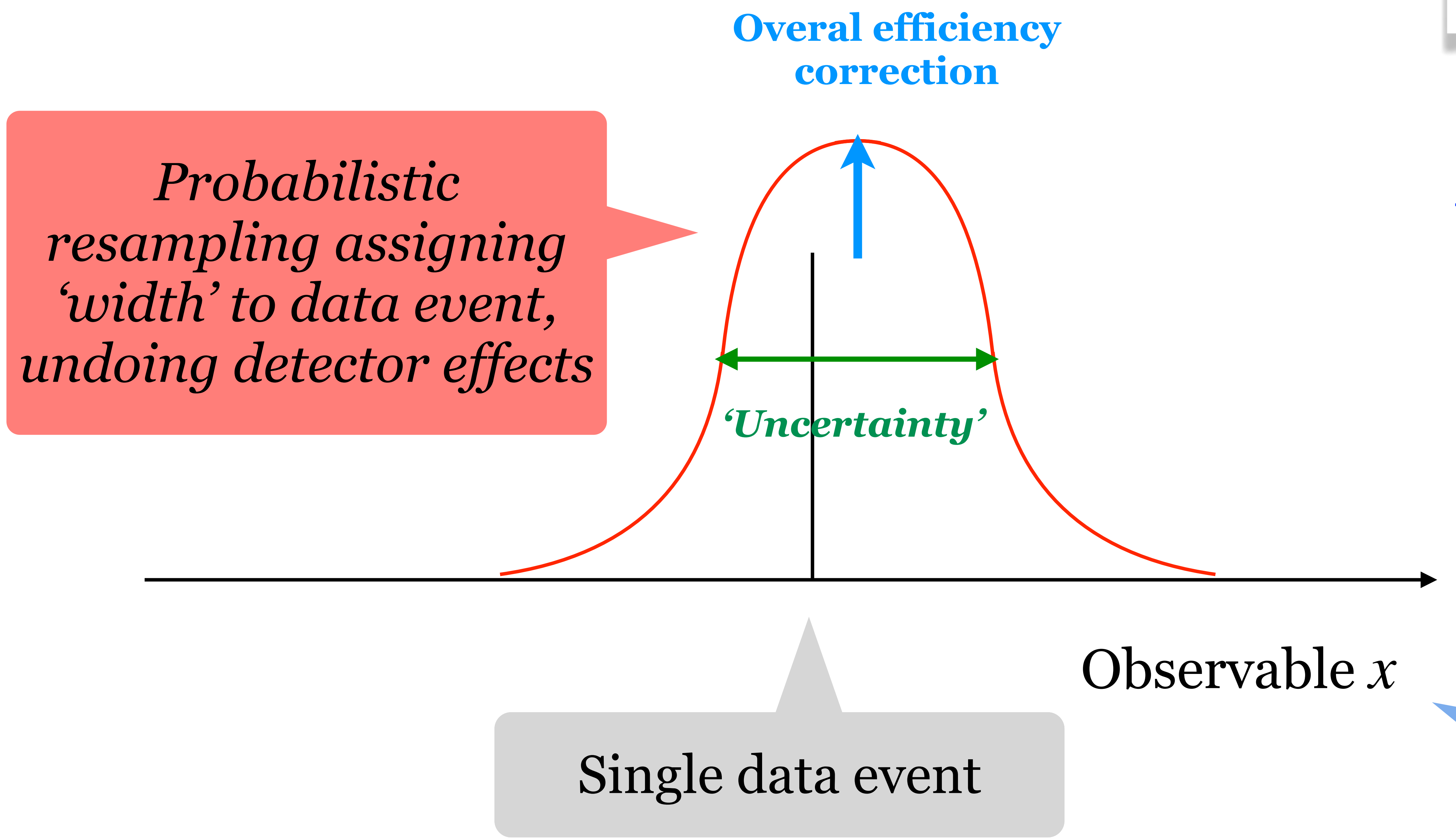
Only one observables shown, but this happens for many at the same time

Principle of generative models

An unfolding method based on conditional Invertible Neural Networks (cINN) using iterative training

Mathias Backes[1], Anja Butter[2,3], Monica Dunford[1], and Bogdan Malaescu[2]

Example paper — proof of principle with MC
[arXiv:2212.08674](https://arxiv.org/abs/2212.08674)



Overal efficiency correction

Probabilistic resampling assigning ‘width’ to data event, undoing detector effects

‘Uncertainty’

Observable x

Single data event

Only one observables shown, but this happens for many at the same time

Hence, each observed data event generates collection of new events that gives it a multidimensional width.

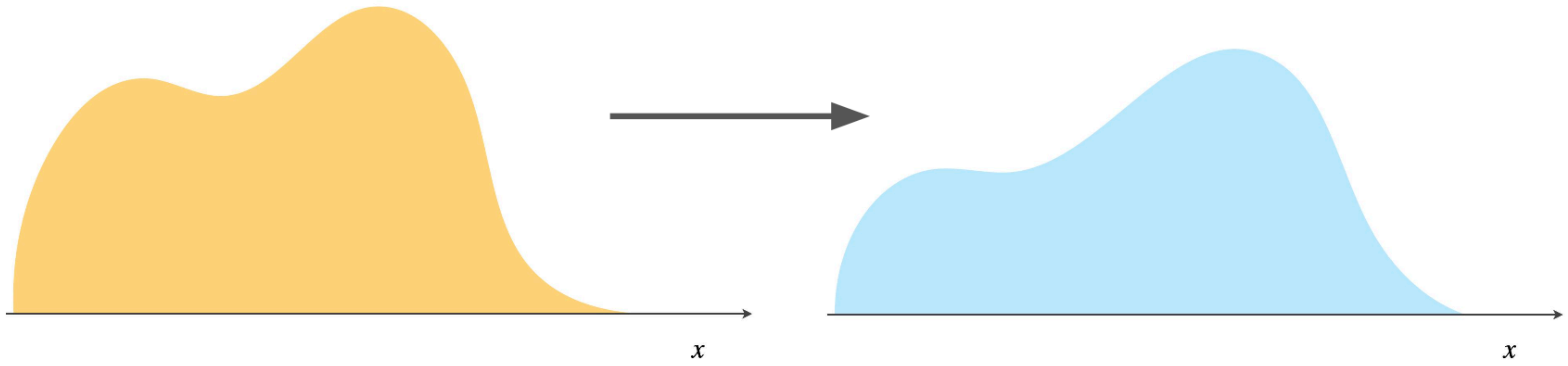
1 data even → ~100 published events

1M data events → ~100M published events + more for other uncertainties

Plotting an observable for all events will give a differential cross section.

Principle of discriminative approach *aka density reweighing*

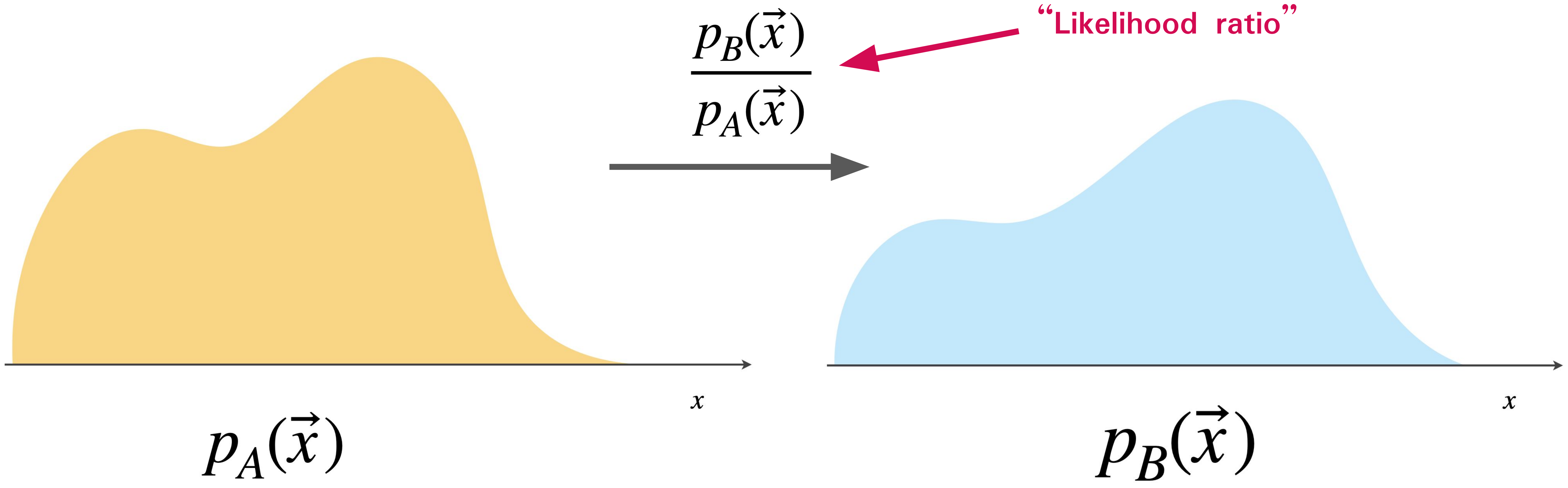
Q: How can we adjust one distribution to look like another?



Principle of discriminative approach *aka density reweighing*

Q: How can we adjust one distribution to look like another?

A: Learn a reweighting function based on the ratio of their probability densities.



$$\frac{p_B(\vec{x})}{p_A(\vec{x})}$$

"Likelihood ratio"

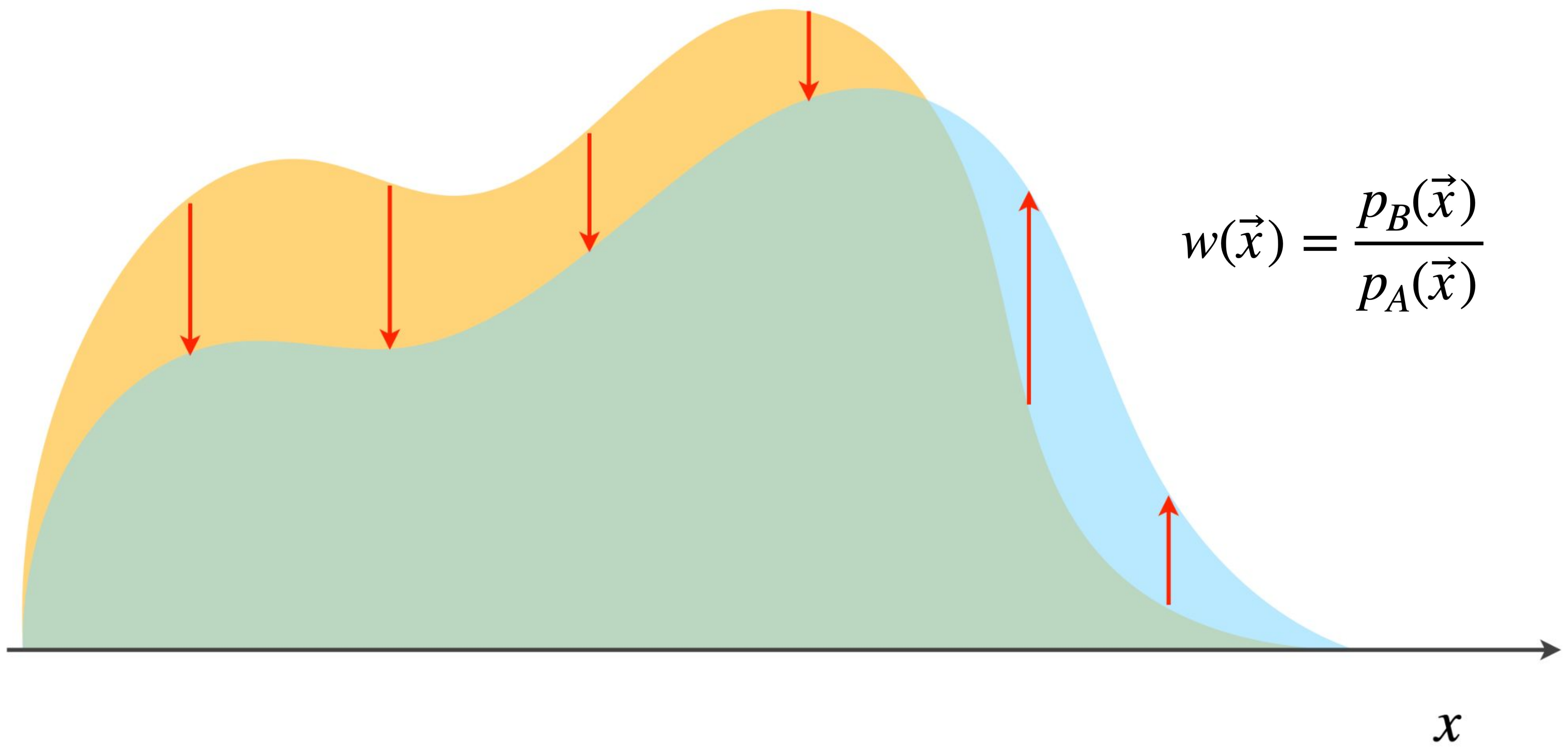
$p_A(\vec{x})$

$p_B(\vec{x})$

Principle of discriminative approach *aka density reweighing*

Q: How can we adjust one distribution to look like another?

A: Learn a reweighting function based on the ratio of their probability densities.



Principle of discriminative approach *aka density reweighing*

How do we use NNs to learn the likelihood ratio?

It's actually quite straight forward …

We can train a **classifier** $f(\vec{x})$ and use the it for this.

For NNs, need to use the cross entropy as loss function such that the NN output (= the classification score score $f(\vec{x})$) has the right meaning

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

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

Principle of discriminative approach *aka density reweighing*

How do we use NNs to learn the likelihood ratio?

It's actually quite straight forward …

We can train a **classifier** $f(\vec{x})$ and use the it for this.

For NNs, need to use the cross entropy as loss function such that the NN output (= the classification score score $f(\vec{x})$) has the right meaning

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

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

Then the NN output with approximate the 'purity':

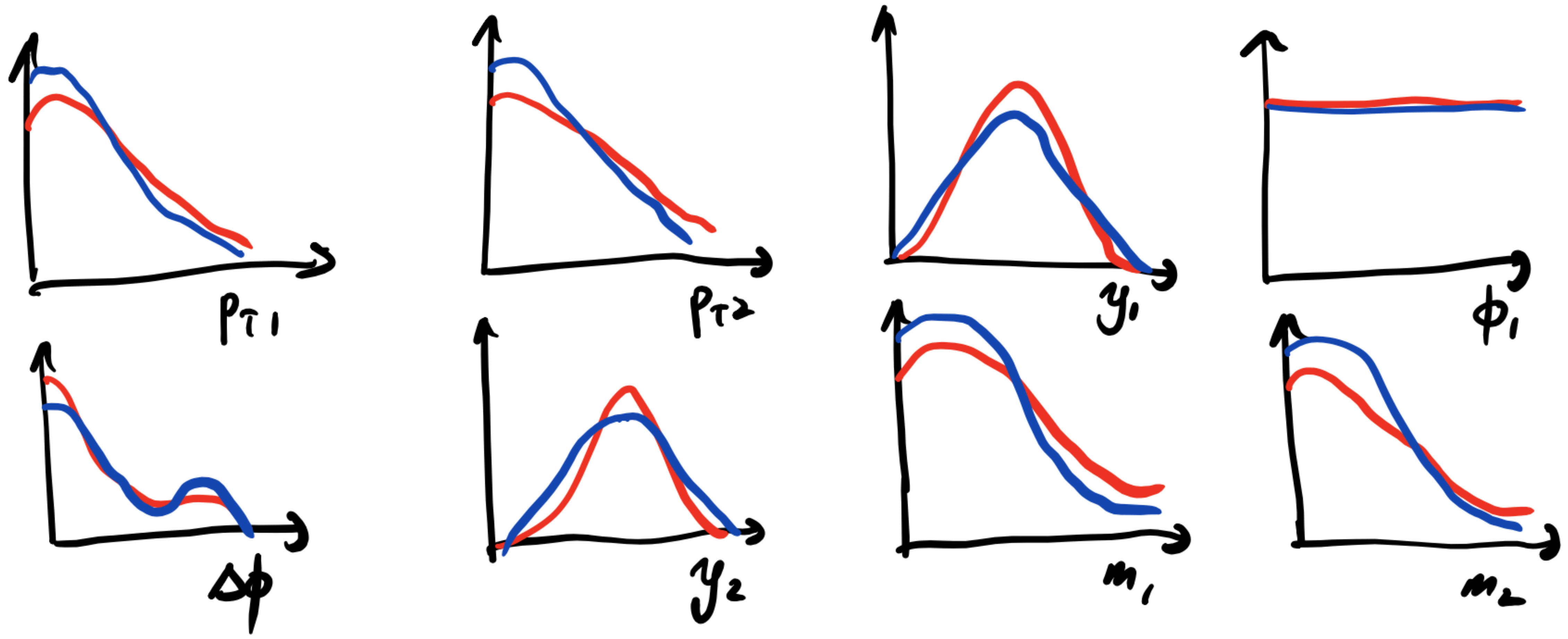
$$f(\vec{x}) \approx \frac{p_s(\vec{x})}{p_s(\vec{x}) + p_b(\vec{x})}$$

The likelihood ratio (density reweighs):

$$\frac{p_s(\vec{x})}{p_b(\vec{x})} \approx \frac{f(\vec{x})}{1 - f(\vec{x})}$$

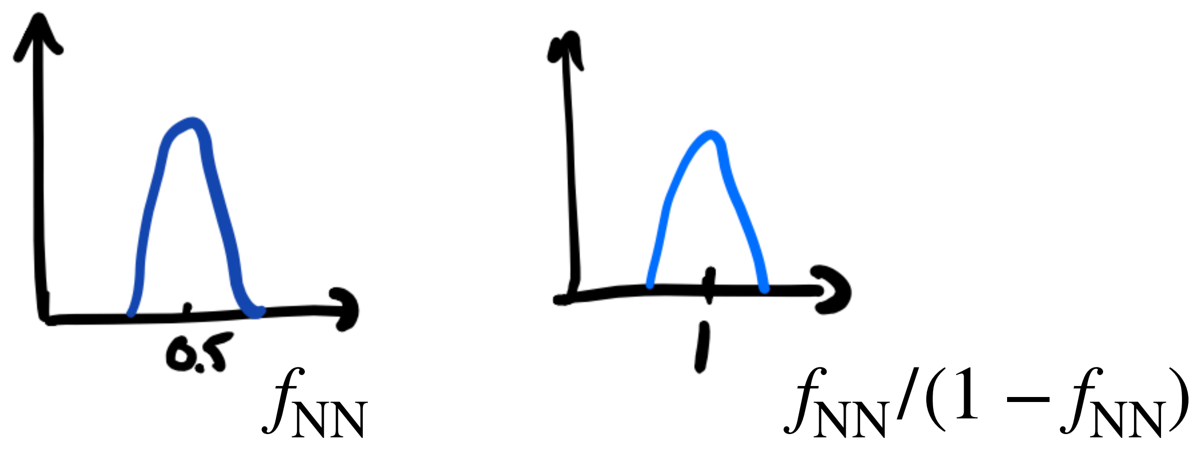
Using ML to reweight event samples

- 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 $\vec{x} = (x_1, \ldots, x_8)$



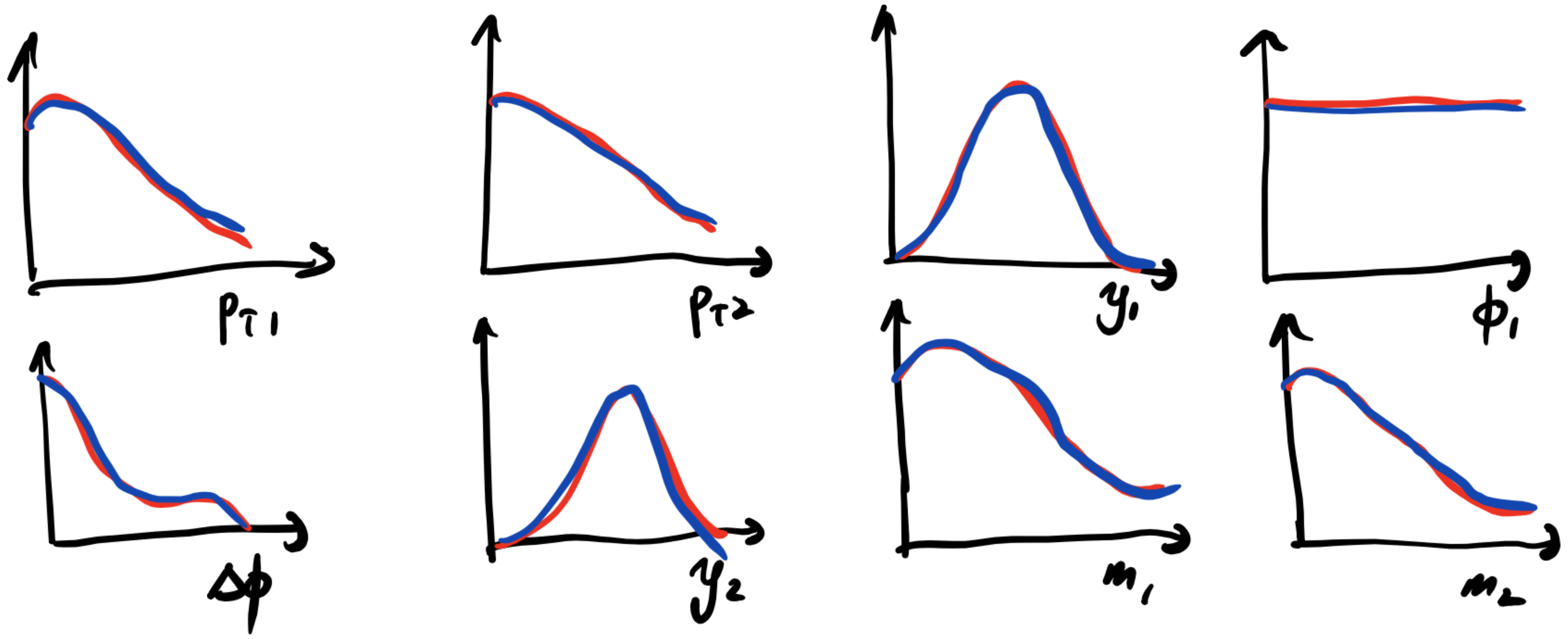
A neural network trained with cross entropy as loss function will return $f_{\mathrm{NN}}(\vec{x})$, that estimates the purity. An estimate of the likelihood ratio is given by

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



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

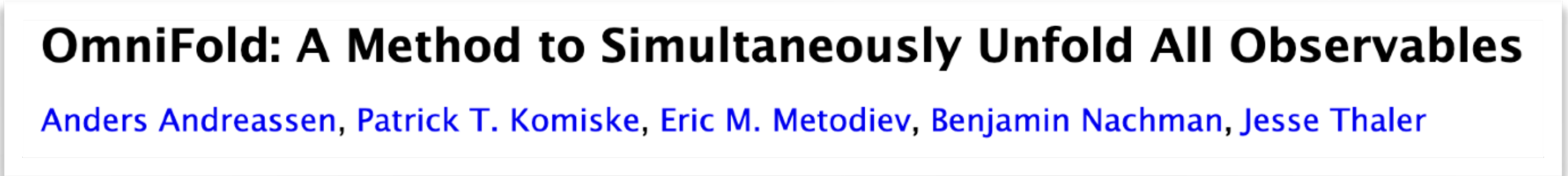
$$\hat{\lambda}(\vec{x}) = \frac{f_{\mathrm{NN}}(\vec{x})}{1 - f_{\mathrm{NN}}(\vec{x})}$$

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

The OmniFold method



OmniFold: A Method to Simultaneously Unfold All Observables

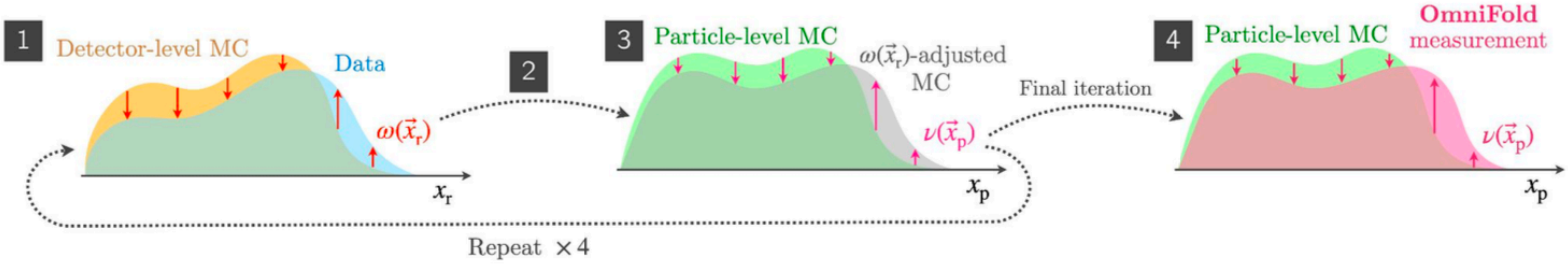
Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, Jesse Thaler

arXiv:1911.09107

The OmniFold procedure requires two datasets* as inputs:
(*In practice, we use samples from different MC generators as well as systematically-shifted samples to determine uncertainties.)

- MC sample with events at both detector-level and particle-level
- Real data
    - (In fact, it's the only unfolding method of this kind that has been applied to real data)

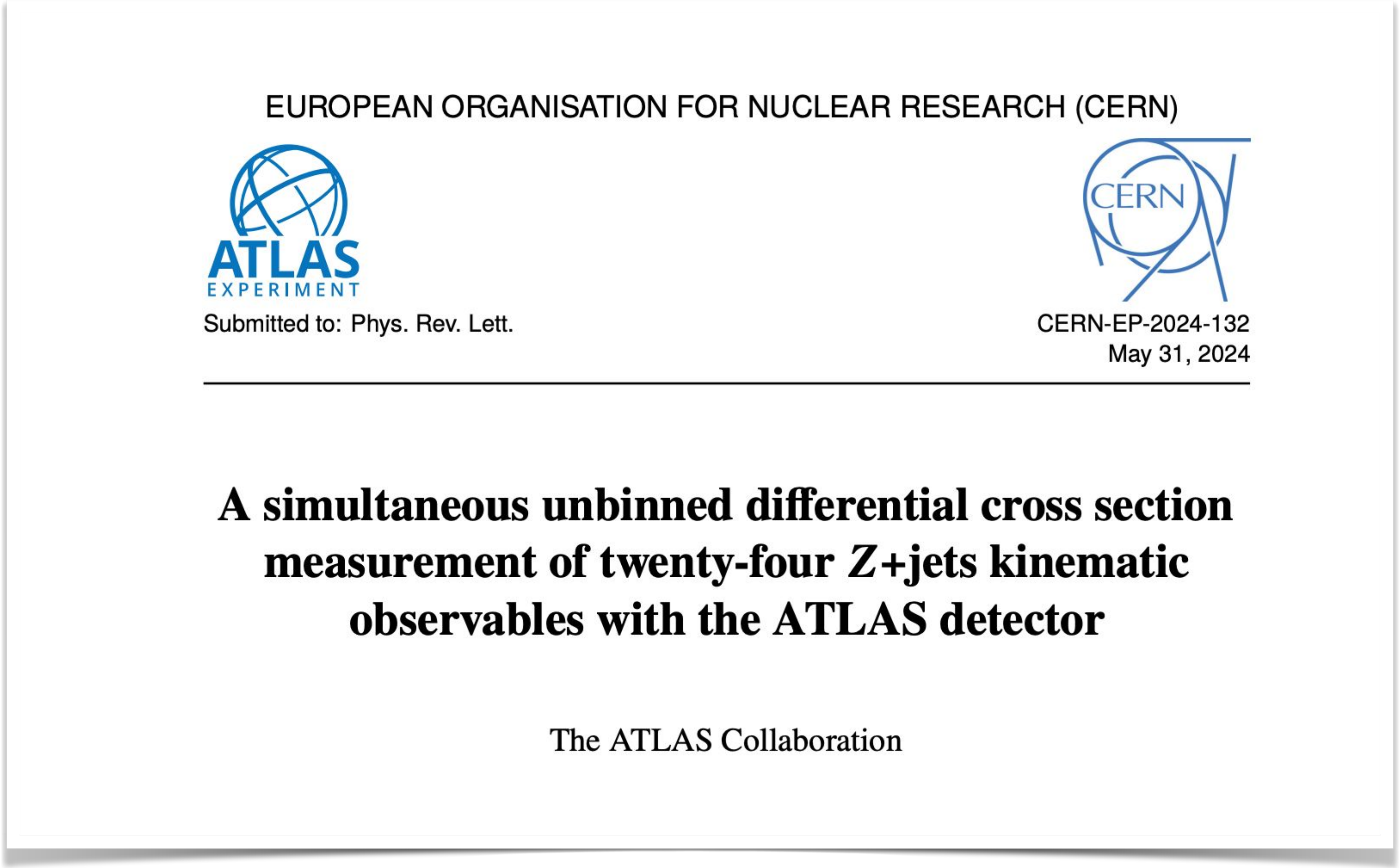
In a multi-stage and iterative process, a series of neural networks are trained to learn a reweighting function that maps particle-level MC distributions to particle-level data distributions.



Note: Binned OmniFold reduces to Iterative Bayesian Unfolding (IBU)!

arXiv:2405.20041

Concrete example


EUROPEAN ORGANISATION FOR NUCLEAR RESEARCH (CERN)

Submitted to: Phys. Rev. Lett.

CERN-EP-2024-132
May 31, 2024

A simultaneous unbinned differential cross section measurement of twenty-four Z+jets kinematic observables with the ATLAS detector

The ATLAS Collaboration

Process: $Z + \text{jets} \rightarrow \mu\mu + \text{jets}$

24 observables measured simultaneously:

- Leading & sub-leading jet: p_T, y, ϕ, τ_1, τ_2, τ_3, m, $n_{\text{charged tracks}}$
- Leading & sub-leading muon: p_T, η, ϕ
- Di-muon system: p_T, y

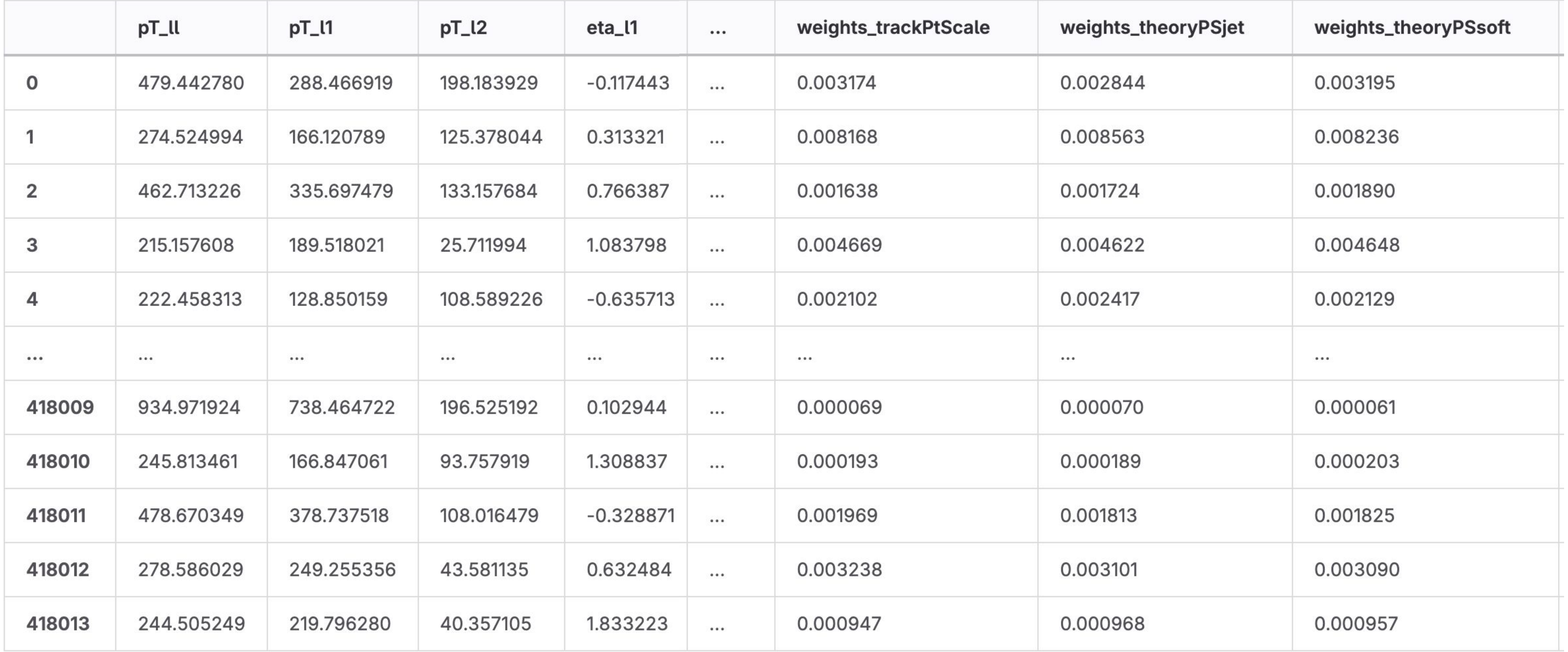
Laura Miller's PhD topic

Concrete example

The measurement — pubic data

Nominal event dataset

420k events



	pT_ll	pT_l1	pT_l2	eta_l1	...	weights_trackPtScale	weights_theoryPSjet	weights_theoryPSsoft
0	479.442780	288.466919	198.183929	-0.117443	...	0.003174	0.002844	0.003195
1	274.524994	166.120789	125.378044	0.313321	...	0.008168	0.008563	0.008236
2	462.713226	335.697479	133.157684	0.766387	...	0.001638	0.001724	0.001890
3	215.157608	189.518021	25.711994	1.083798	...	0.004669	0.004622	0.004648
4	222.458313	128.850159	108.589226	-0.635713	...	0.002102	0.002417	0.002129
...	...	...	...	...	...	...	...	...
418009	934.971924	738.464722	196.525192	0.102944	...	0.000069	0.000070	0.000061
418010	245.813461	166.847061	93.757919	1.308837	...	0.000193	0.000189	0.000203
418011	478.670349	378.737518	108.016479	-0.328871	...	0.001969	0.001813	0.001825
418012	278.586029	249.255356	43.581135	0.632484	...	0.003238	0.003101	0.003090
418013	244.505249	219.796280	40.357105	1.833223	...	0.000947	0.000968	0.000957

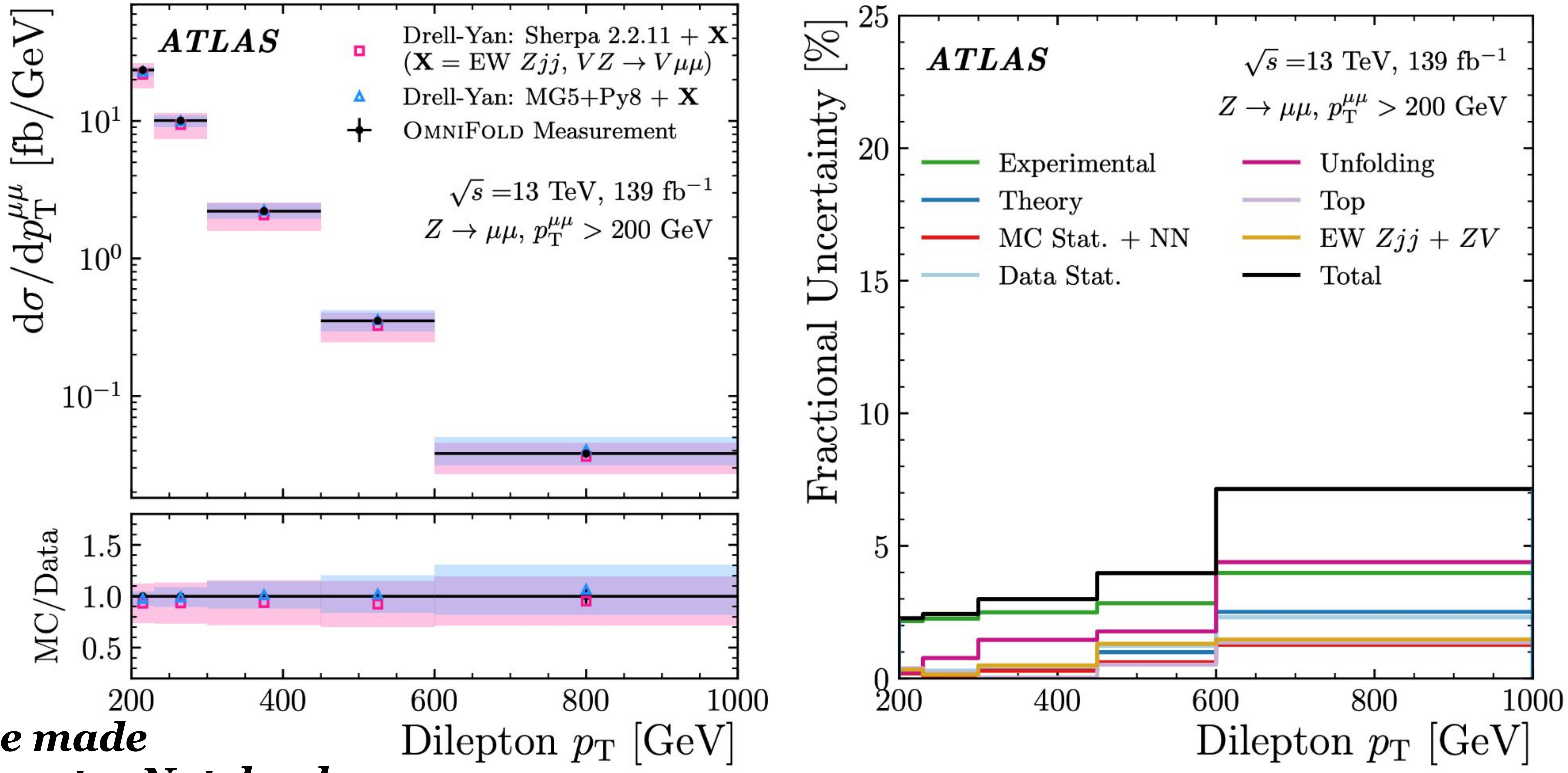
24 observables
(at particle level)

One **nominal weight**
Lots of alternative **weights** encoding **uncertainty**
Unit of weight is fb.

Concrete example
Plotting one variable

The measurement is a 24-dimensional object.
Let's check out two examples of measured differential cross-sections:

Dilepton p_T



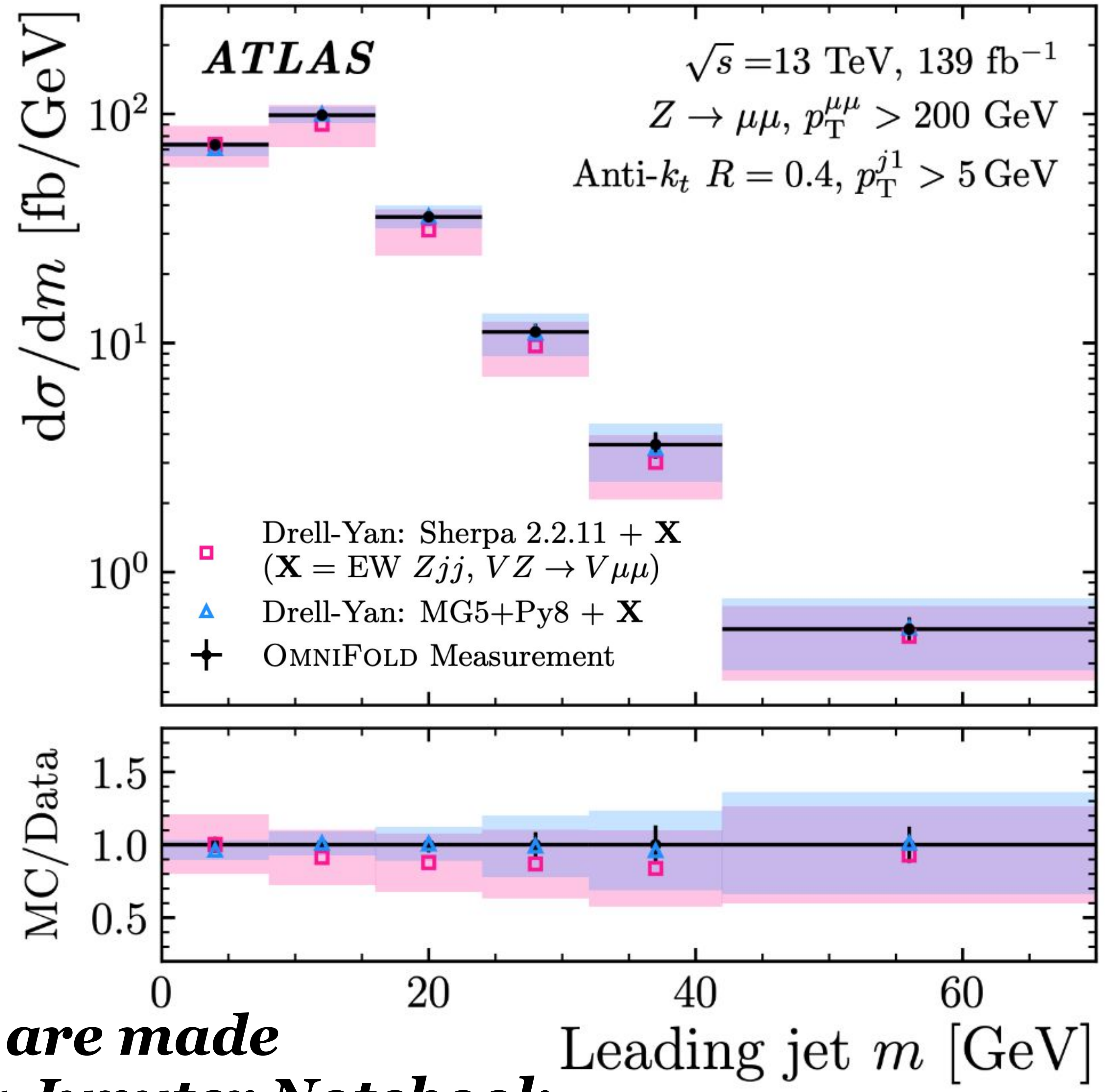
Note: these plots are made interactively in a Jupyter Notebook

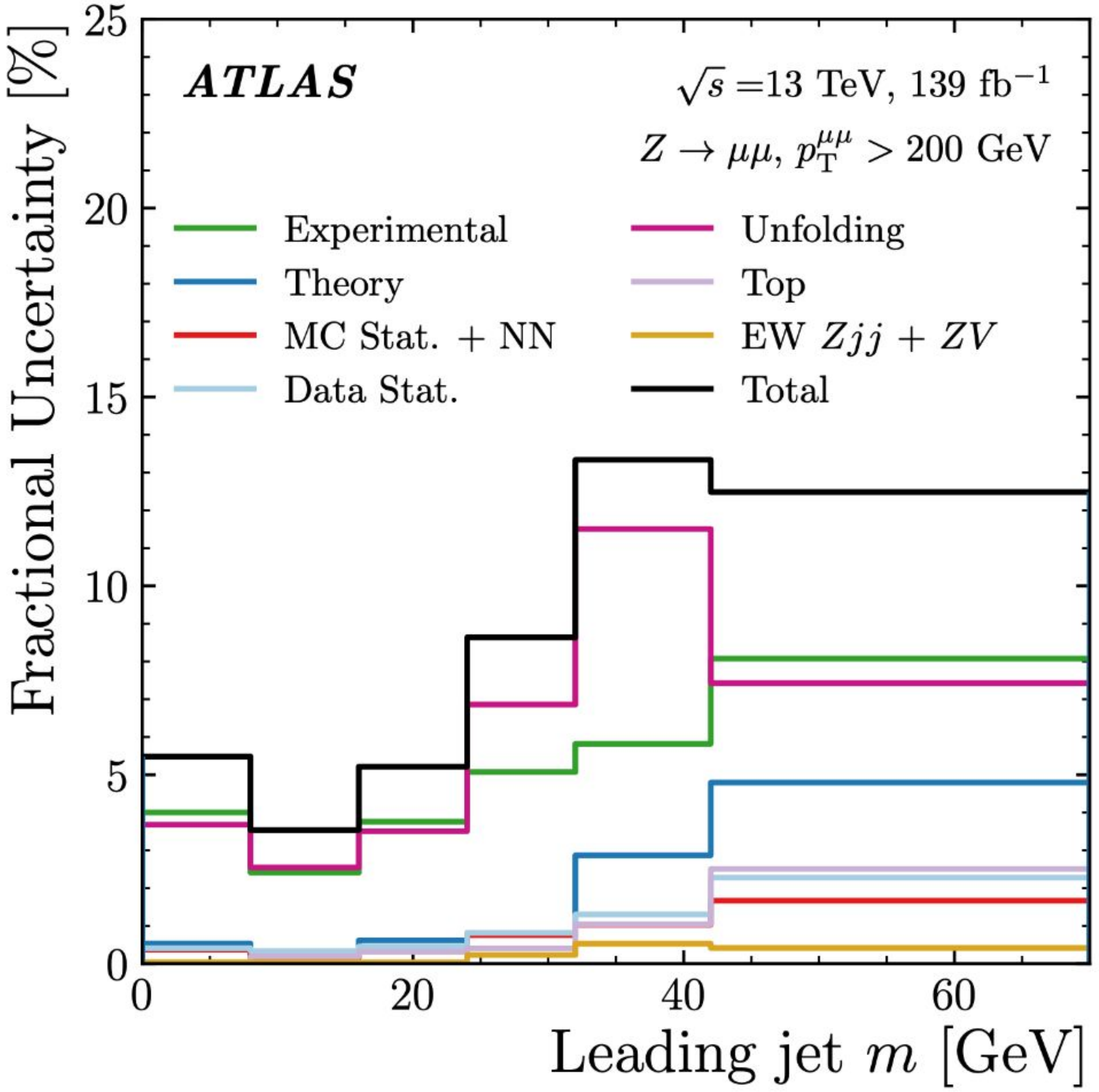
Concrete example

Plotting one variable

The measurement is a 24-dimensional object.
Let's check out two examples of measured differential cross-sections:

Leading jet mass





Note: these plots are made interactively in a Jupyter Notebook

But we can go beyond just measuring the 24 input variables...
we can also imagine **brand new observables** that we want to measure,
and even probe **different bins or regions of phase space**.

Let's construct some new observables...

$$\tau_{21} = \tau_2/\tau_1$$

This ratio of two parameters measuring jet substructure is useful for e.g. W vs. QCD jet classification:

$$\tau_N = \frac{1}{d_0} \sum_k p_{T,k} \min_A \{\Delta R_{A,k}\}$$

↑ Sum over constituents

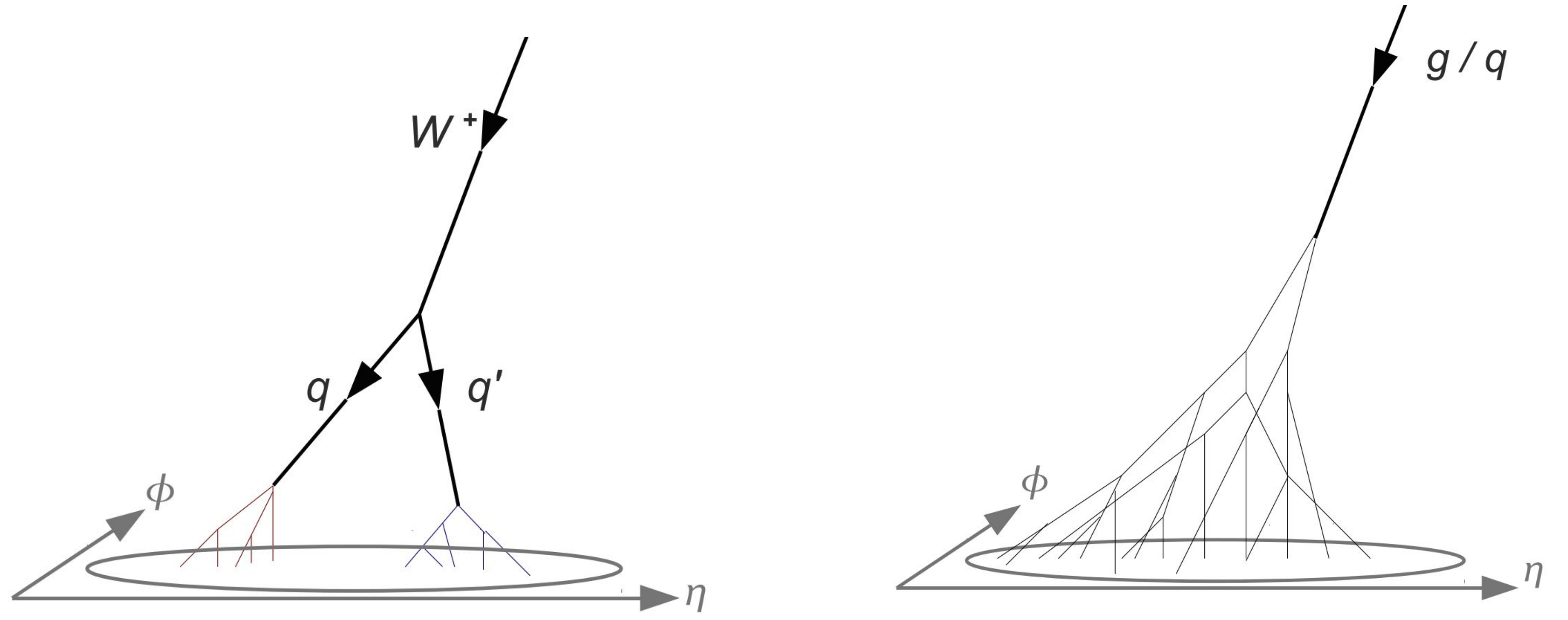
↑ Minimize distance to candidate subjet axes

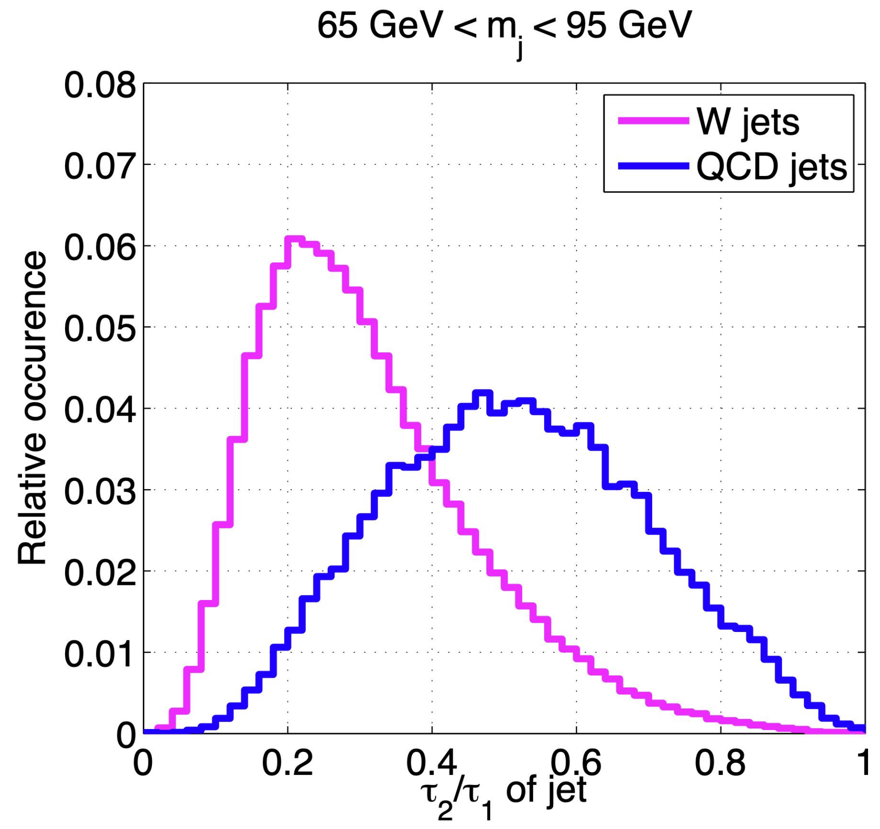
Jet shape that "counts" number of subjets!

subjets: ≤ N   > N

τN: 0   1

Adapted from "N-jettiness" [Stewart, Tackmann, Waalewijn: 1004.2489]



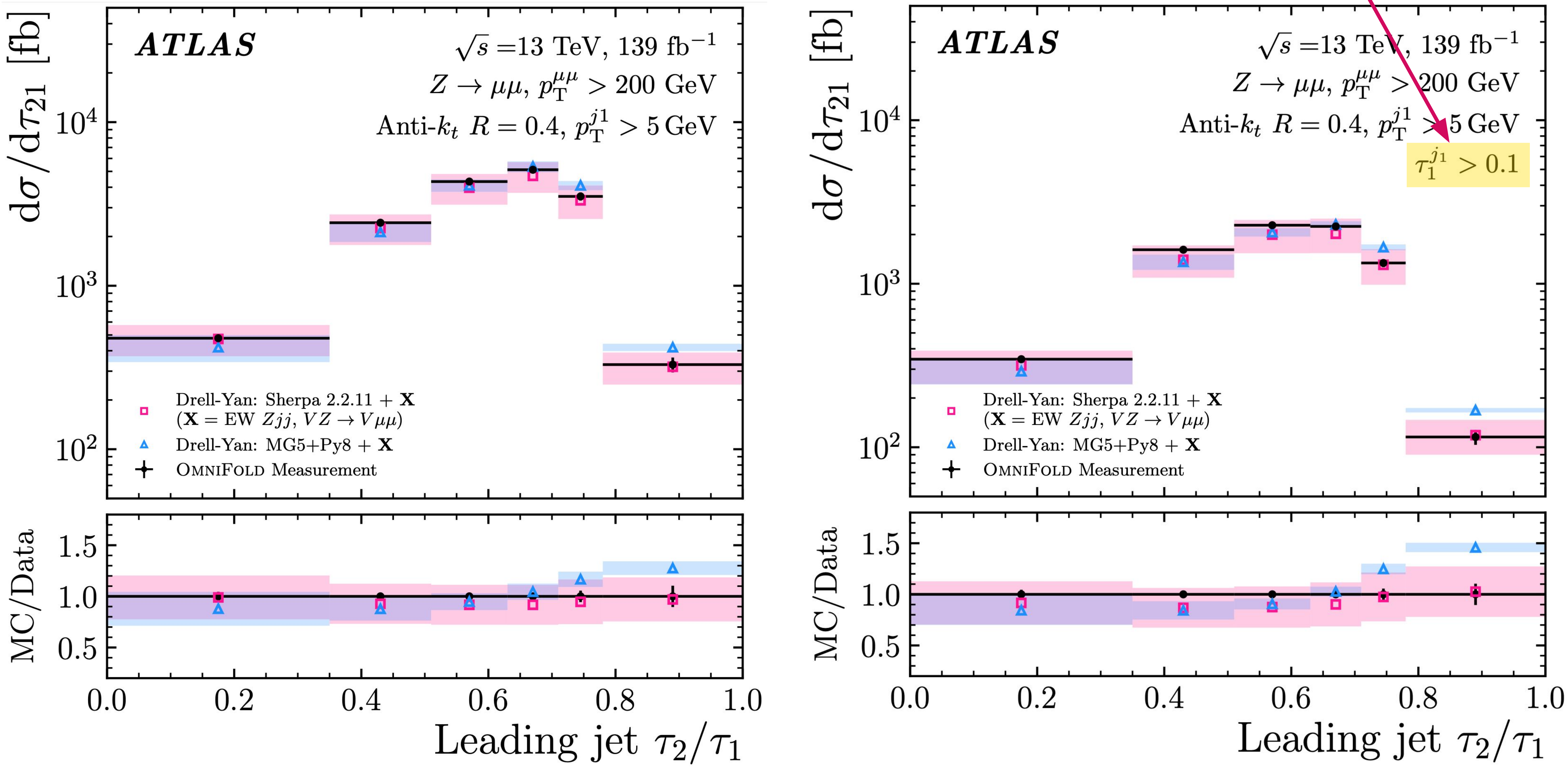


Images from J. Thaler's talk and paper from 2011

Let's construct some new observables...

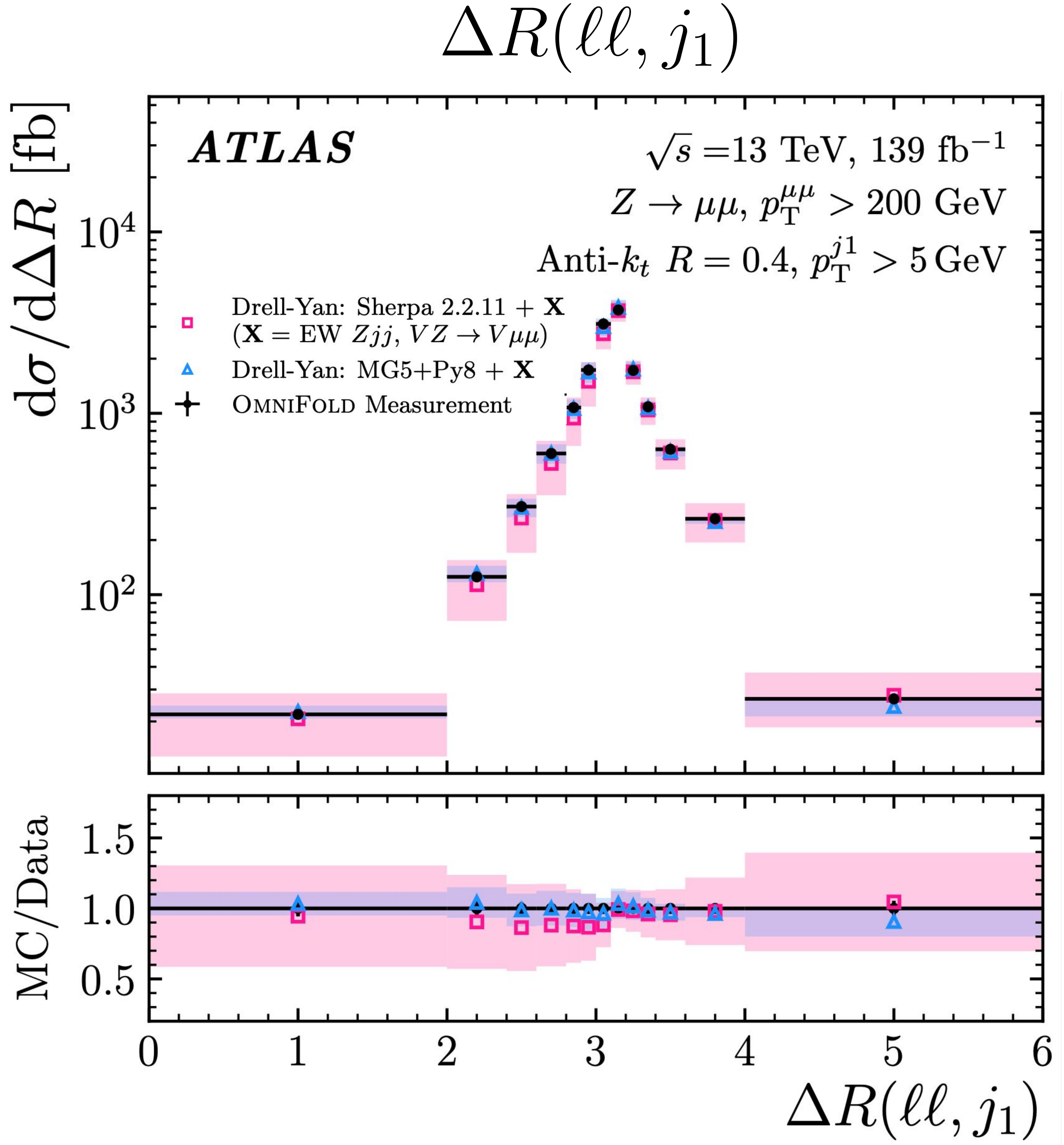
$$\tau_{21} = \tau_2/\tau_1$$

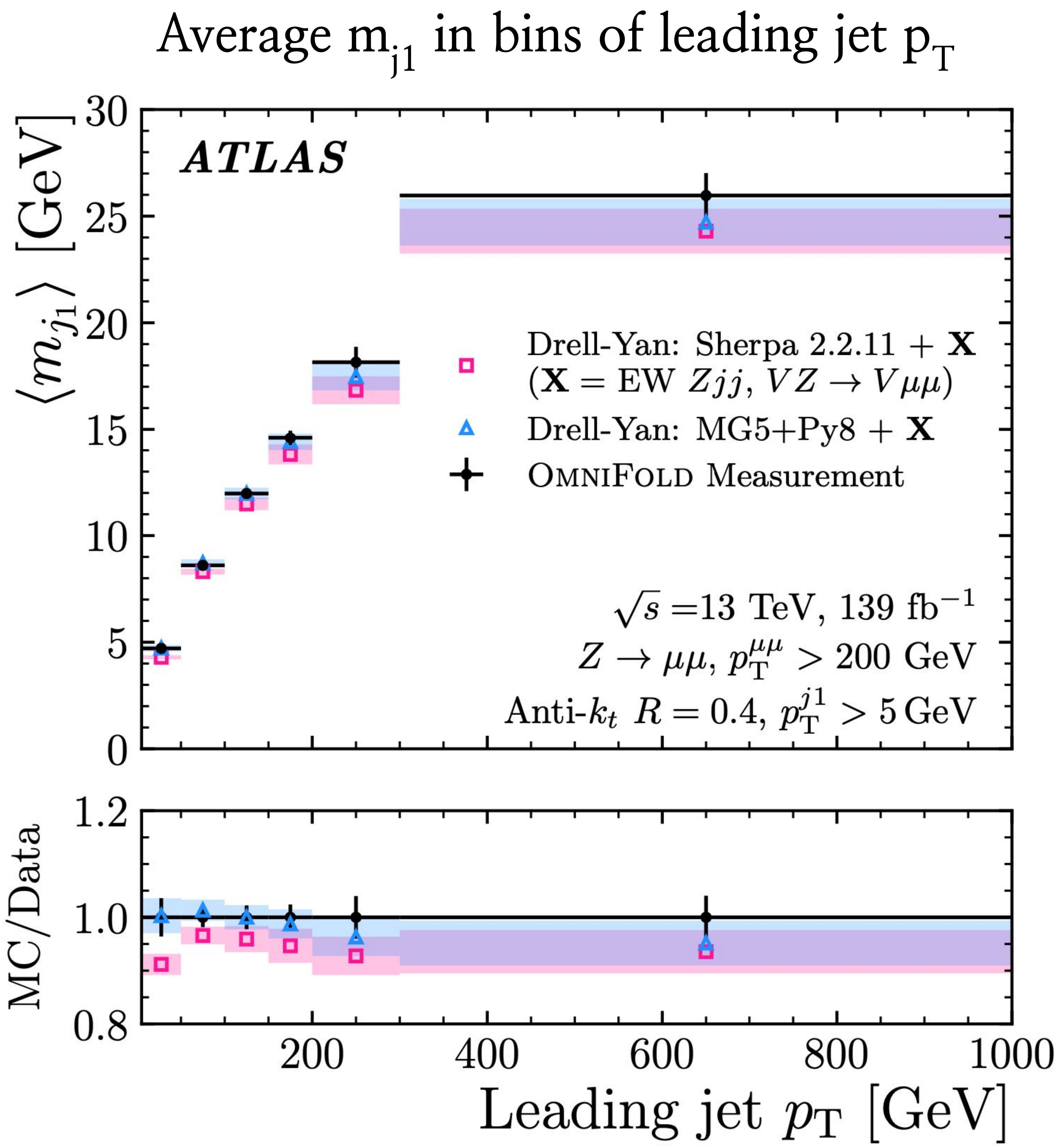
IRC-safe phase space!



Mariel Pettee

Let’s construct some new observables…





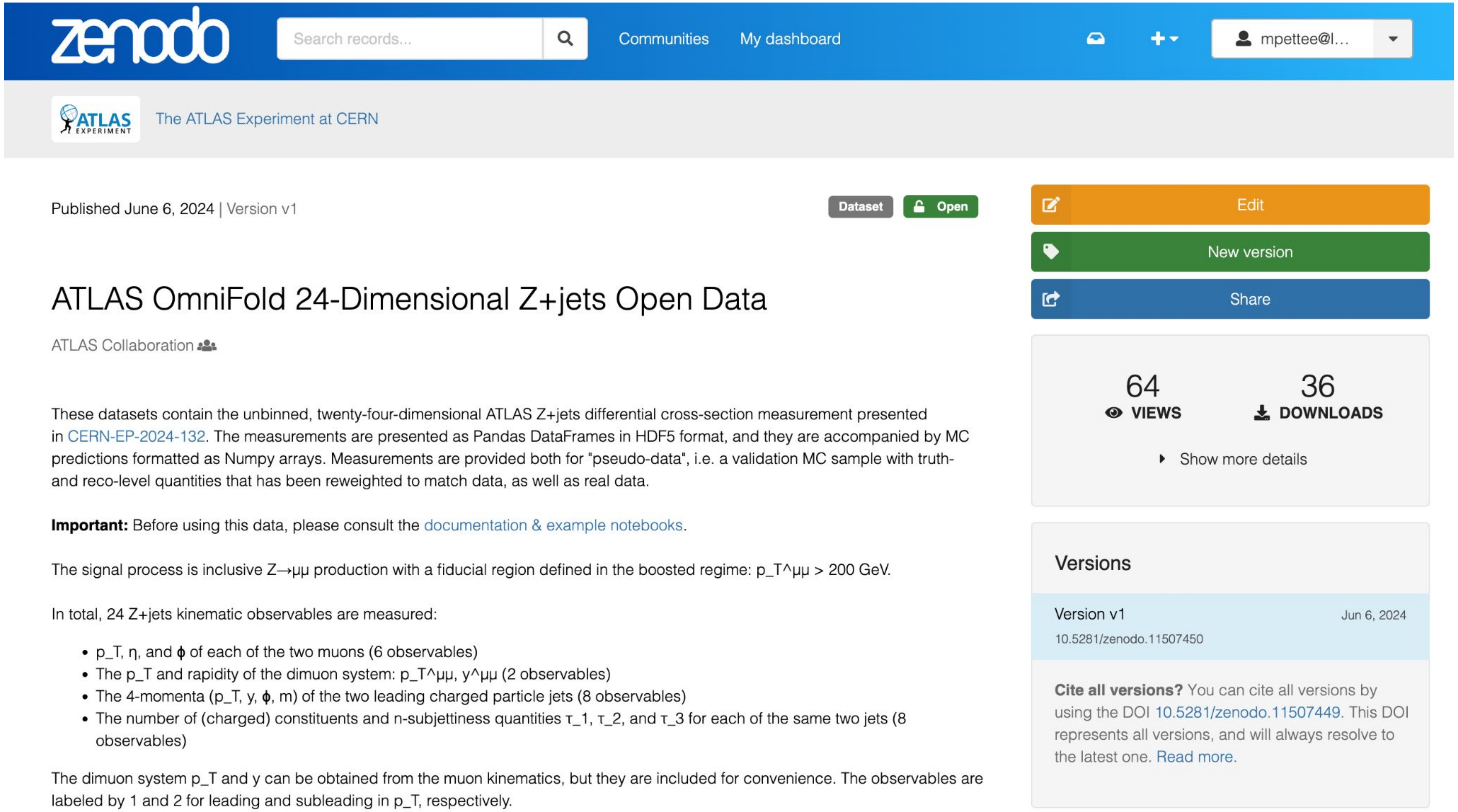
Note: neither of these were ‘directly measured’
Created ‘on-the-fly’ using the public measurement (i.e. the public event dataset)

Mariel Pettee

https://zenodo.org/records/11507450

Public dataset

Our measurements are published here on Zenodo: https://zenodo.org/records/11507450



Published June 6, 2024 | Version v1

ATLAS OmniFold 24-Dimensional Z+jets Open Data

ATLAS Collaboration

These datasets contain the unbinned, twenty-four-dimensional ATLAS Z+jets differential cross-section measurement presented in CERN-EP-2024-132. The measurements are presented as Pandas DataFrames in HDF5 format, and they are accompanied by MC predictions formatted as Numpy arrays. Measurements are provided both for "pseudo-data", i.e. a validation MC sample with truth- and reco-level quantities that has been reweighted to match data, as well as real data.

Important: Before using this data, please consult the documentation & example notebooks.

The signal process is inclusive Z→μμ production with a fiducial region defined in the boosted regime: p_T^μμ > 200 GeV.

In total, 24 Z+jets kinematic observables are measured:

- p_T, η, and ϕ of each of the two muons (6 observables)
- The p_T and rapidity of the dimuon system: p_T^μμ, y^μμ (2 observables)
- The 4-momenta (p_T, y, ϕ, m) of the two leading charged particle jets (8 observables)
- The number of (charged) constituents and n-subjettiness quantities τ_1, τ_2, and τ_3 for each of the same two jets (8 observables)

The dimuon system p_T and y can be obtained from the muon kinematics, but they are included for convenience. The observables are labeled by 1 and 2 for leading and subleading in p_T, respectively.

Versions

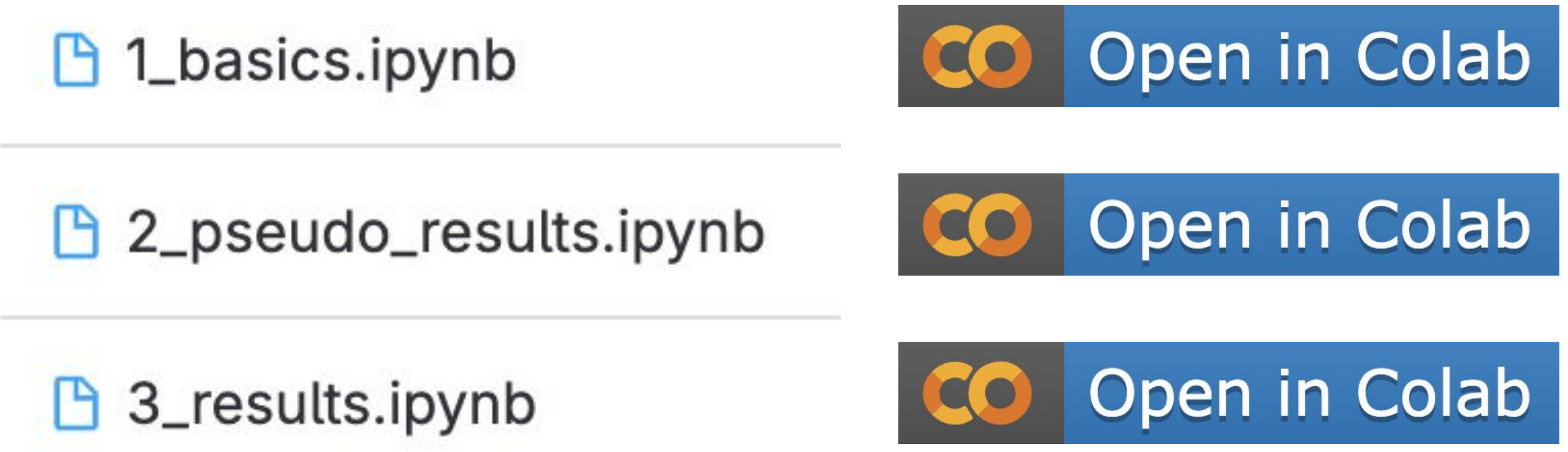
Version v1 Jun 6, 2024
10.5281/zenodo.11507450

Cite all versions? You can cite all versions by using the DOI 10.5281/zenodo.11507449. This DOI represents all versions, and will always resolve to the latest one. Read more.

User guide and example analysis code

We have also published a detailed README & several Jupyter notebooks with instructions about how to use the public datasets:

https://gitlab.cern.ch/atlas-physics/public/sm-z-jets-omnifold-2024

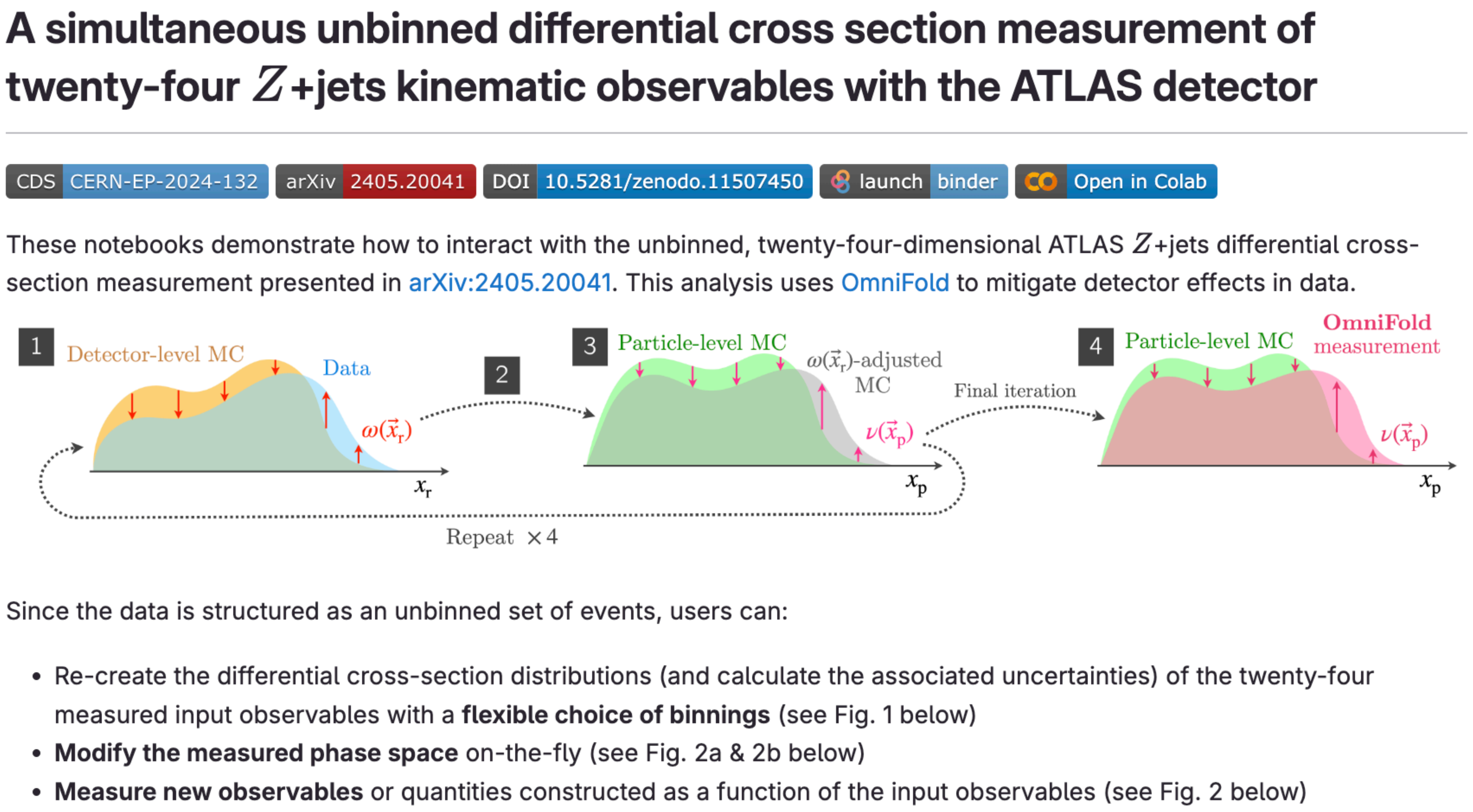


1_basics.ipynb

2_pseudo_results.ipynb

3_results.ipynb

User guide and example analysis code



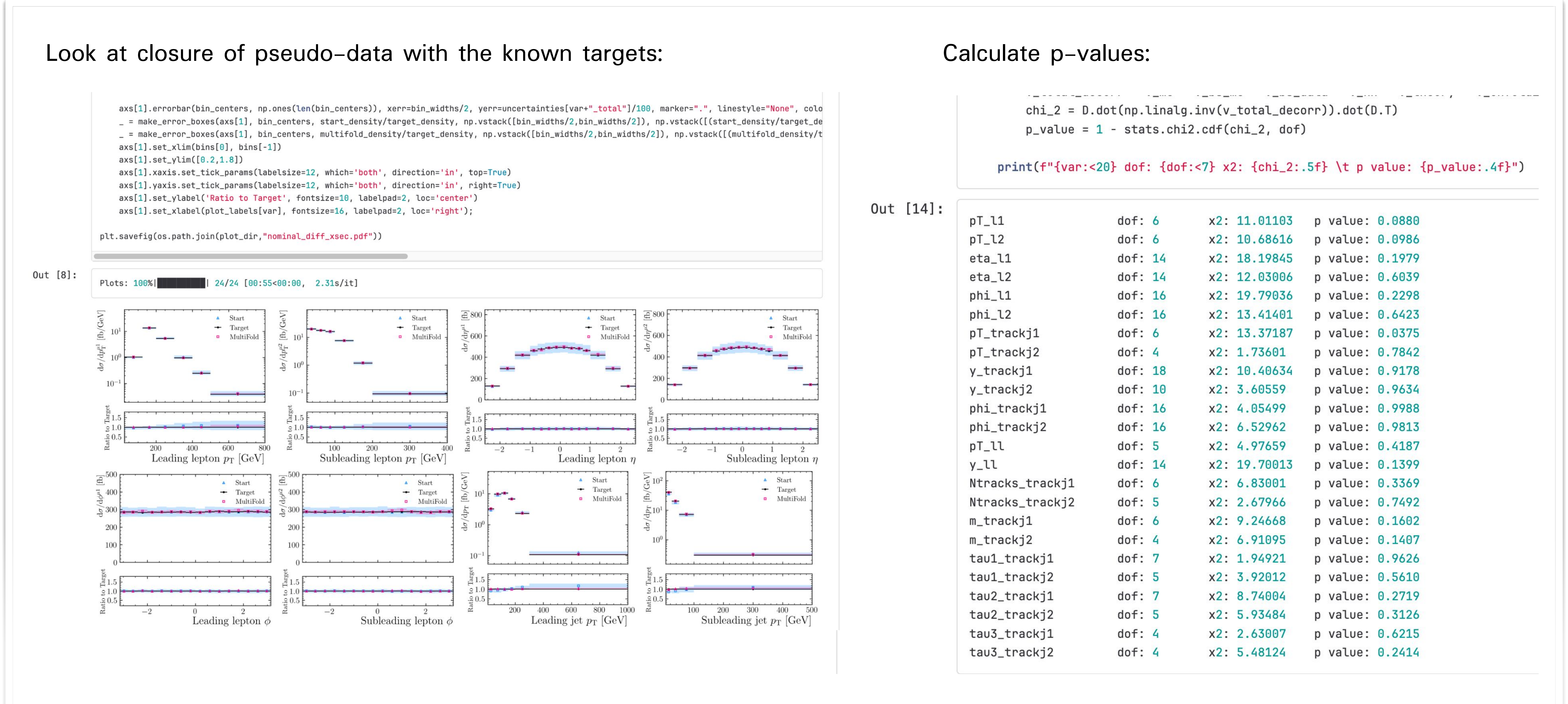
A simultaneous unbinned differential cross section measurement of twenty-four Z+jets kinematic observables with the ATLAS detector

These notebooks demonstrate how to interact with the unbinned, twenty-four-dimensional ATLAS Z+jets differential cross-section measurement presented in arXiv:2405.20041. This analysis uses OmniFold to mitigate detector effects in data.

Since the data is structured as an unbinned set of events, users can:

- Re-create the differential cross-section distributions (and calculate the associated uncertainties) of the twenty-four measured input observables with a **flexible choice of binnings** (see Fig. 1 below)
- **Modify the measured phase space** on-the-fly (see Fig. 2a & 2b below)
- **Measure new observables** or quantities constructed as a function of the input observables (see Fig. 2 below)

User guide and example analysis code



Look at closure of pseudo-data with the known targets:

```
        axs[1].errorbar(bin_centers, np.ones(len(bin_centers)), xerr=bin_widths/2, yerr=uncertainties[var+"_total"]/100, marker=".", linestyle="None", colo
        _ = make_error_boxes(axs[1], bin_centers, start_density/target_density, np.vstack([bin_widths/2,bin_widths/2]), np.vstack([(start_density/target_de
        _ = make_error_boxes(axs[1], bin_centers, multifold_density/target_density, np.vstack([bin_widths/2,bin_widths/2]), np.vstack([(multifold_density/t
        axs[1].set_xlim(bins[0], bins[-1])
        axs[1].set_ylim([0.2,1.8])
        axs[1].xaxis.set_tick_params(labelsize=12, which='both', direction='in', top=True)
        axs[1].yaxis.set_tick_params(labelsize=12, which='both', direction='in', right=True)
        axs[1].set_ylabel('Ratio to Target', fontsize=10, labelpad=2, loc='center')
        axs[1].set_xlabel(plot_labels[var], fontsize=16, labelpad=2, loc='right');

    plt.savefig(os.path.join(plot_dir,"nominal_diff_xsec.pdf"))
```

Out [8]:

```
Plots: 100%|██████████| 24/24 [00:55<00:00,  2.31s/it]
```

Calculate p-values:

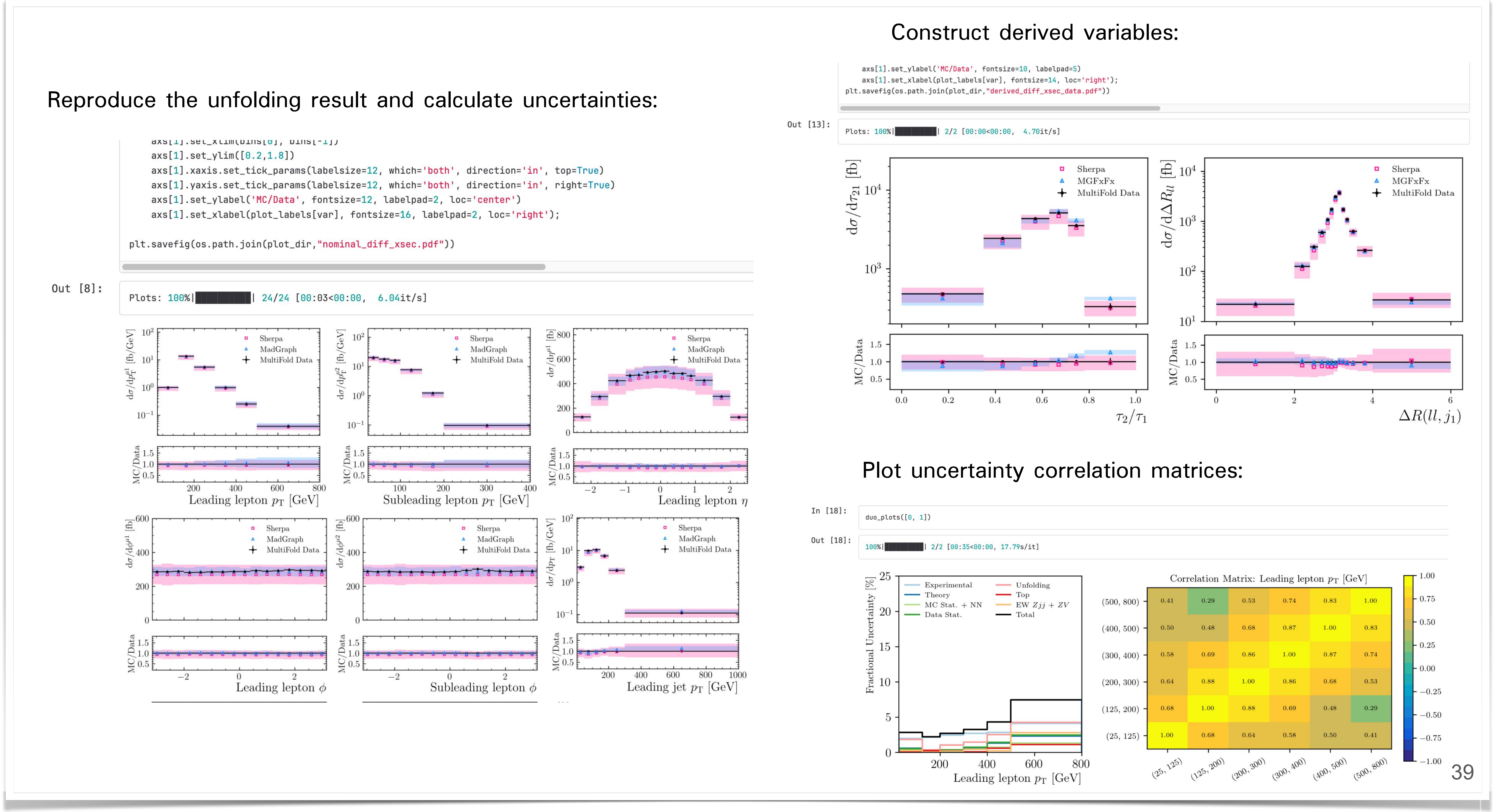
```
        chi_2 = D.dot(np.linalg.inv(v_total_decorr)).dot(D.T)
        p_value = 1 - stats.chi2.cdf(chi_2, dof)

    print(f"{var:<20} dof: {dof:<7} x2: {chi_2:.5f} \t p value: {p_value:.4f}")
```

Out [14]:

```
pT_l1                dof: 6       x2: 11.01103    p value: 0.0880
pT_l2                dof: 6       x2: 10.68616    p value: 0.0986
eta_l1               dof: 14      x2: 18.19845    p value: 0.1979
eta_l2               dof: 14      x2: 12.03006    p value: 0.6039
phi_l1               dof: 16      x2: 19.79036    p value: 0.2298
phi_l2               dof: 16      x2: 13.41401    p value: 0.6423
pT_trackj1           dof: 6       x2: 13.37187    p value: 0.0375
pT_trackj2           dof: 4       x2: 1.73601     p value: 0.7842
y_trackj1            dof: 18      x2: 10.40634    p value: 0.9178
y_trackj2            dof: 10      x2: 3.60559     p value: 0.9634
phi_trackj1          dof: 16      x2: 4.05499     p value: 0.9988
phi_trackj2          dof: 16      x2: 6.52962     p value: 0.9813
pT_ll                dof: 5       x2: 4.97659     p value: 0.4187
y_ll                 dof: 14      x2: 19.70013    p value: 0.1399
Ntracks_trackj1      dof: 6       x2: 6.83001     p value: 0.3369
Ntracks_trackj2      dof: 5       x2: 2.67966     p value: 0.7492
m_trackj1            dof: 6       x2: 9.24668     p value: 0.1602
m_trackj2            dof: 4       x2: 6.91095     p value: 0.1407
tau1_trackj1         dof: 7       x2: 1.94921     p value: 0.9626
tau1_trackj2         dof: 5       x2: 3.92012     p value: 0.5610
tau2_trackj1         dof: 7       x2: 8.74004     p value: 0.2719
tau2_trackj2         dof: 5       x2: 5.93484     p value: 0.3126
tau3_trackj1         dof: 4       x2: 2.63007     p value: 0.6215
tau3_trackj2         dof: 4       x2: 5.48124     p value: 0.2414
```

[arXiv:2405.20041](https://arxiv.org/abs/2405.20041)

User guide and example analysis code



Reproduce the unfolding result and calculate uncertainties:

```
    axs[1].set_xlim(bins[0], bins[-1])
    axs[1].set_ylim([0.2,1.8])
    axs[1].xaxis.set_tick_params(labelsize=12, which='both', direction='in', top=True)
    axs[1].yaxis.set_tick_params(labelsize=12, which='both', direction='in', right=True)
    axs[1].set_ylabel('MC/Data', fontsize=12, labelpad=2, loc='center')
    axs[1].set_xlabel(plot_labels[var], fontsize=16, labelpad=2, loc='right');

plt.savefig(os.path.join(plot_dir,"nominal_diff_xsec.pdf"))
```

Out [8]:

```
Plots: 100%|██████████| 24/24 [00:03<00:00,  6.04it/s]
```

Construct derived variables:

```
    axs[1].set_ylabel('MC/Data', fontsize=10, labelpad=5)
    axs[1].set_xlabel(plot_labels[var], fontsize=14, loc='right');
plt.savefig(os.path.join(plot_dir,"derived_diff_xsec_data.pdf"))
```

Out [13]:

```
Plots: 100%|██████████| 2/2 [00:00<00:00,  4.70it/s]
```

Plot uncertainty correlation matrices:

In [18]:

```
duo_plots([0, 1])
```

Out [18]:

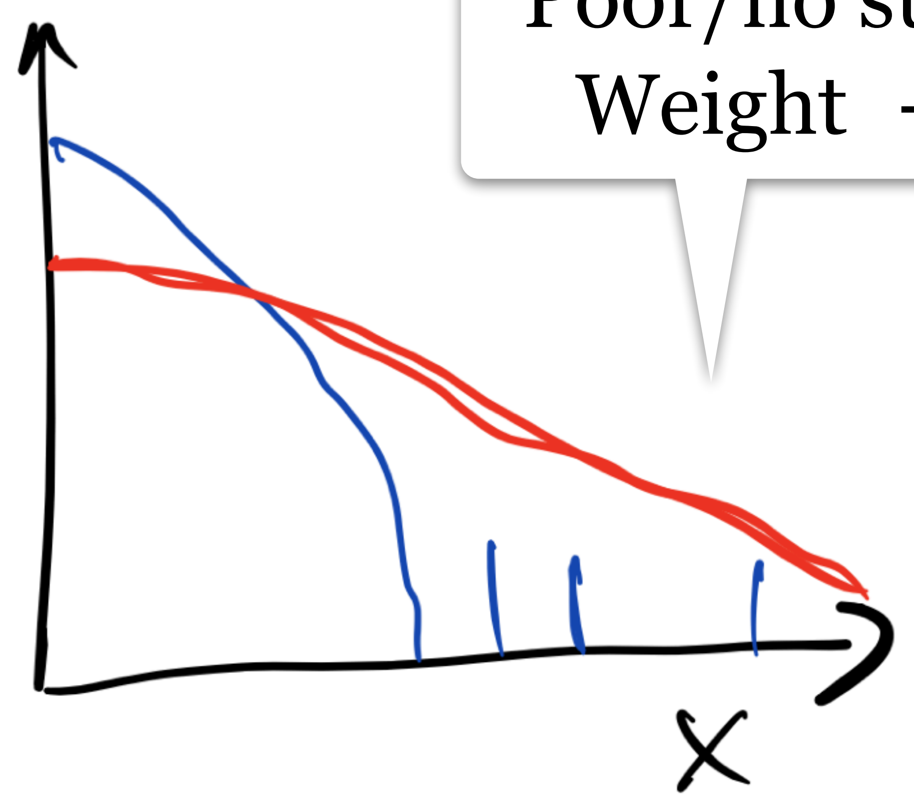
```
100%|██████████| 2/2 [00:35<00:00, 17.79s/it]
```

Correlation Matrix: Leading lepton p_T [GeV]

	(25, 125)	(125, 200)	(200, 300)	(300, 400)	(400, 500)	(500, 800)
(500, 800)	0.41	0.29	0.53	0.74	0.83	1.00
(400, 500)	0.50	0.48	0.68	0.87	1.00	0.83
(300, 400)	0.58	0.69	0.86	1.00	0.87	0.74
(200, 300)	0.64	0.88	1.00	0.86	0.68	0.53
(125, 200)	0.68	1.00	0.88	0.69	0.48	0.29
(25, 125)	1.00	0.68	0.64	0.58	0.50	0.41

Shortcomings and challenges

- A bunch of challenges were faced and overcome
  - Network gets confused by discontinuity in ϕ. It assumes smooth functions. Solved by letting network used sin(ϕ) and cos(ϕ).
- Insufficient support across full phase space
  - If we have regions of phase space with too few initial MC events, need 'infinite weight'
    → easily spotted, check for large weights.
    Solved by using MC with decent prediction of data
- Instabilities of the network
  - Networks (Keras Tensorflow) initialized with random seed. Quickly finds solution. But different dep. on seed
    → per-event instabilities
  - Hyperparameter optimization, and ensembling (add computing power …)
  - 'New type of uncertainty'



Poor/no support
Weight → ∞

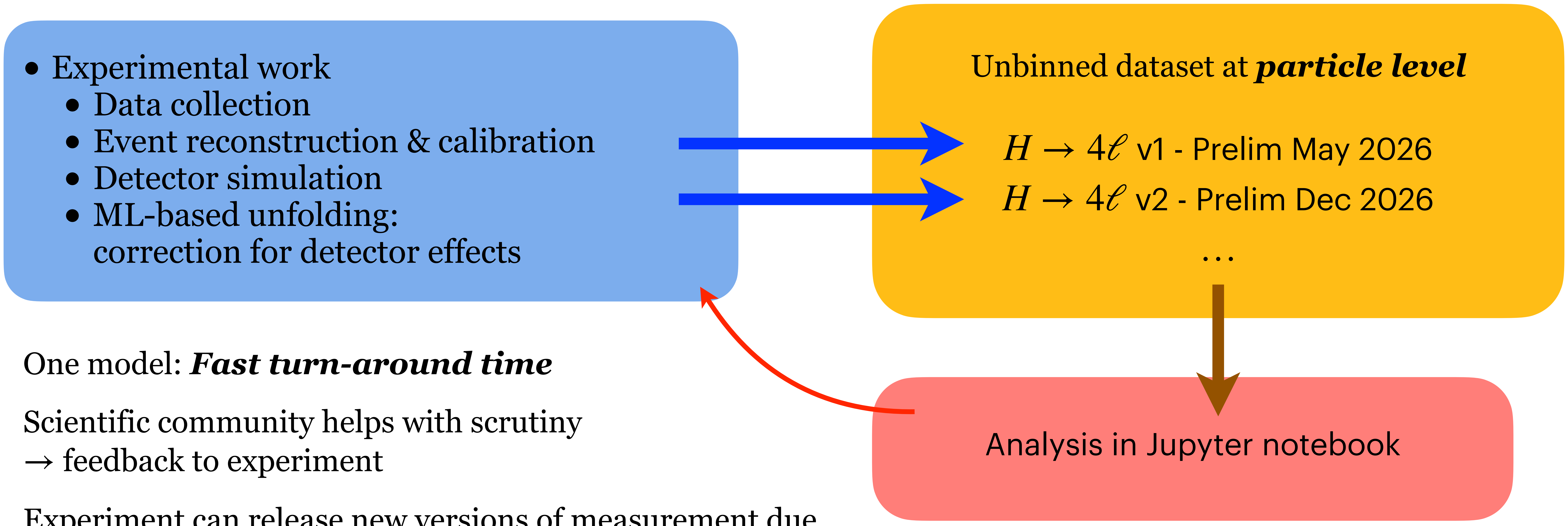
Shortcomings and challenges

- Background subtraction:
  - Thus far measurements only done for which backgrounds are small
  - ATLAS: Z+jet ($t\bar{t}$ bkg <1%)
  - CMS: inelastic QCD (no background)
- Several approaches to subtract backgrounds suggested, but not tested for a real analysis when backgrounds are signficant
  - Fun problem to try to solve!
- Example: $H \rightarrow \gamma\gamma$, big backgrounds, but can get decent precision
- Low statistics measurements, e.g. $H \rightarrow 4\ell$
  - Should work ‘out-of-the-box’ but data statistical uncertainties will be significant, and might need special treatment (Poisson uncertainties with low mean)
- Complex final states, with poor resolution objects have not been validated yet
  - Neutral hadrons

Next steps for 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. $p_T^{\ell 1}, p_T^{\ell 2}, \eta^{\ell 2}, \eta^{\ell 1}, p_T^{j1},\ p_T^{j2}$
    - Note 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) and type of all particles in an event (One event might have 50 particles, another 1200)
    - Can then e.g. build jets with different jet algorithms as specified by user

Precision measurements: Future approach



Experiment (ATLAS, CMS, …)

- Experimental work
  - Data collection
  - Event reconstruction & calibration
  - Detector simulation
  - ML-based unfolding: correction for detector effects

Public results ('open data')

Unbinned dataset at ***particle level***

$H \to 4\ell$ v1 - Prelim May 2026

$H \to 4\ell$ v2 - Prelim Dec 2026

…

Analysis in Jupyter notebook

One model: ***Fast turn-around time***

Scientific community helps with scrutiny
→ feedback to experiment

Experiment can release new versions of measurement due to either issue found or perhaps new improved calibration

Equity and education

- Currently, use of data and measurments quite restricted to ‘the privileged few’
  - “Grad students at a lab”
- With public data that you can access in interactive Python notebooks, use of data much more accessible
  - The new ATLAS measurements can be opened in Google Colab
    → A high school student with a google account and a web browser can access it
    → Make cool plots — corresponding to a true measurement (Naure) — within minutes
    → Interactively change the code, try existing examples
  - Super accessible for anyone, also underprivileged learners
- Clear use cases in education

Summary

- Rapid development in machine learning opens up for new possibilities in particle physics
- One such development highlighted here: simultaneous unfolding of many variables at once
- Opens up for many future applications
  - Significant more information provided
  - Clear applications to e.g. MC tuning, searches for BSM effects, anomaly detection
- Provides natural way to make data and physics interception open
  - Increased collaboration within the community (theory+experimentalists)
  - Accessible to physics education and much more inclusive to interested people
- Challenges and details around validation and guidelines still being worked out
  - Significant interest+involvemnt from precision measurment community
- Exciting times ahead!

In the future, we might all do our end-analysis using Jupyter notebooks …

Backup

How can I use OmniFold in my own analysis?

- We are working to implement OmniFold in RooUnfold.
  - For most analyses with $O(1)$ observables, this is probably the best way to proceed.
- We have also released a pip-installable version that scales well to many observables:
  - `pip install omnifold`

We are continually improving these methods & tools – feedback and collaboration is most welcome!

Classification

Classifier
$f(\vec{x})$

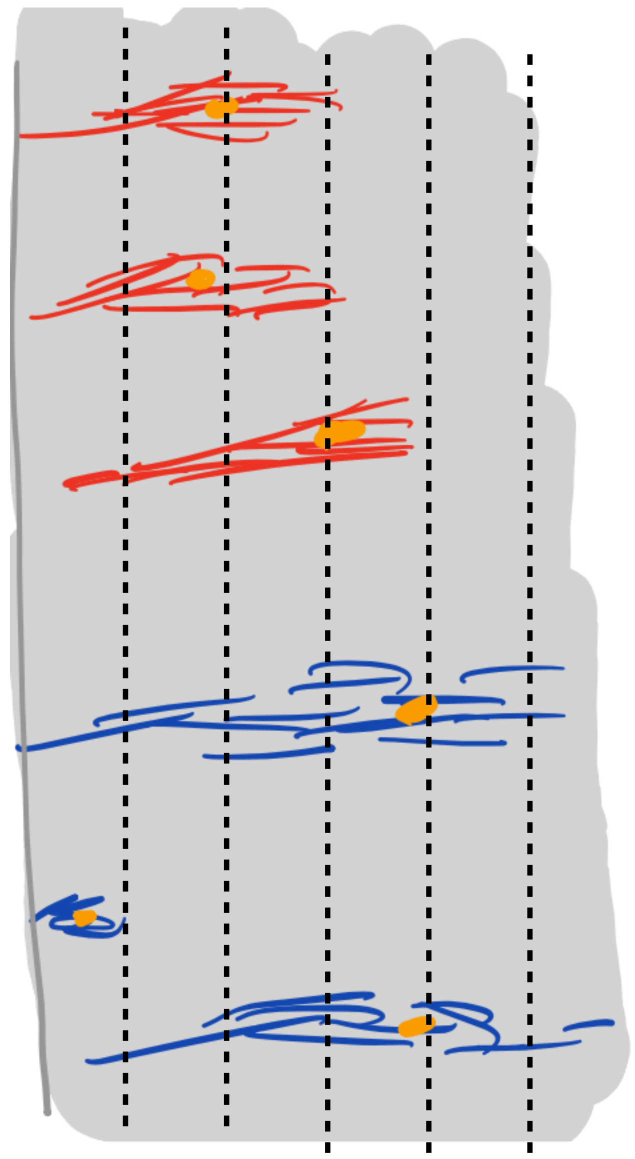
- Most common application of machine learning in particle physics is classification
- Goal: discriminate ‘signal’ from ‘background’
  - Example: Detector signals from real electrons vs hadrons/photons

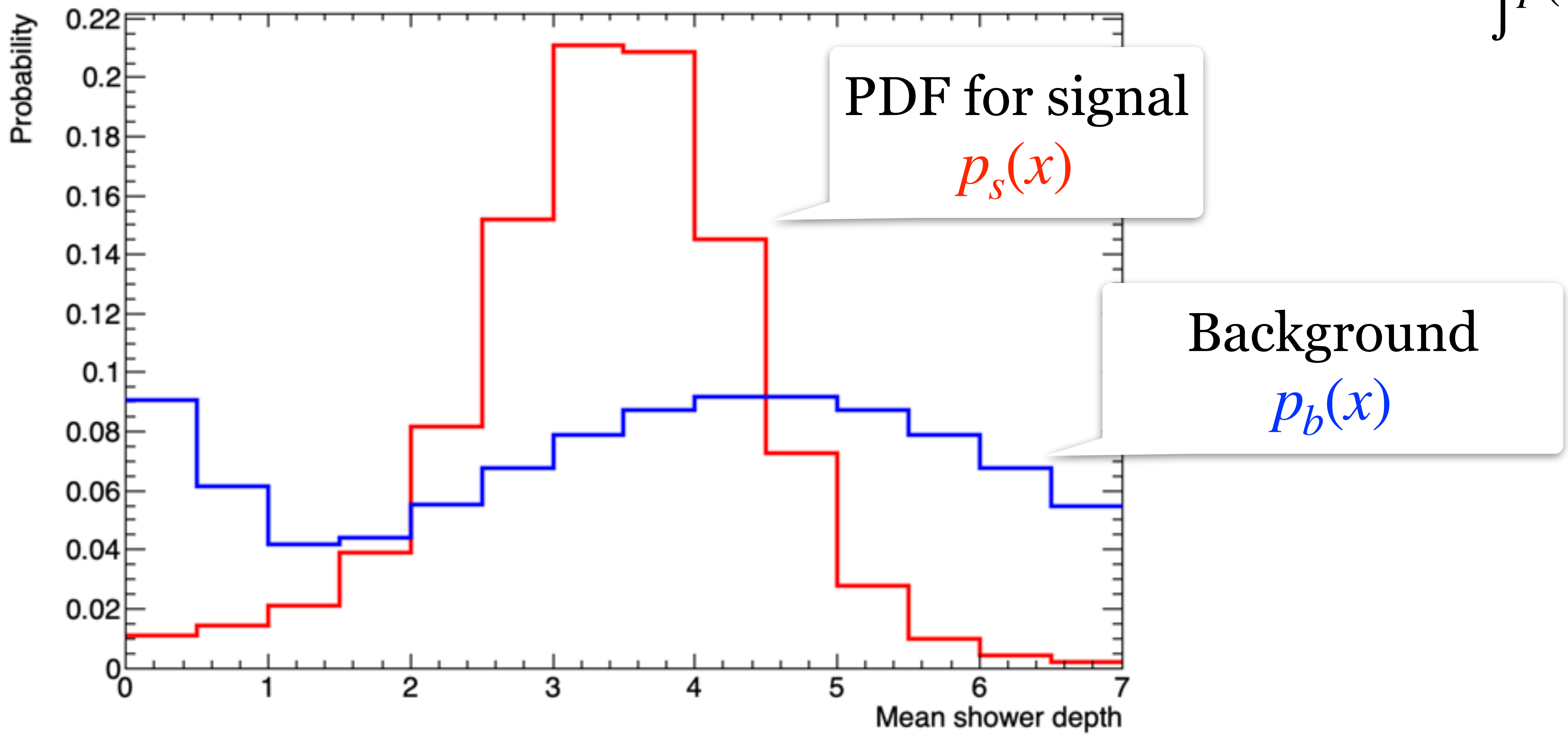
PDF:
$\int p(x)\,dx = 1$

3 real electrons (signal)

3 fake electrons (background)

From hadrons, photons or noise bursts …





PDF for signal
$p_s(x)$

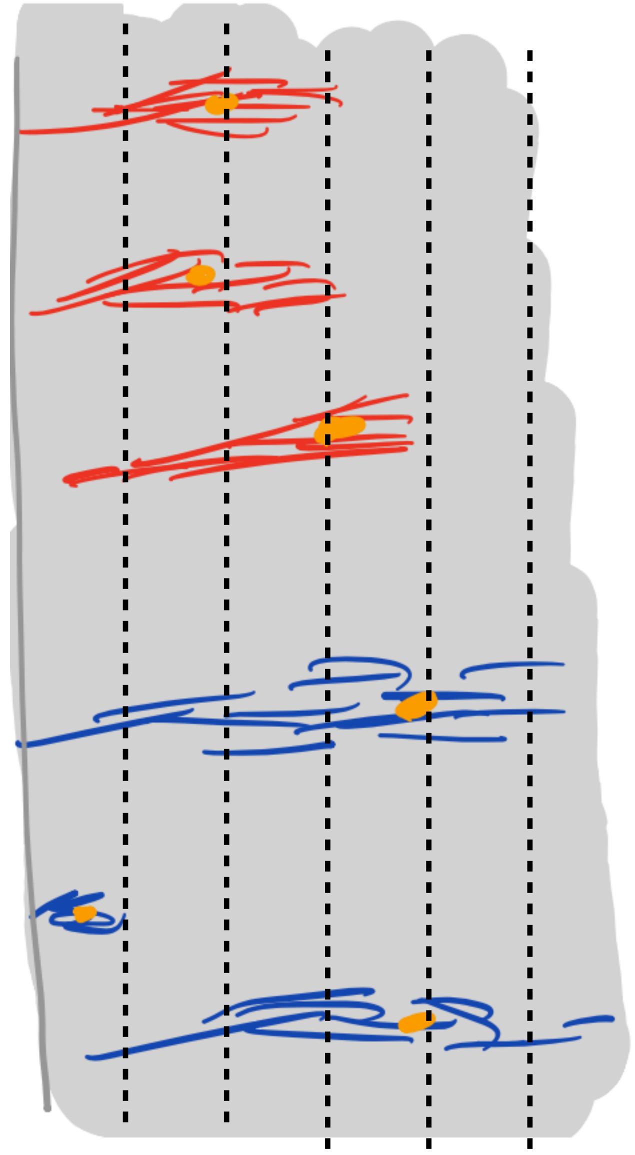
Background
$p_b(x)$

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

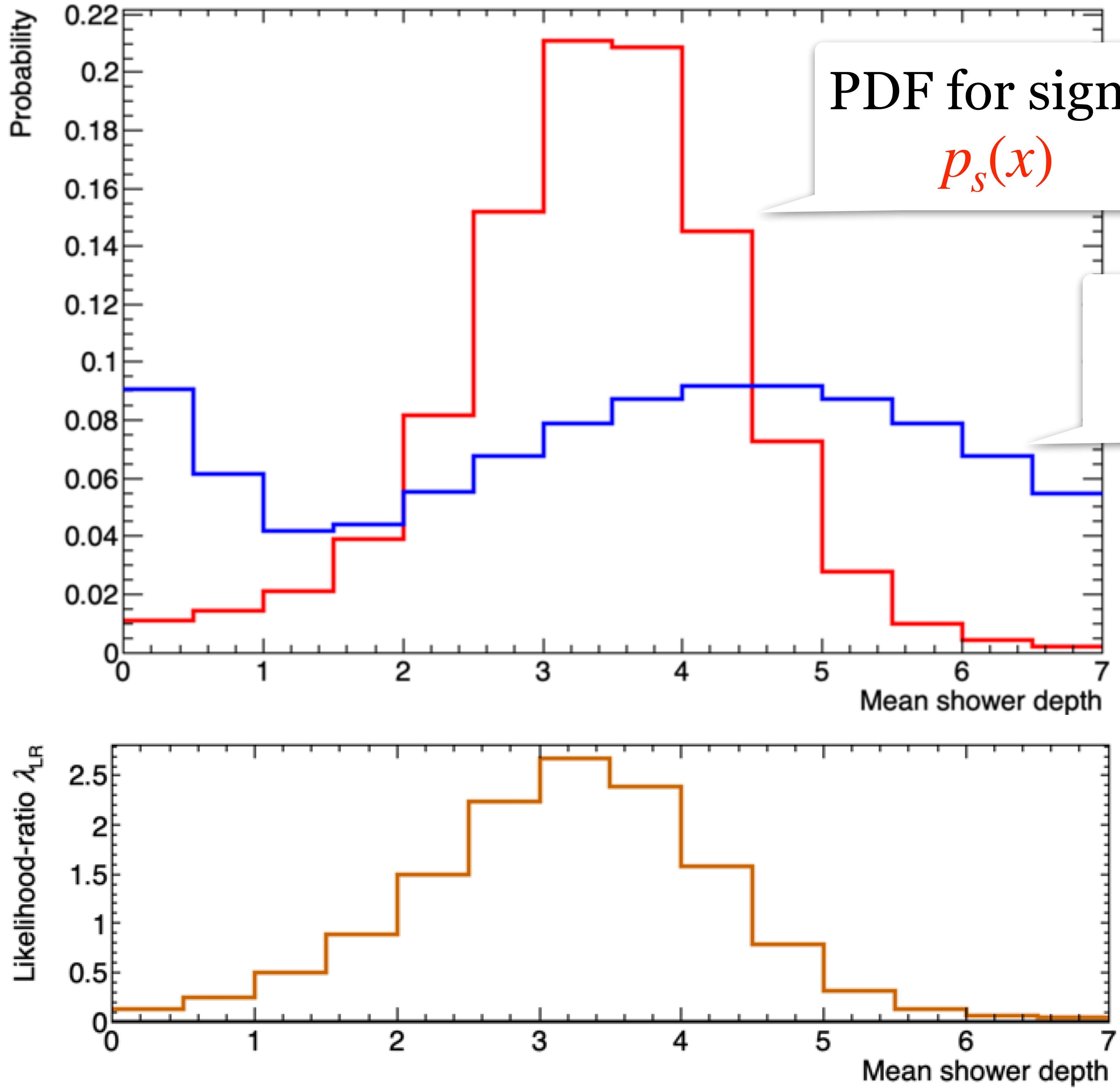
PDF:
$$\int p(x)\,dx = 1$$



3 real electrons (signal)

3 fake electrons (background)

From hadrons, photons or noise bursts …



PDF for signal
$p_s(x)$

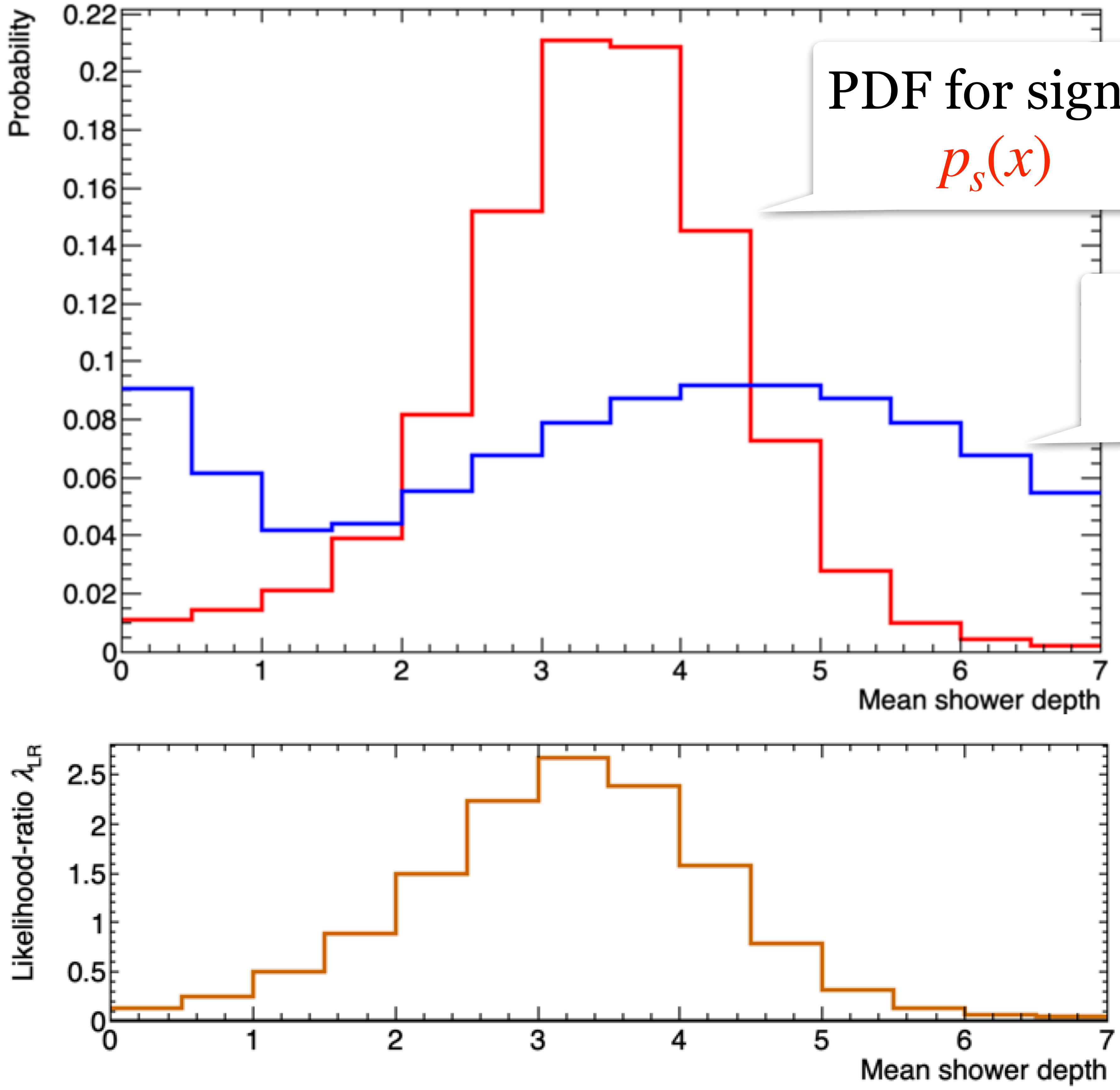
Background
$p_b(x)$

Likelihood ratio:

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

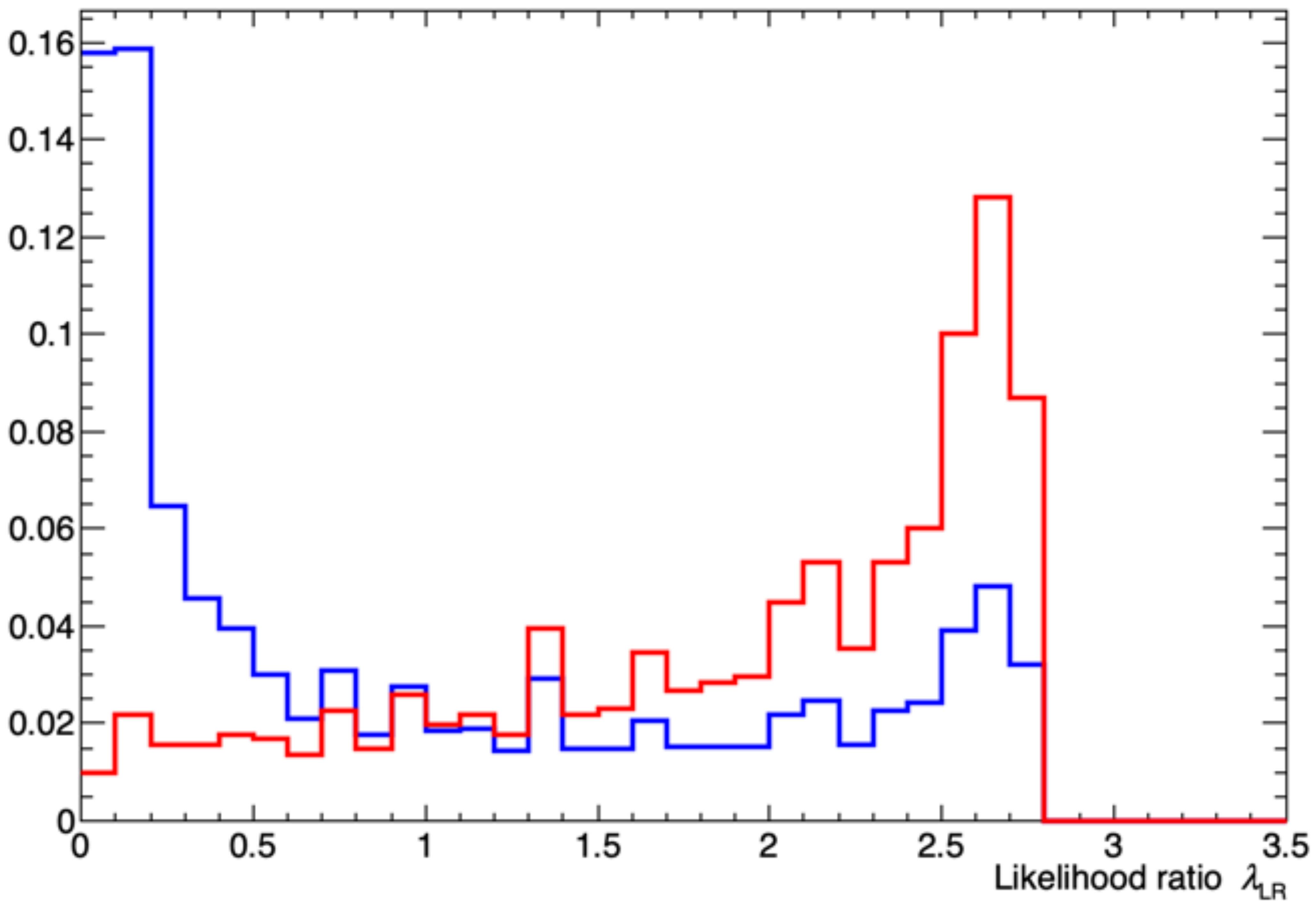
The Neyman-Pearson lemma

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



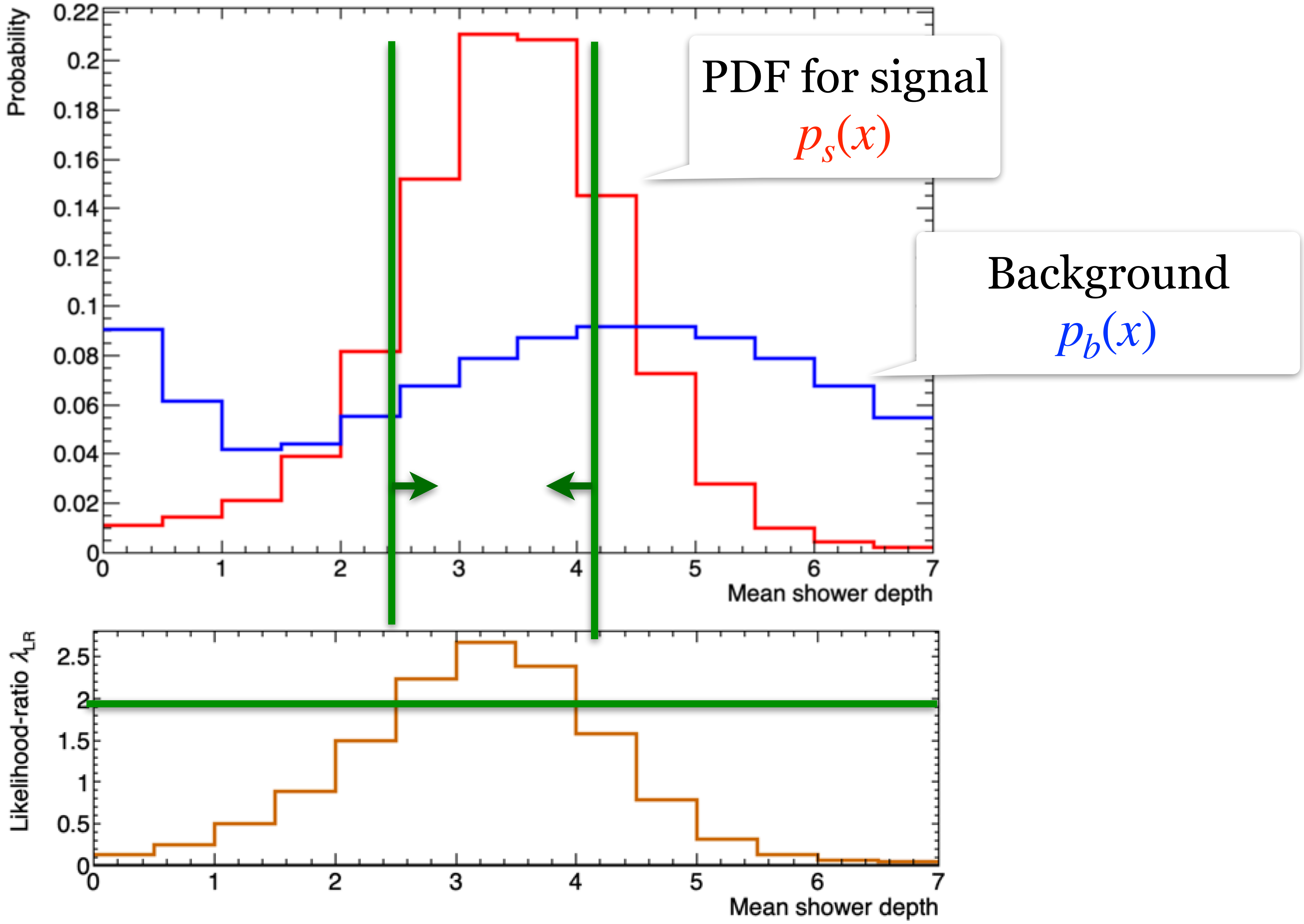
PDF for signal
$p_s(x)$

Background
$p_b(x)$



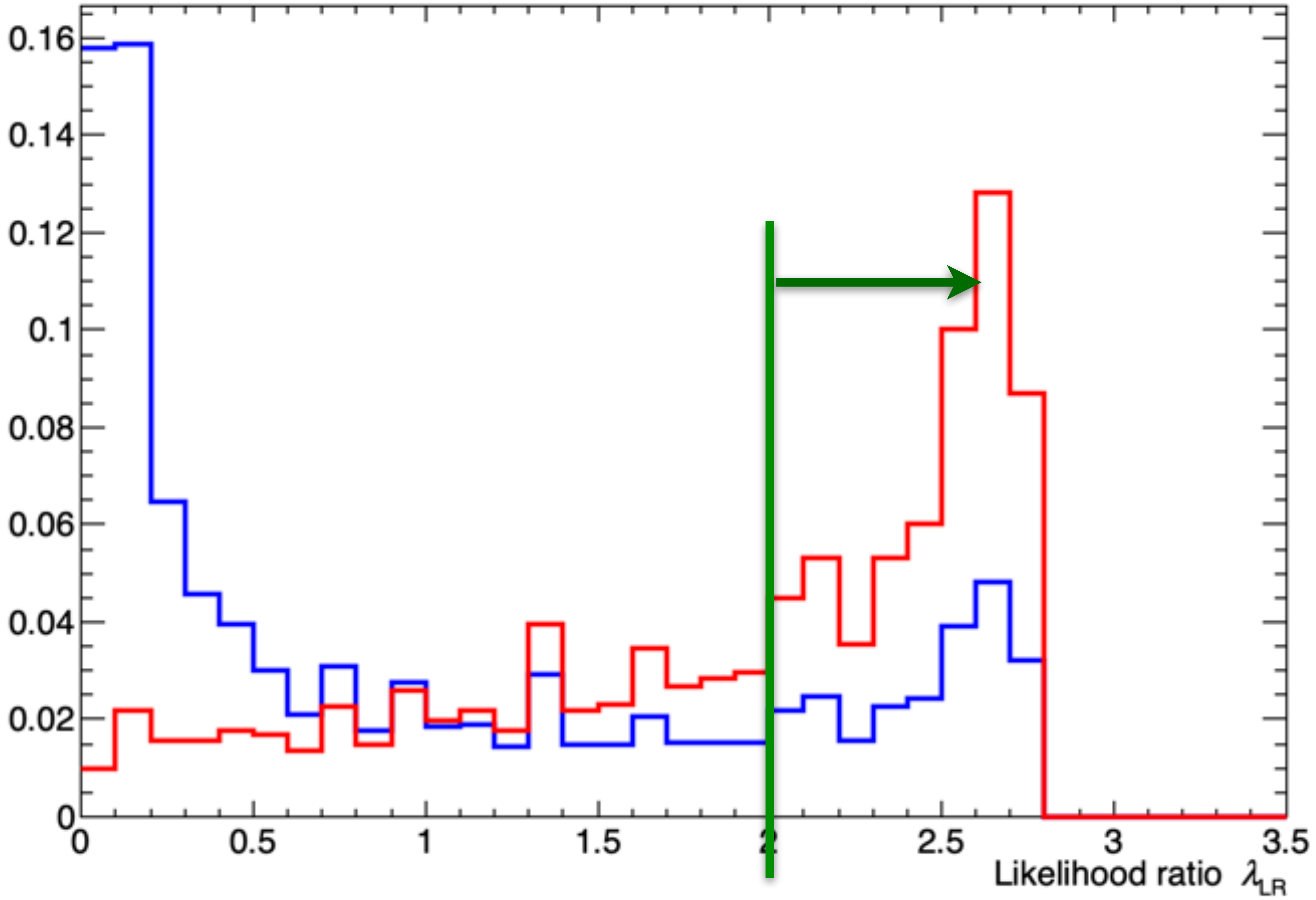
The Neyman-Pearson lemma

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



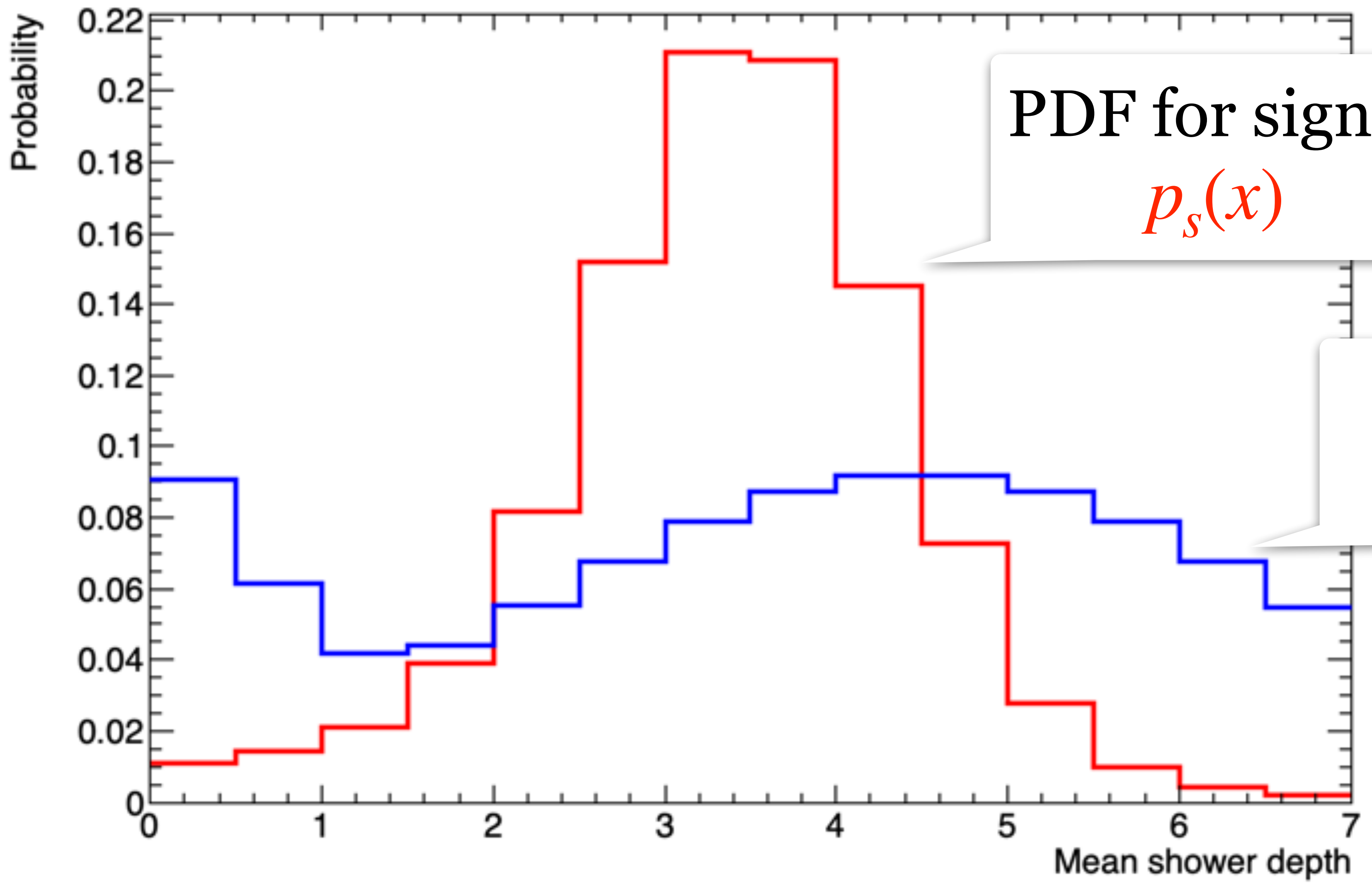
PDF for signal
$p_s(x)$

Background
$p_b(x)$



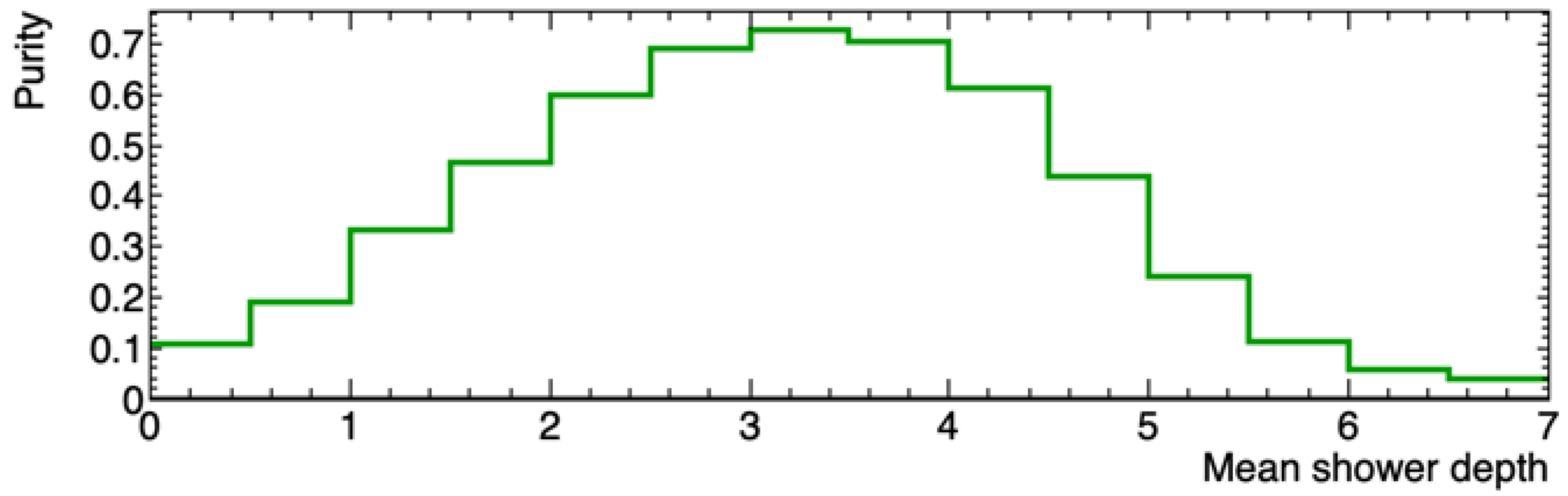
The Neyman-Pearson lemma

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

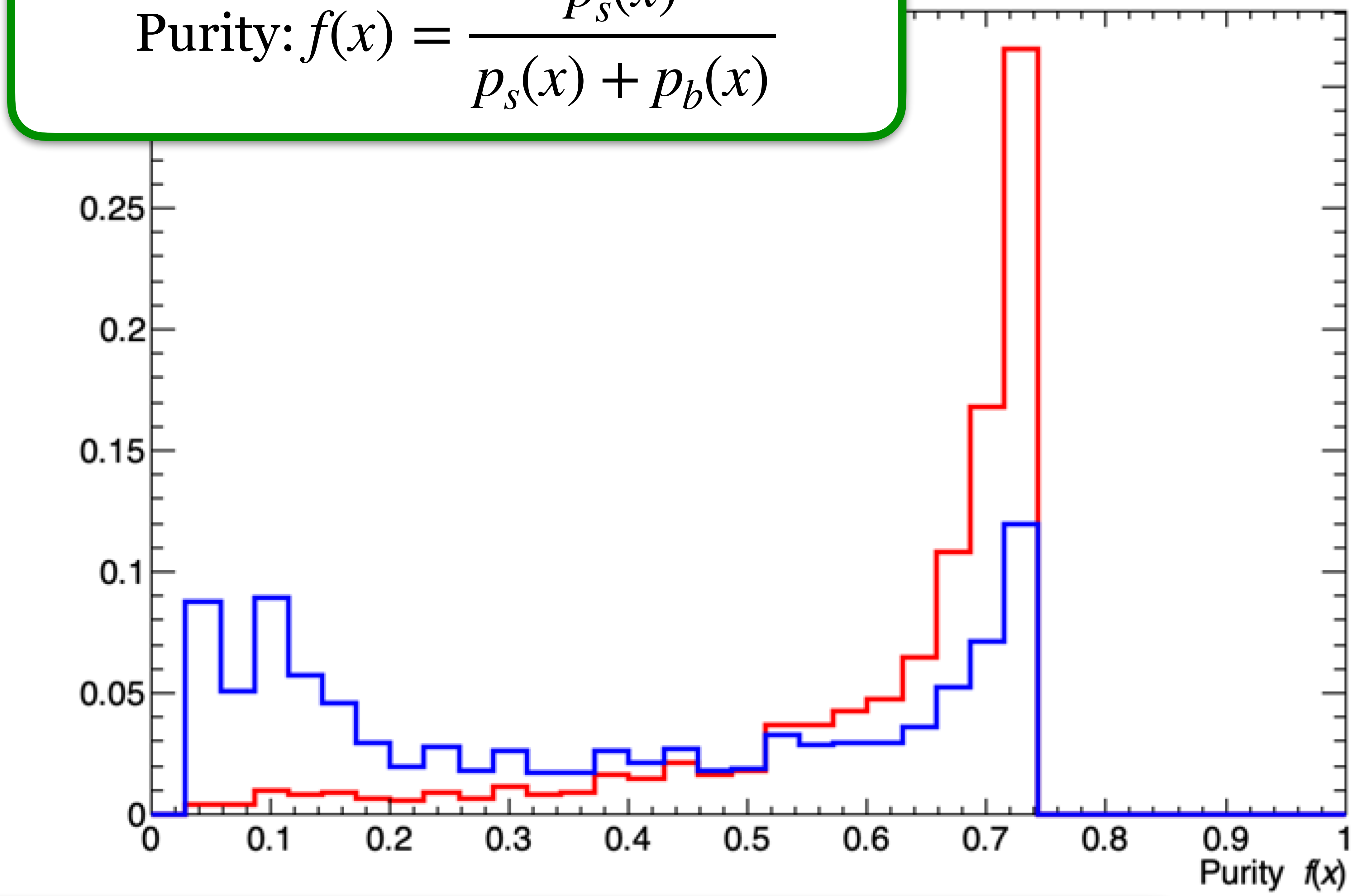


PDF for signal
$p_s(x)$

Background
$p_b(x)$



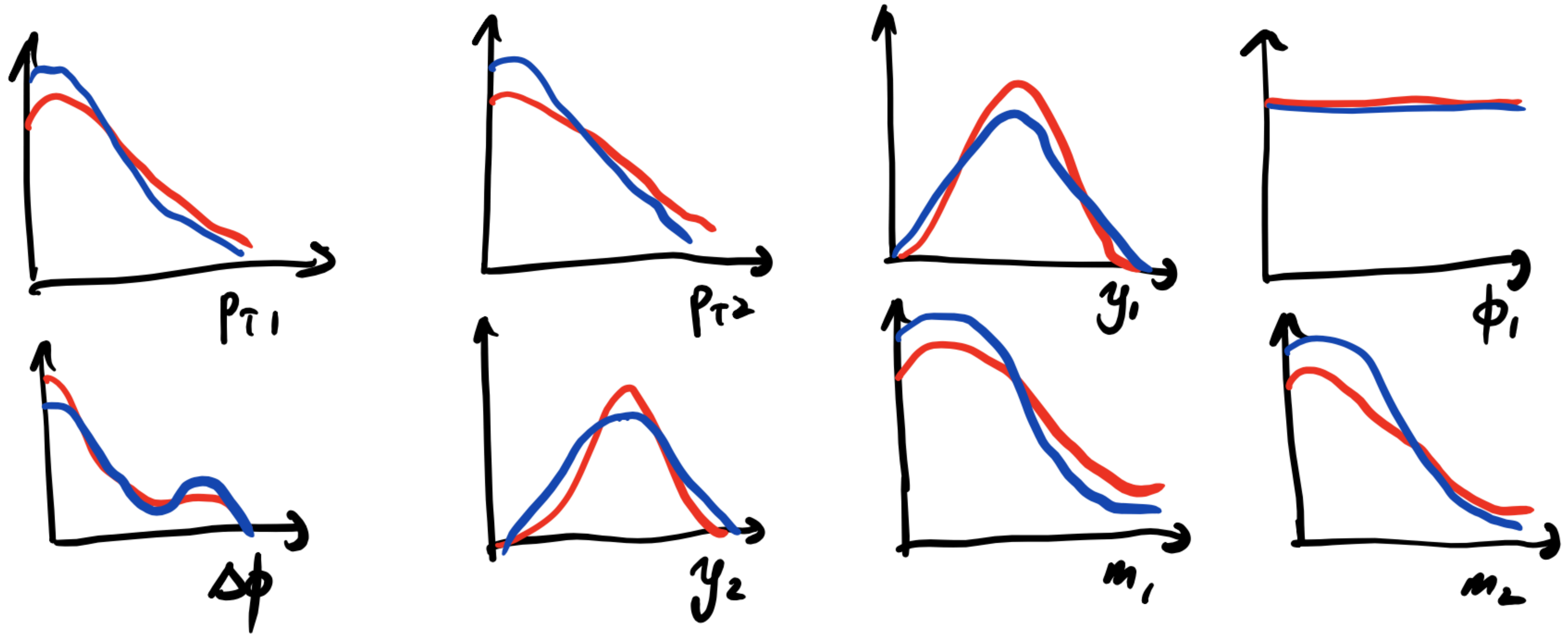
Purity: $f(x) = \dfrac{p_s(x)}{p_s(x) + p_b(x)}$



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

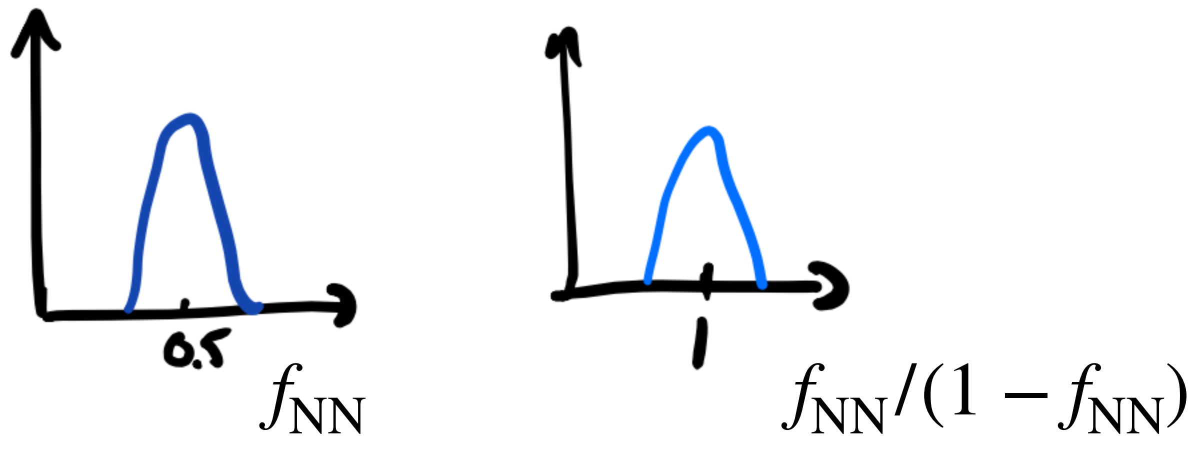
Using ML to reweight event samples

- 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 $\vec{x} = (x_1, \ldots, x_8)$



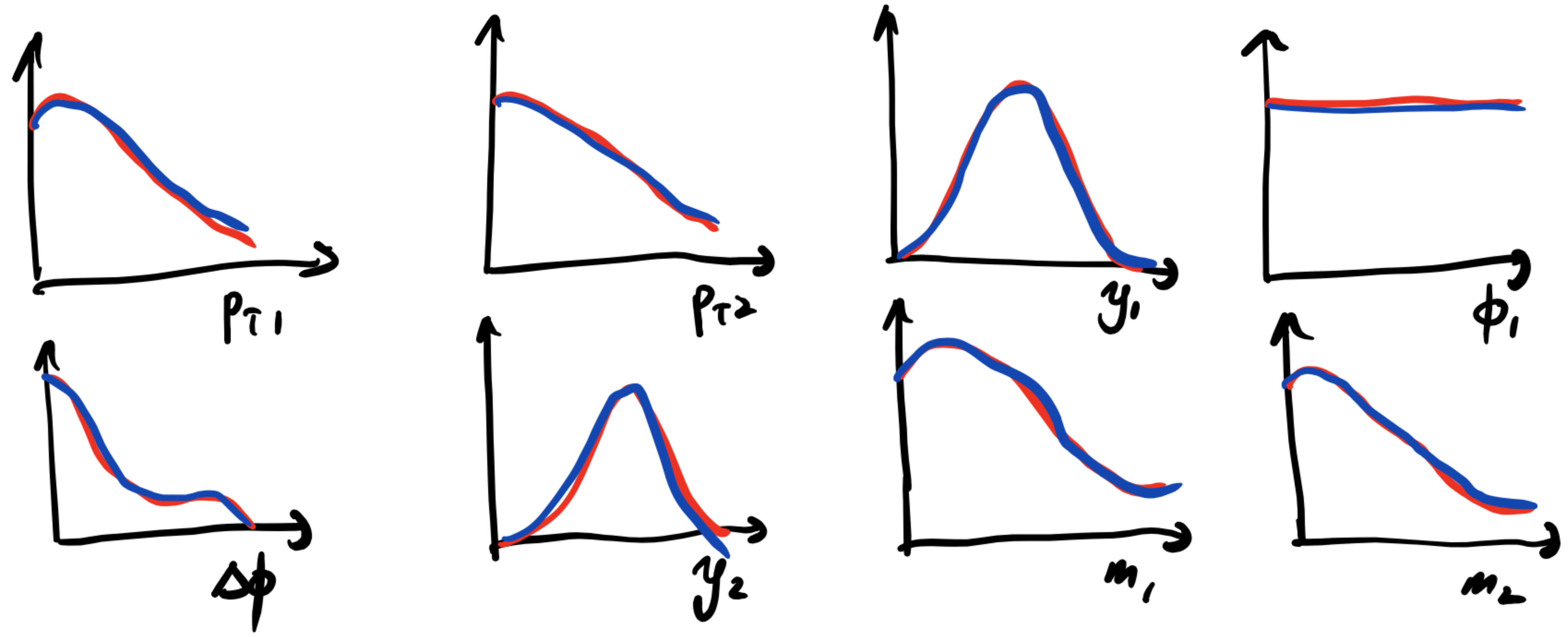
A neural network trained with cross entropy as loss function will return $f_{\mathrm{NN}}(\vec{x})$, that estimates the purity. An estimate of the likelihood ratio is given by

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



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!

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

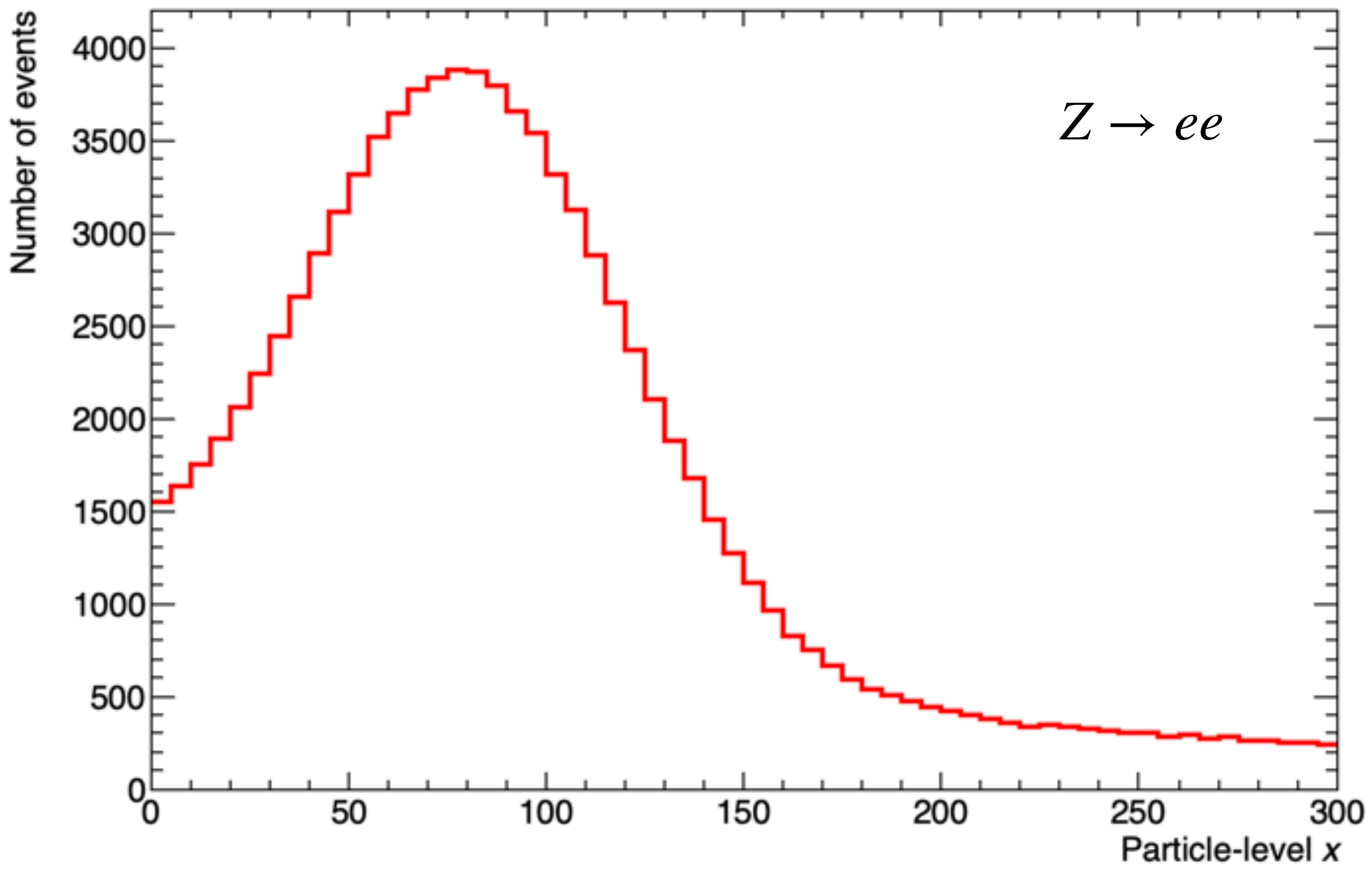
The NN → an 8-dimensional reweighing function

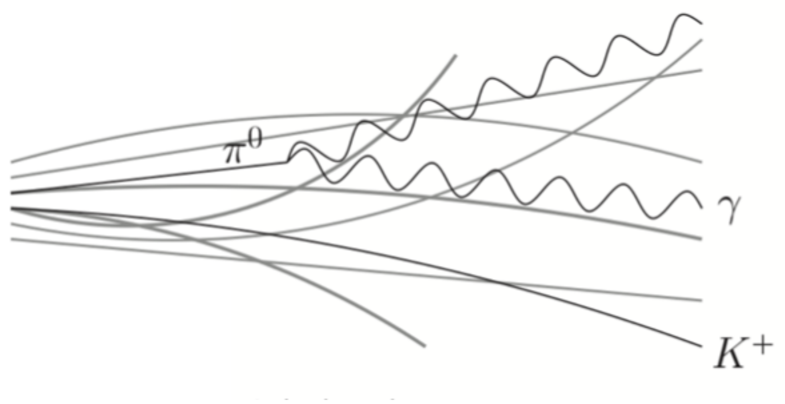
Using ML to weight events

- 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 …

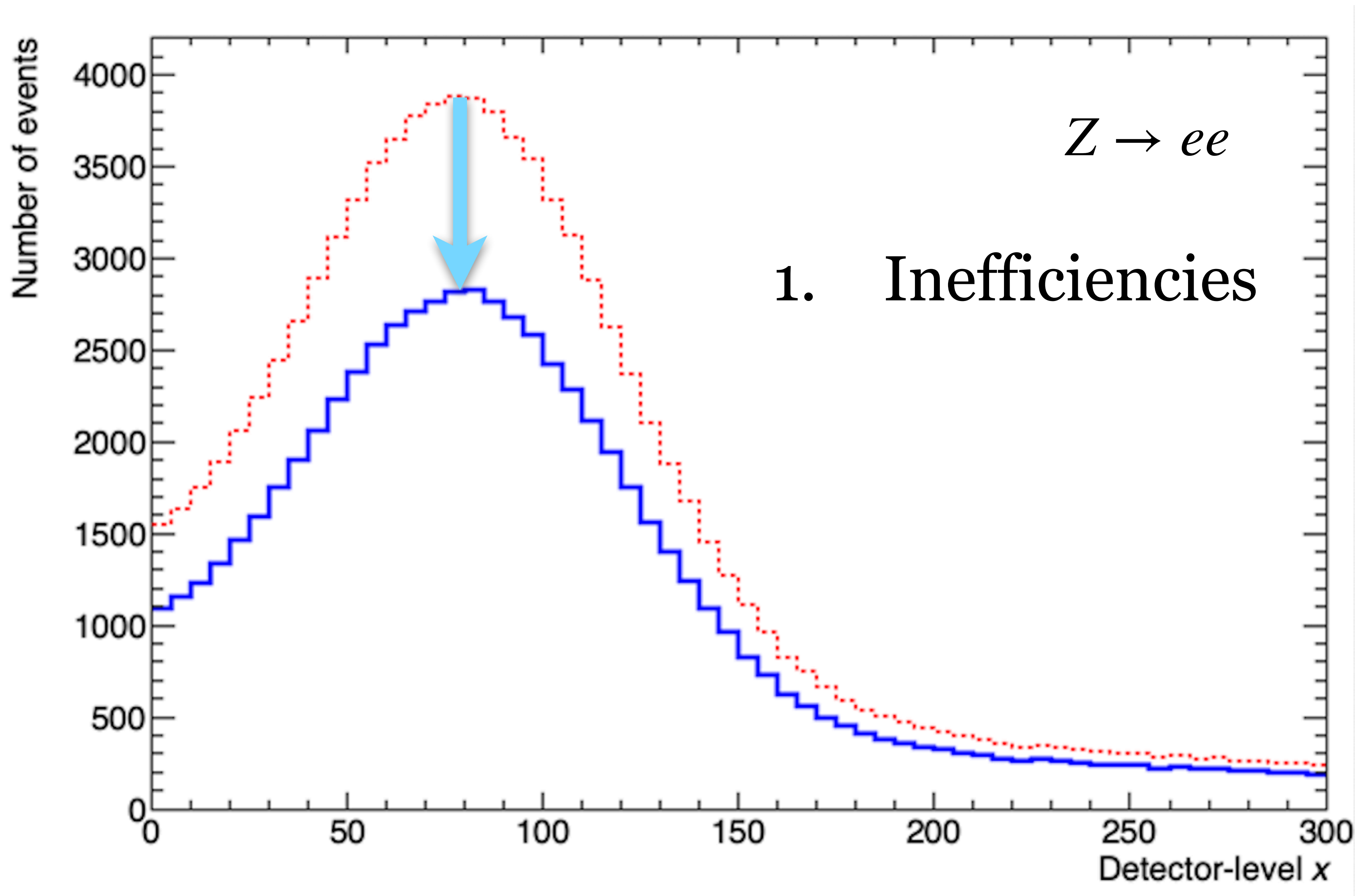
The Omnifold method

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

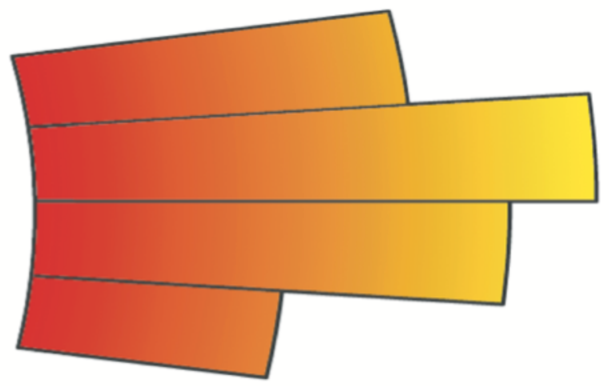




Interaction with the detector, two major effects



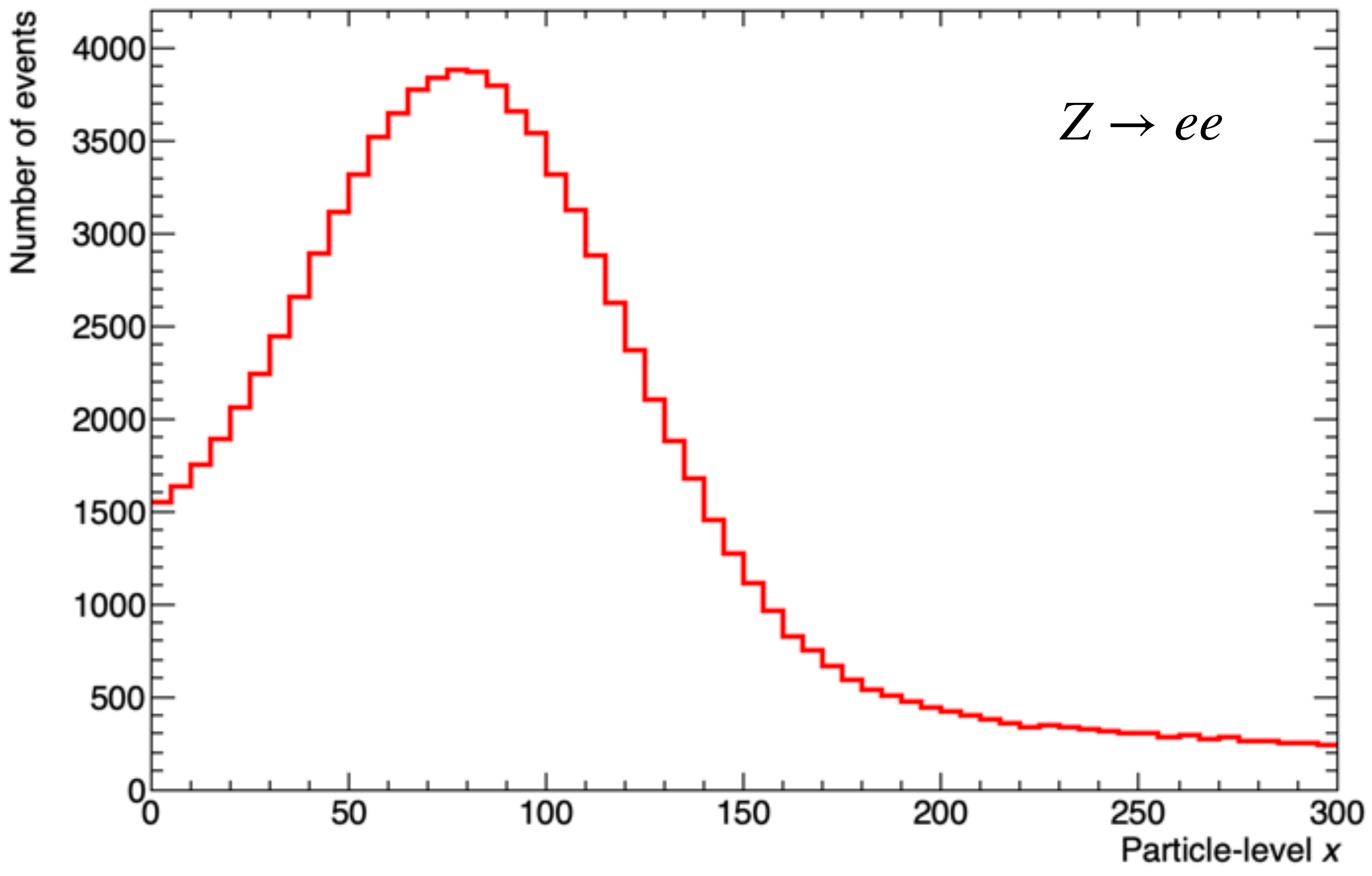
1. Inefficiencies

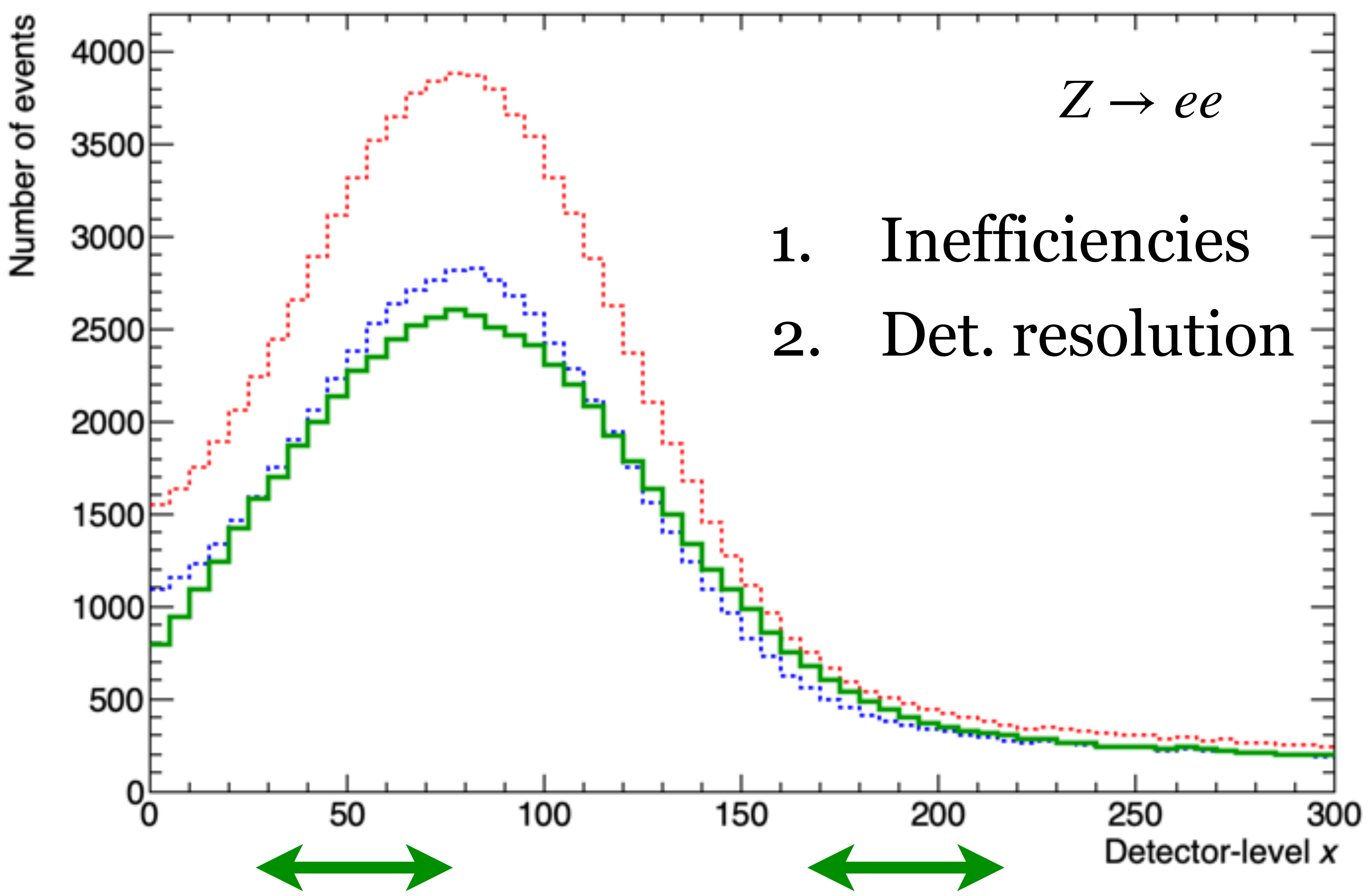


The Omnifold method

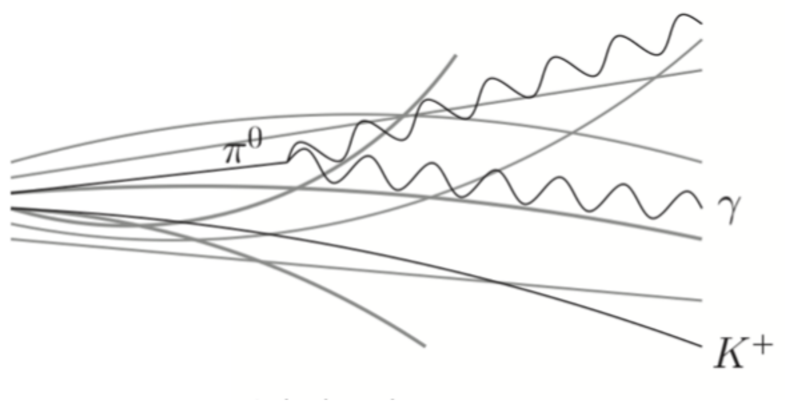
- The Omnifold method uses ML to perform unbinned, high-dimensional measurements
- This includes unfolding to the particle-level

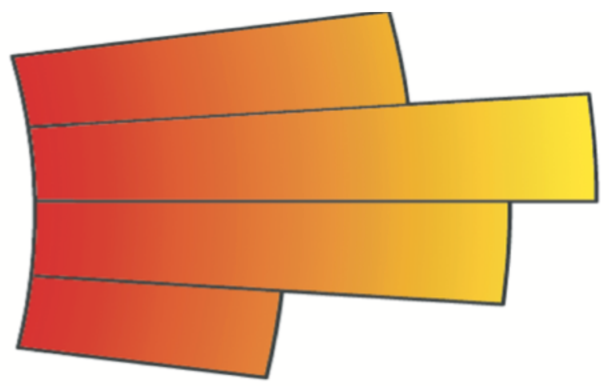
Interaction with the detector, two major effects





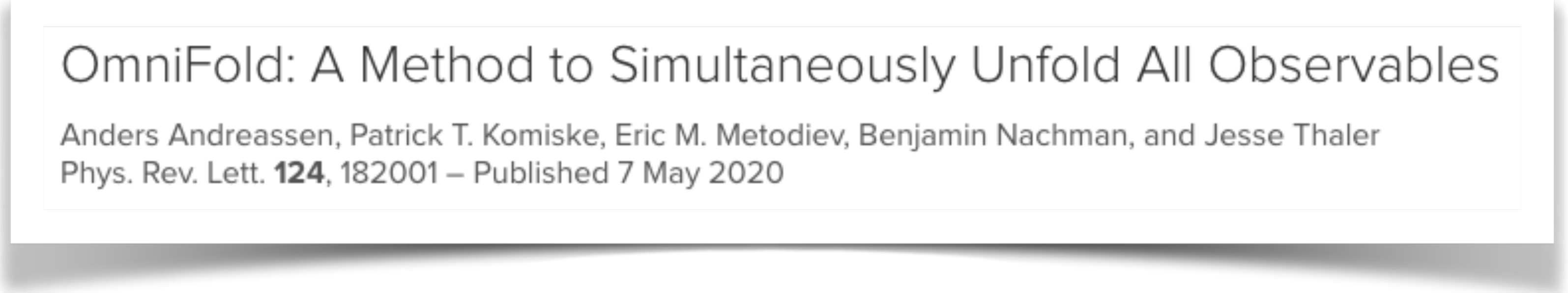
1. Inefficiencies
2. Det. resolution





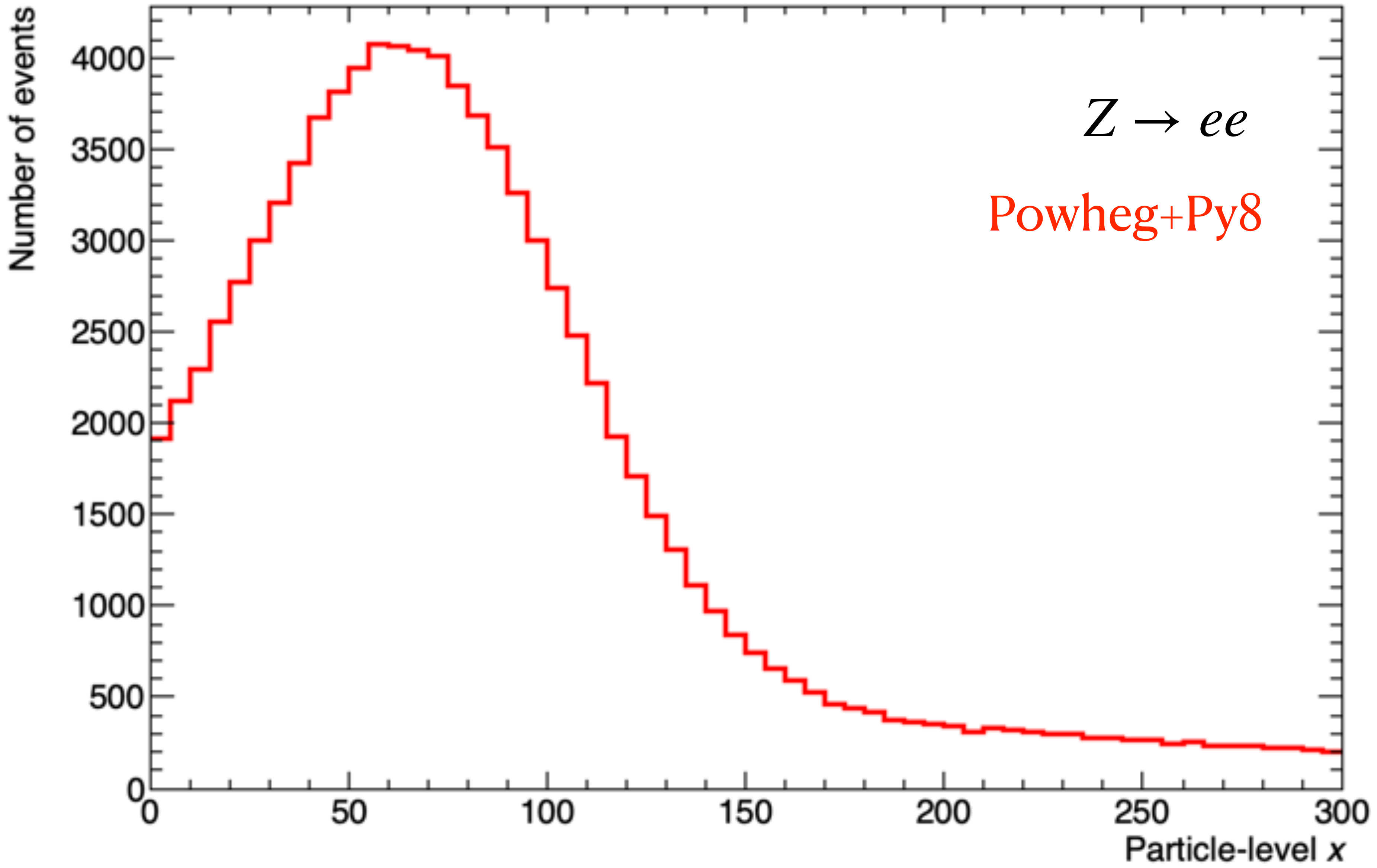
The Omnifold method

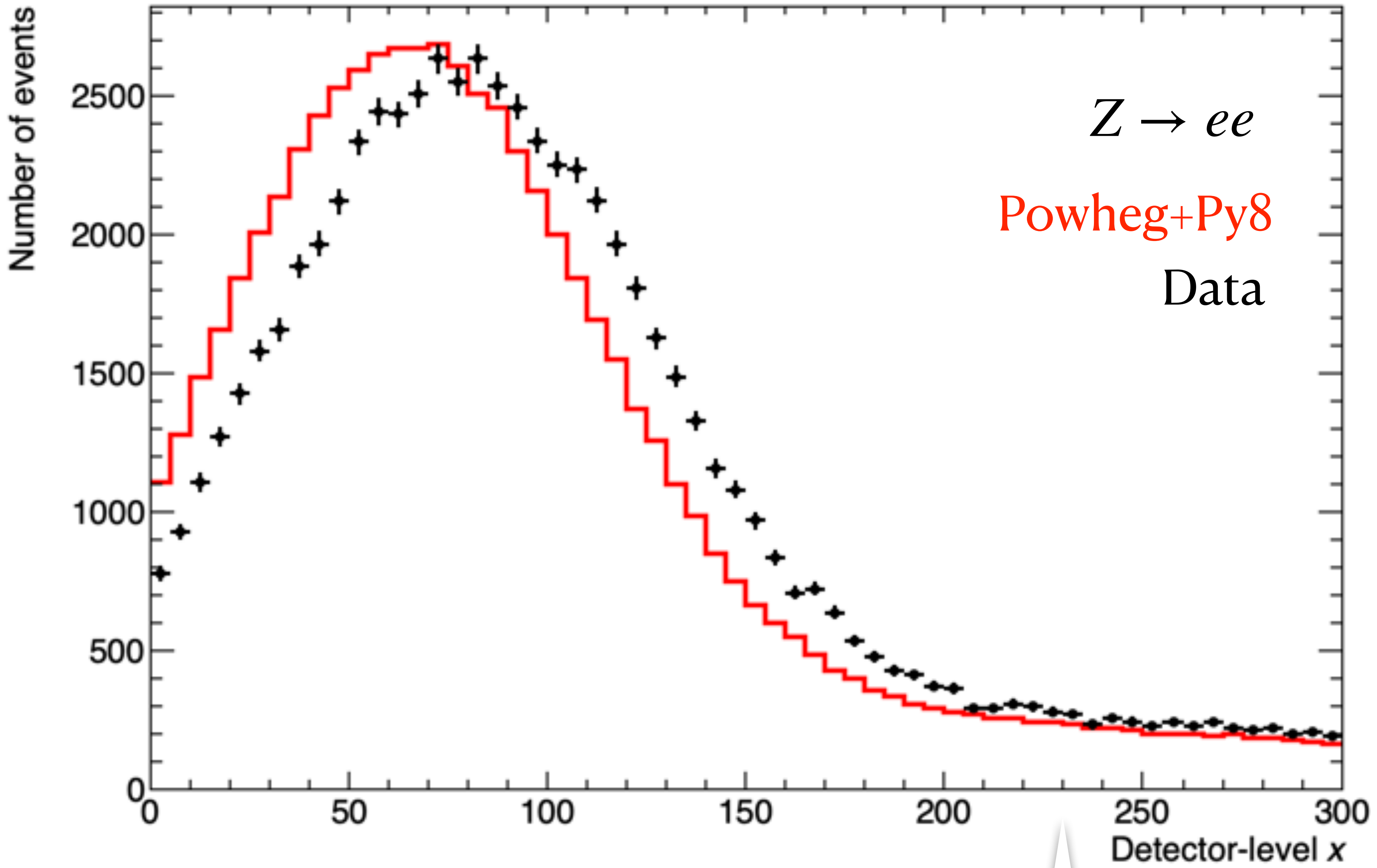
- The Omnifold method uses ML to perform unbinned, high-dimensional measurements
- This includes unfolding to the particle-level

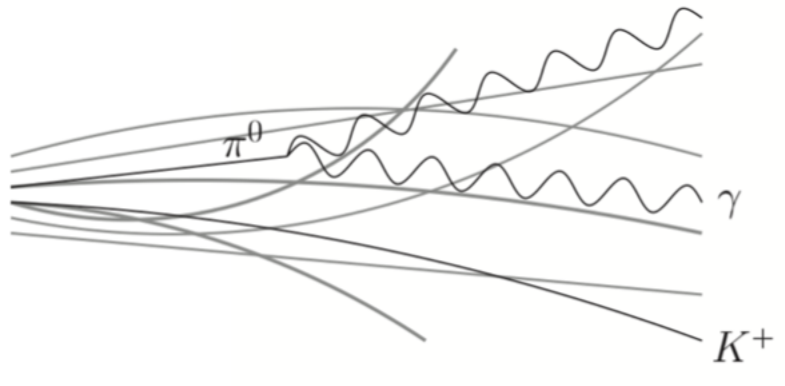


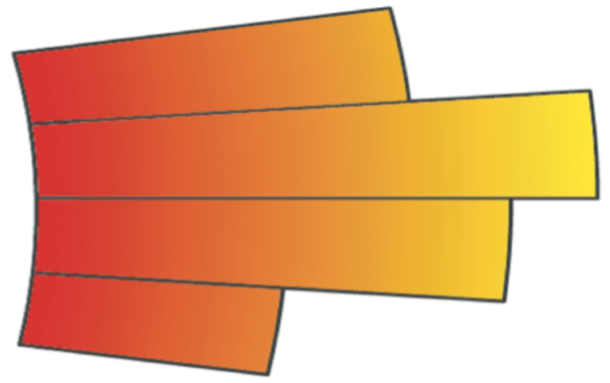
OmniFold: A Method to Simultaneously Unfold All Observables

Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler
Phys. Rev. Lett. **124**, 182001 – Published 7 May 2020





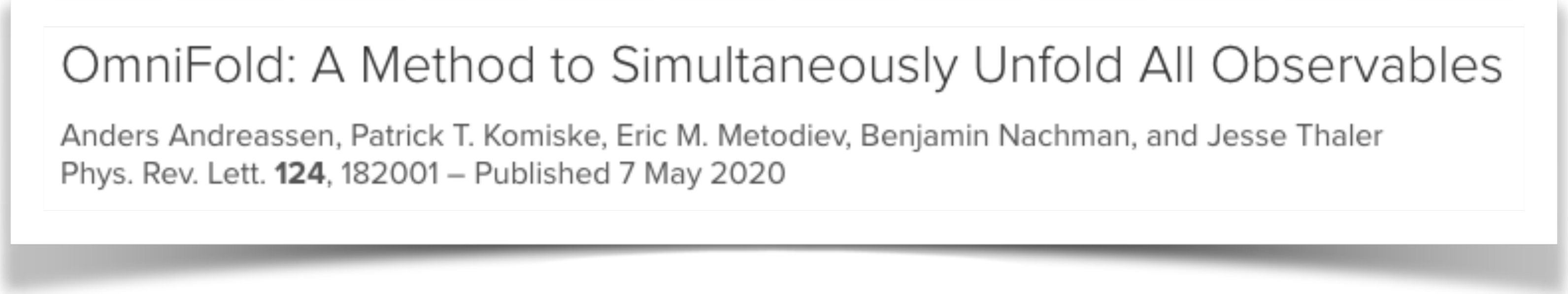




1. Train NN with data as signal, MC bkg
Use to reweight!

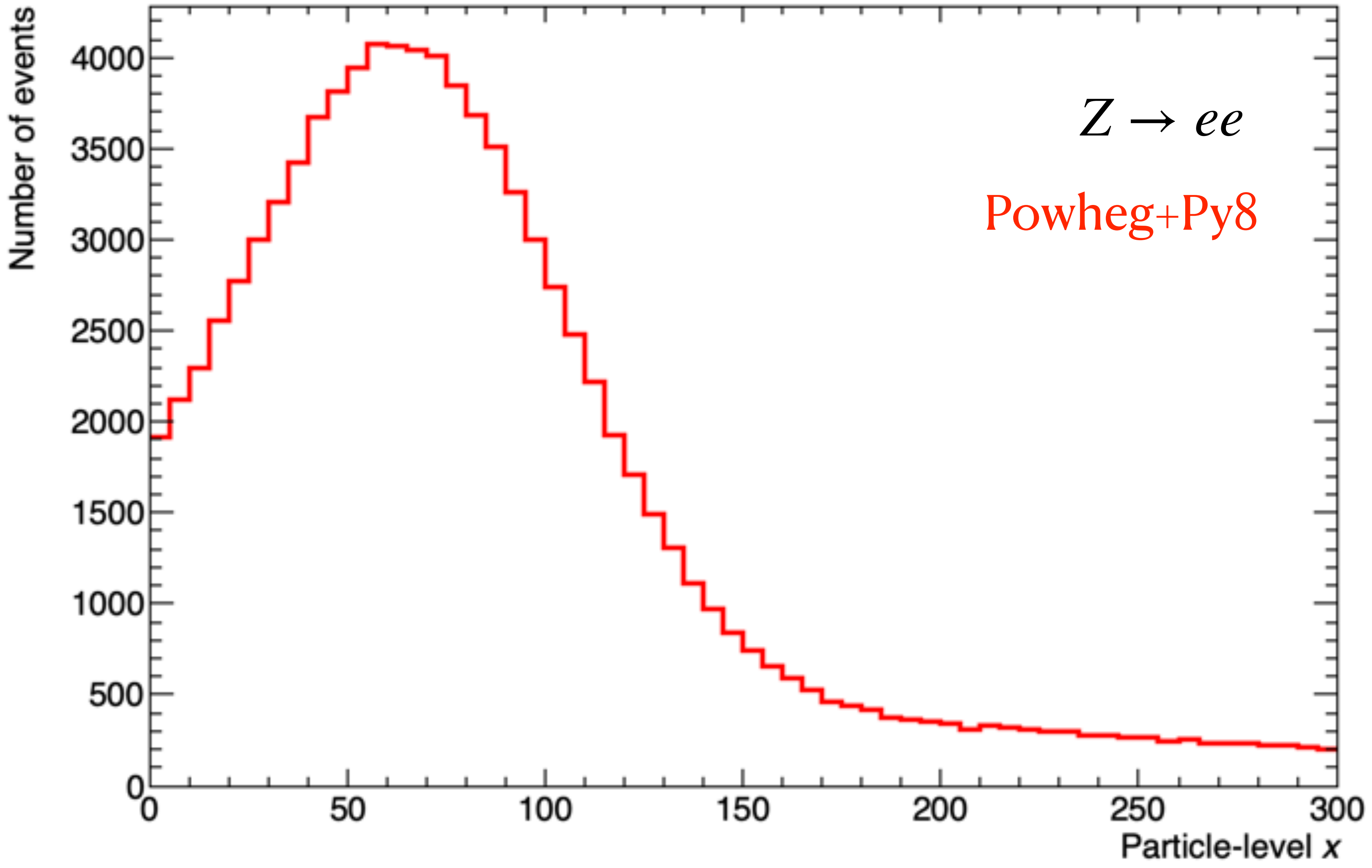
The Omnifold method

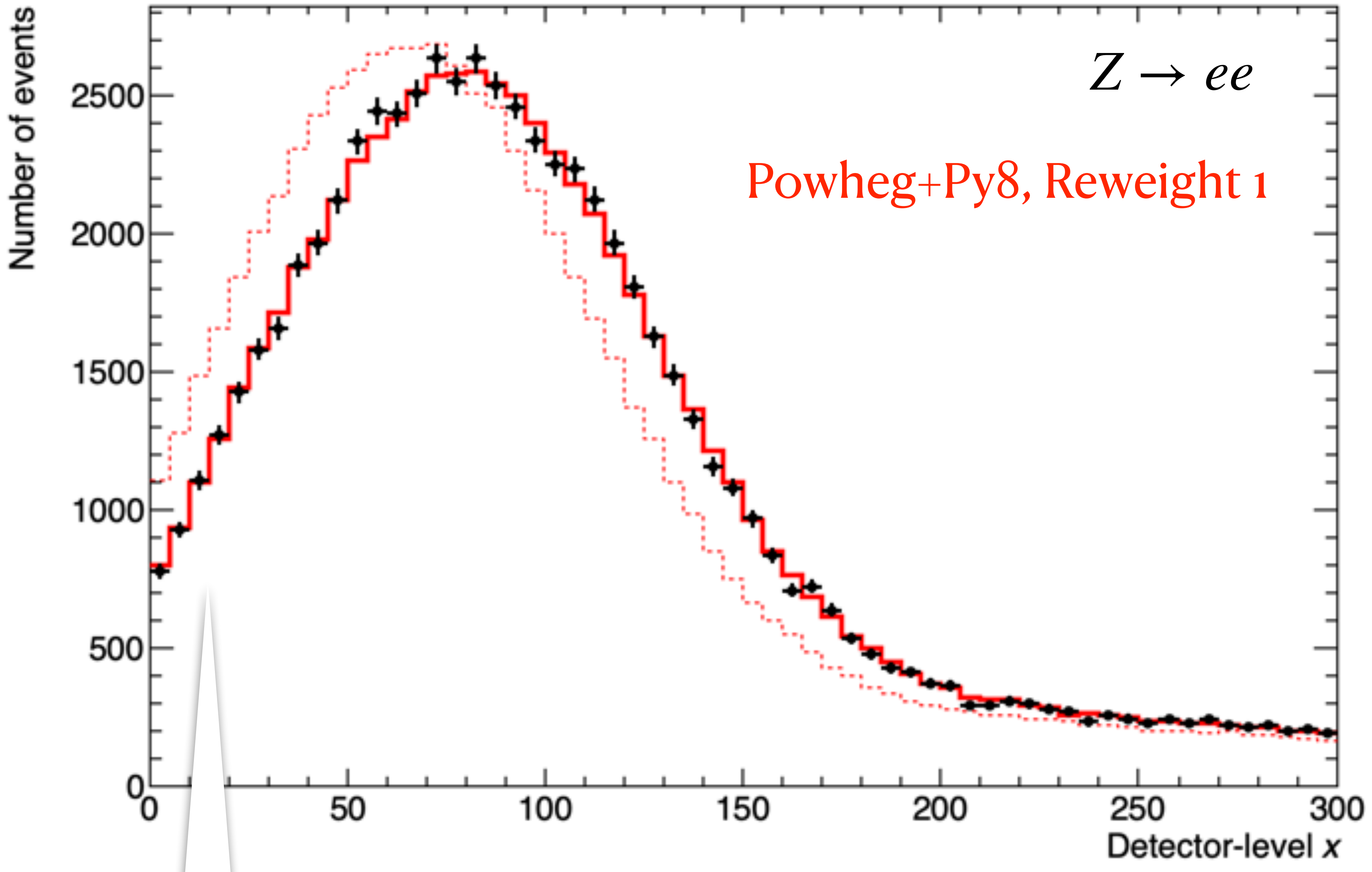
- The Omnifold method uses ML to perform unbinned, high-dimensional measurements
- This includes unfolding to the particle-level

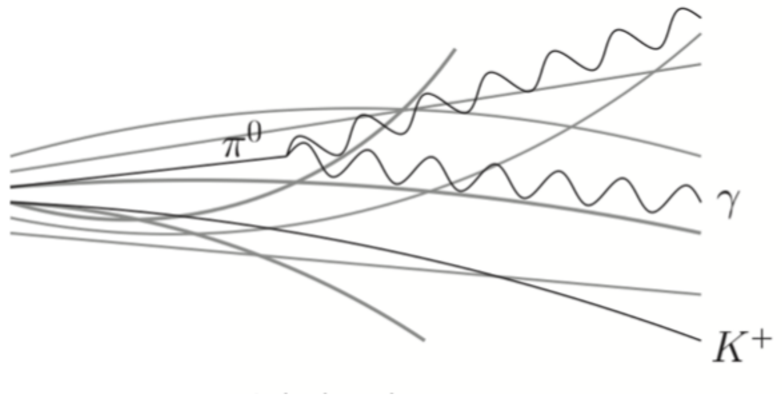


OmniFold: A Method to Simultaneously Unfold All Observables

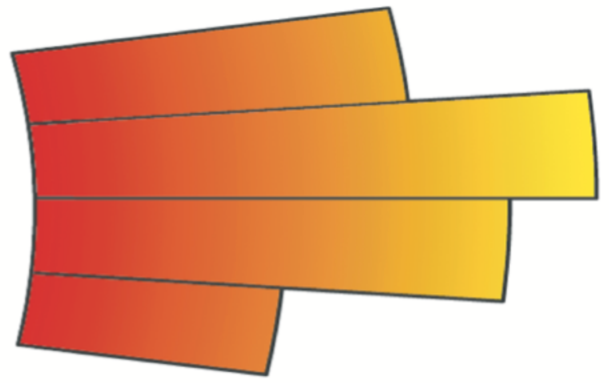
Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler
Phys. Rev. Lett. **124**, 182001 – Published 7 May 2020







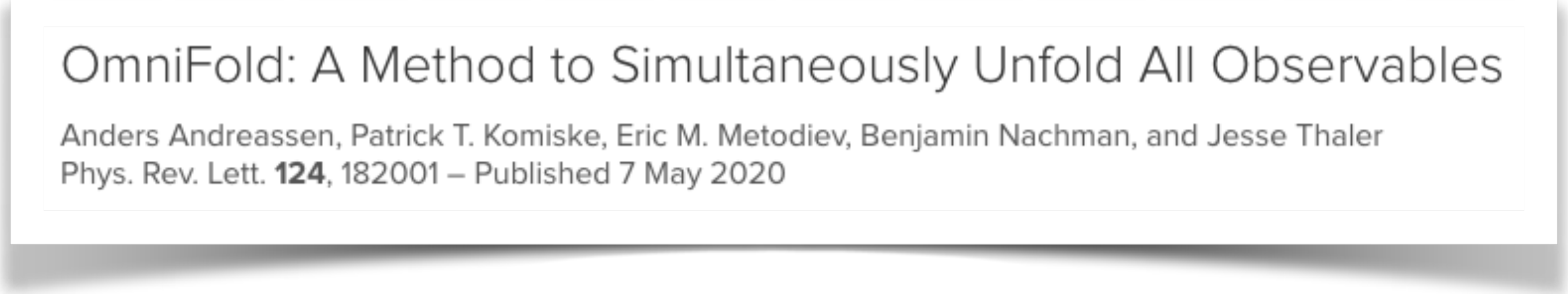
2. Each simulated event has obtained a weight. Propagate this to the partilcle level distribution



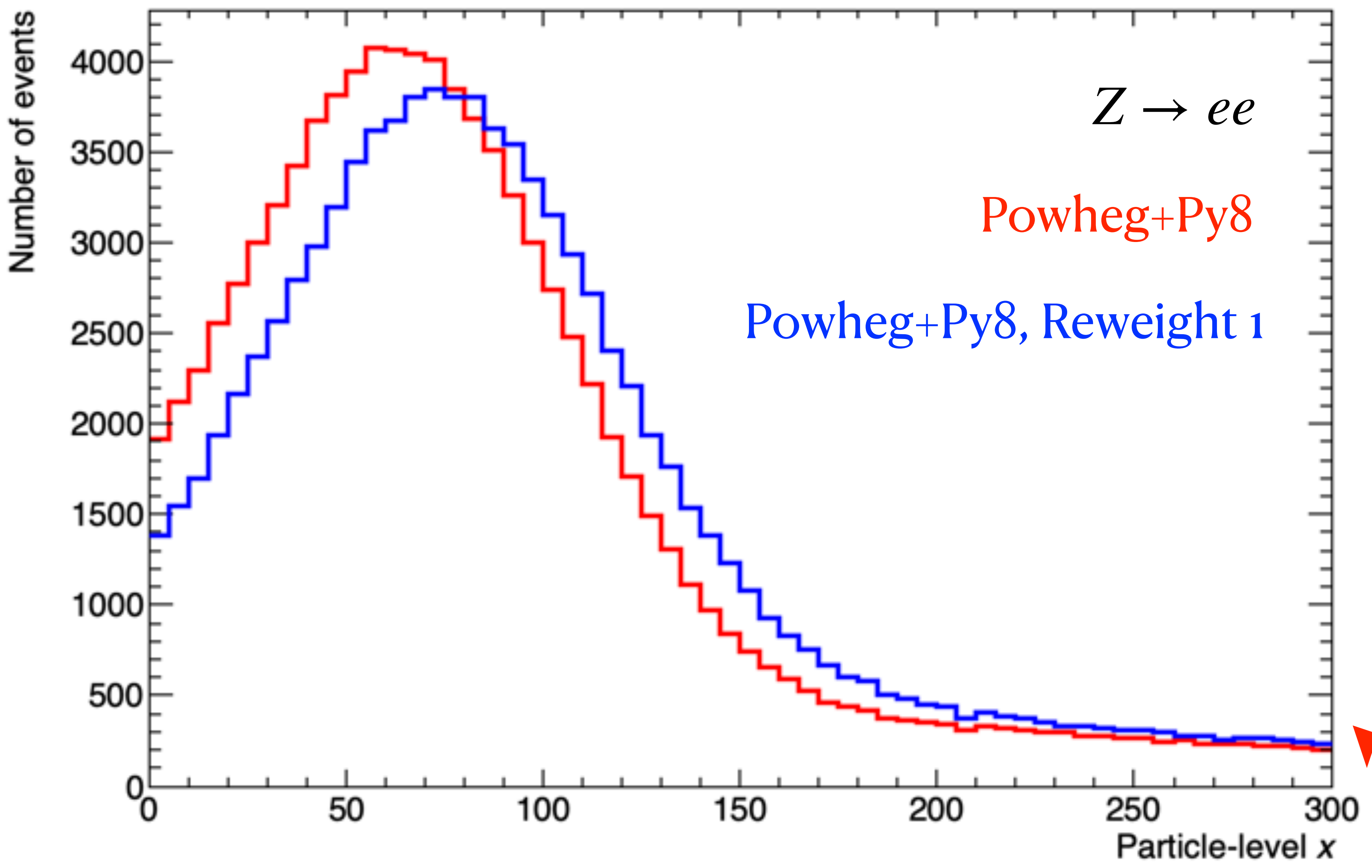
1. Train NN with data as signal, MC bkg
Use to reweight!

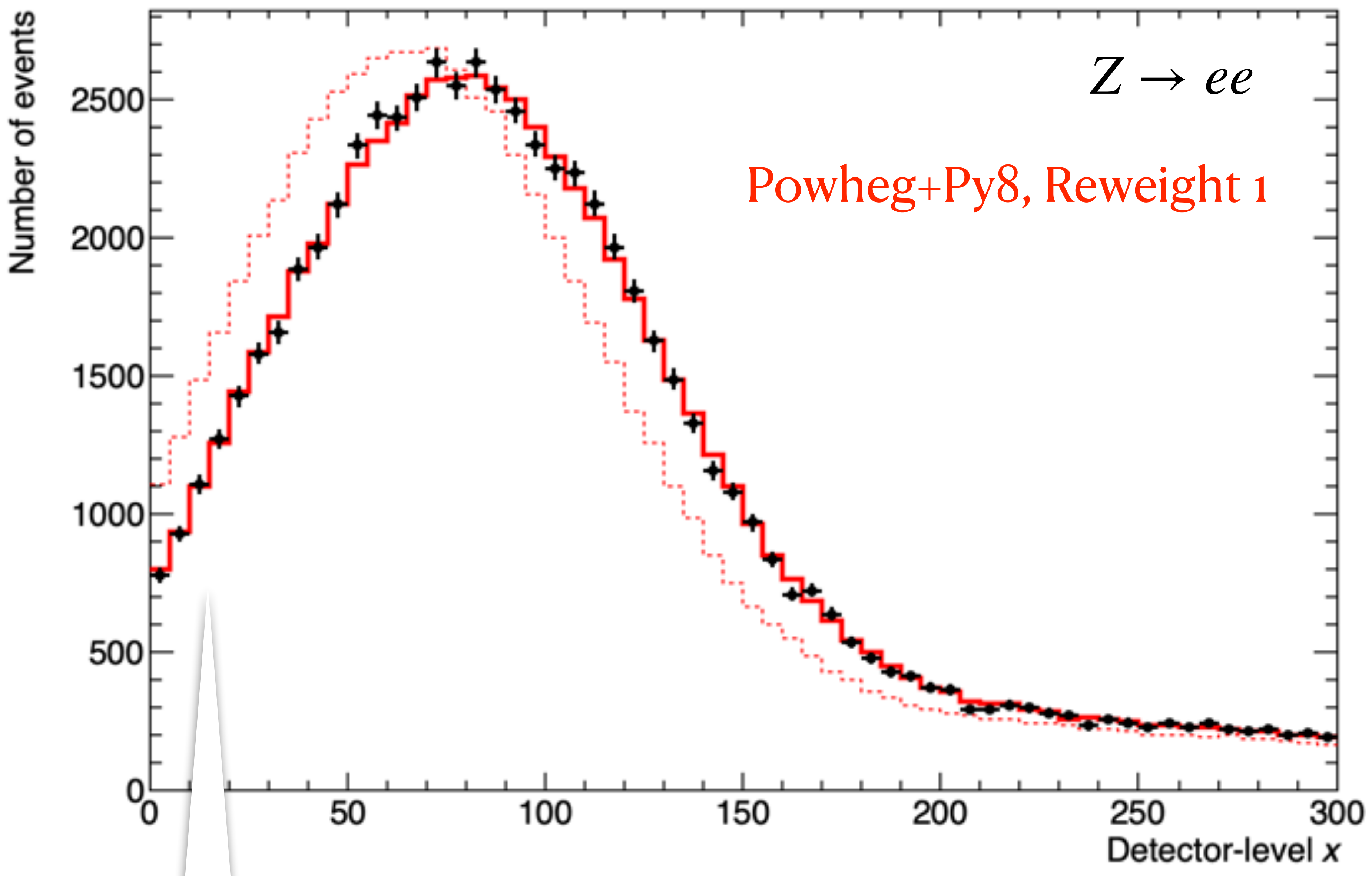
The Omnifold method

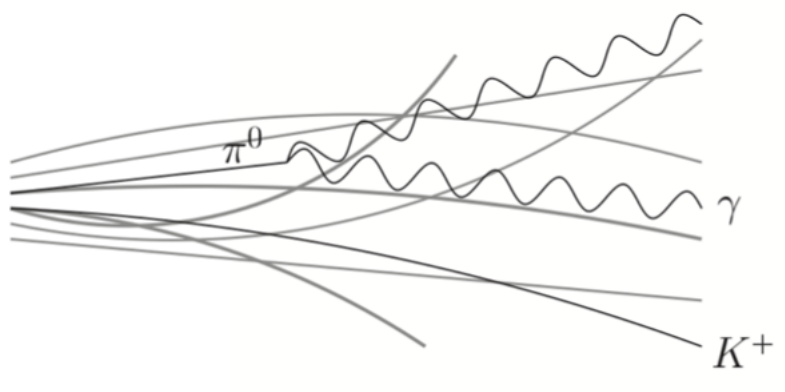
- The Omnifold method uses ML to perform unbinned, high-dimensional measurements
- This includes unfolding to the particle-level


OmniFold: A Method to Simultaneously Unfold All Observables

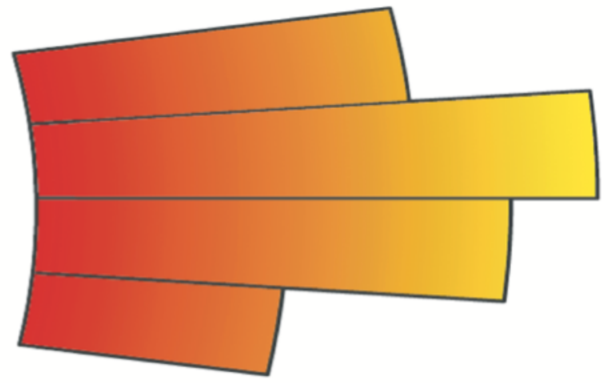
Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler
Phys. Rev. Lett. **124**, 182001 – Published 7 May 2020







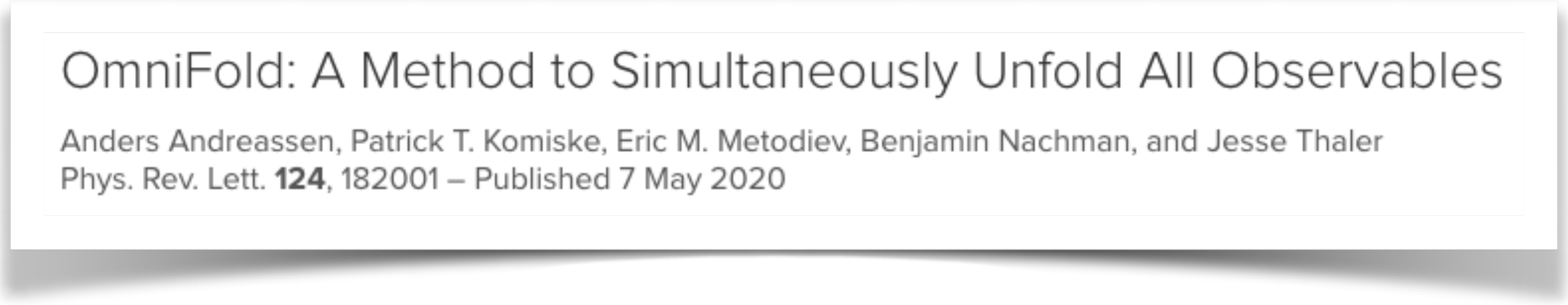
2. Each simulated event has obtained a weight. Propagate this to the partilcle level distribution



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

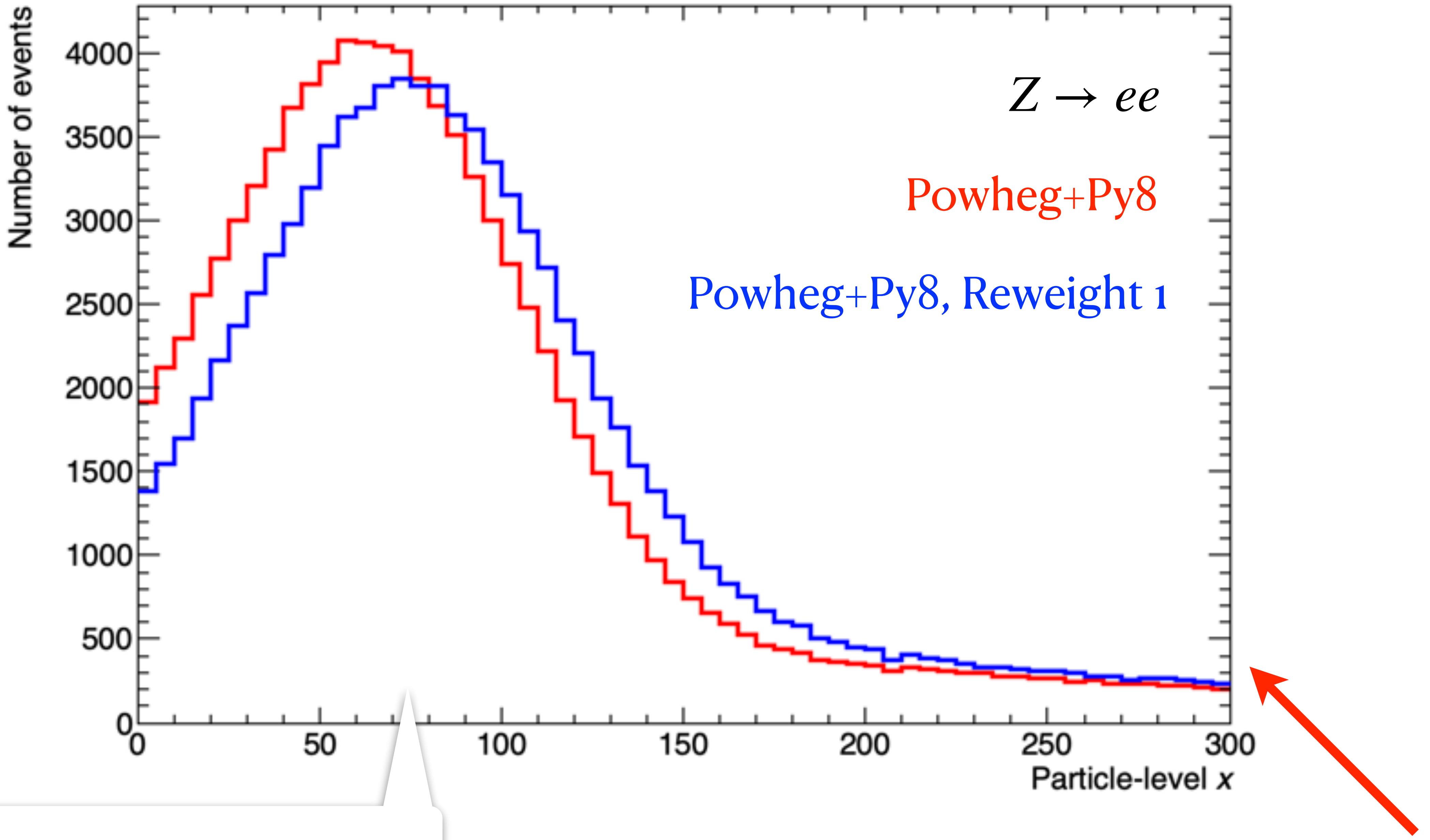
The Omnifold method

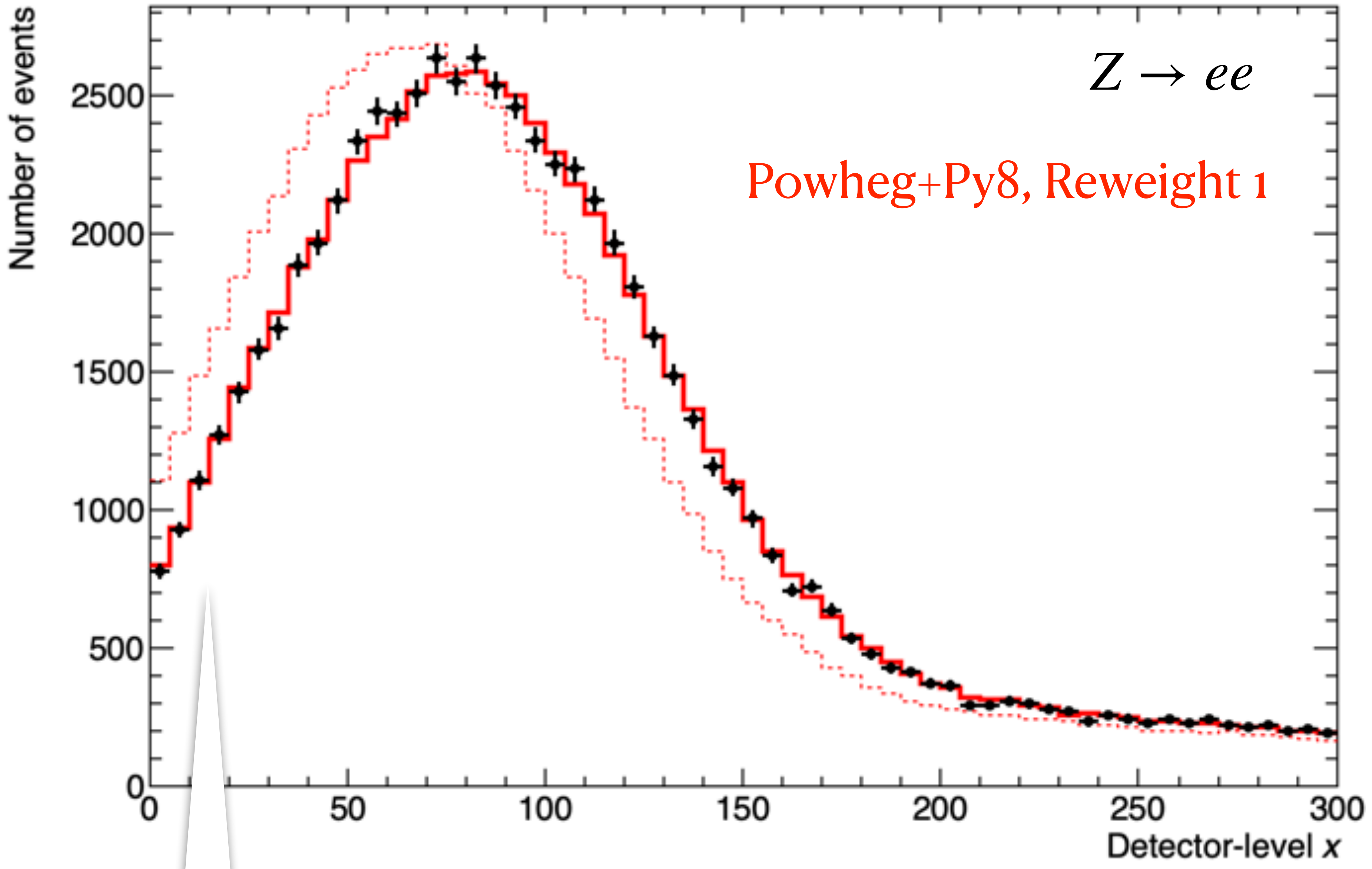
- The Omnifold method uses ML to perform unbinned, high-dimensional measurements
- This includes unfolding to the particle-level



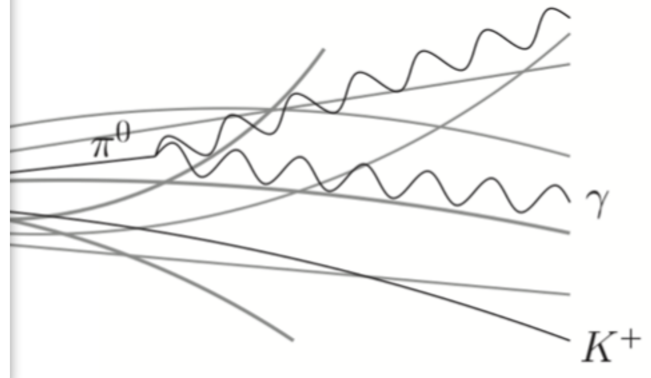
OmniFold: A Method to Simultaneously Unfold All Observables

Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler
Phys. Rev. Lett. **124**, 182001 – Published 7 May 2020

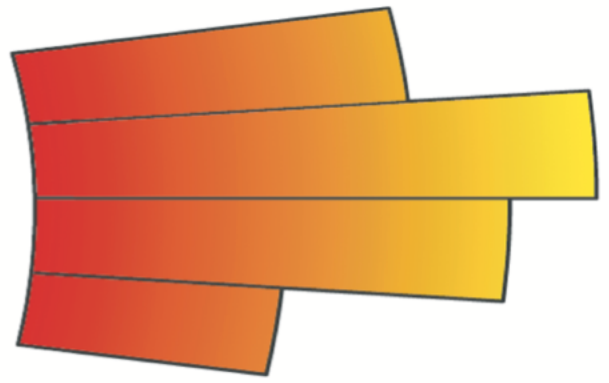




3. Train a new network using using blue as signal, red background (MC-MC). Use to reweight red!



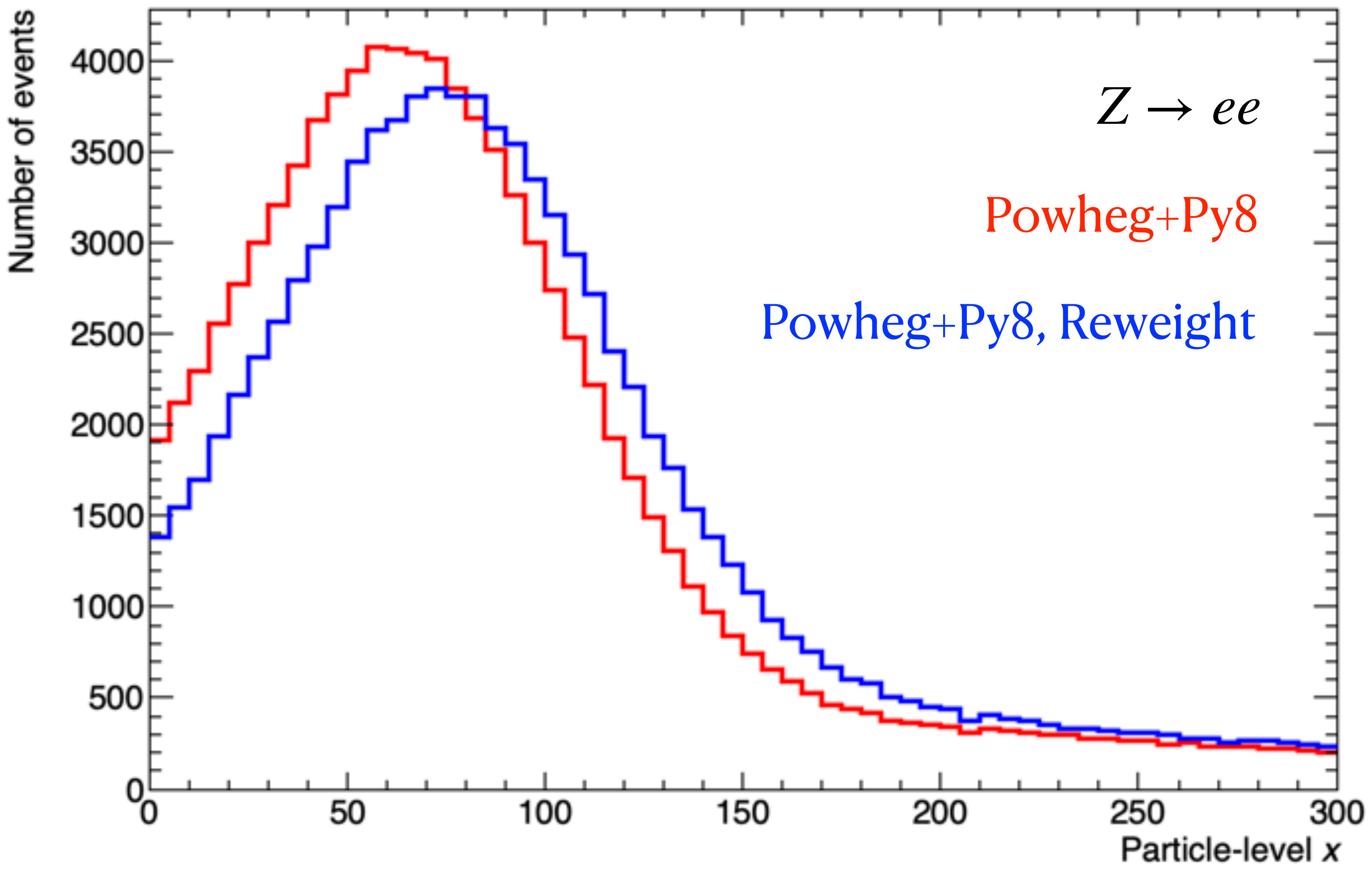
2. Each simulated event has obtained a weight. Propagate this to the partilcle level distribution

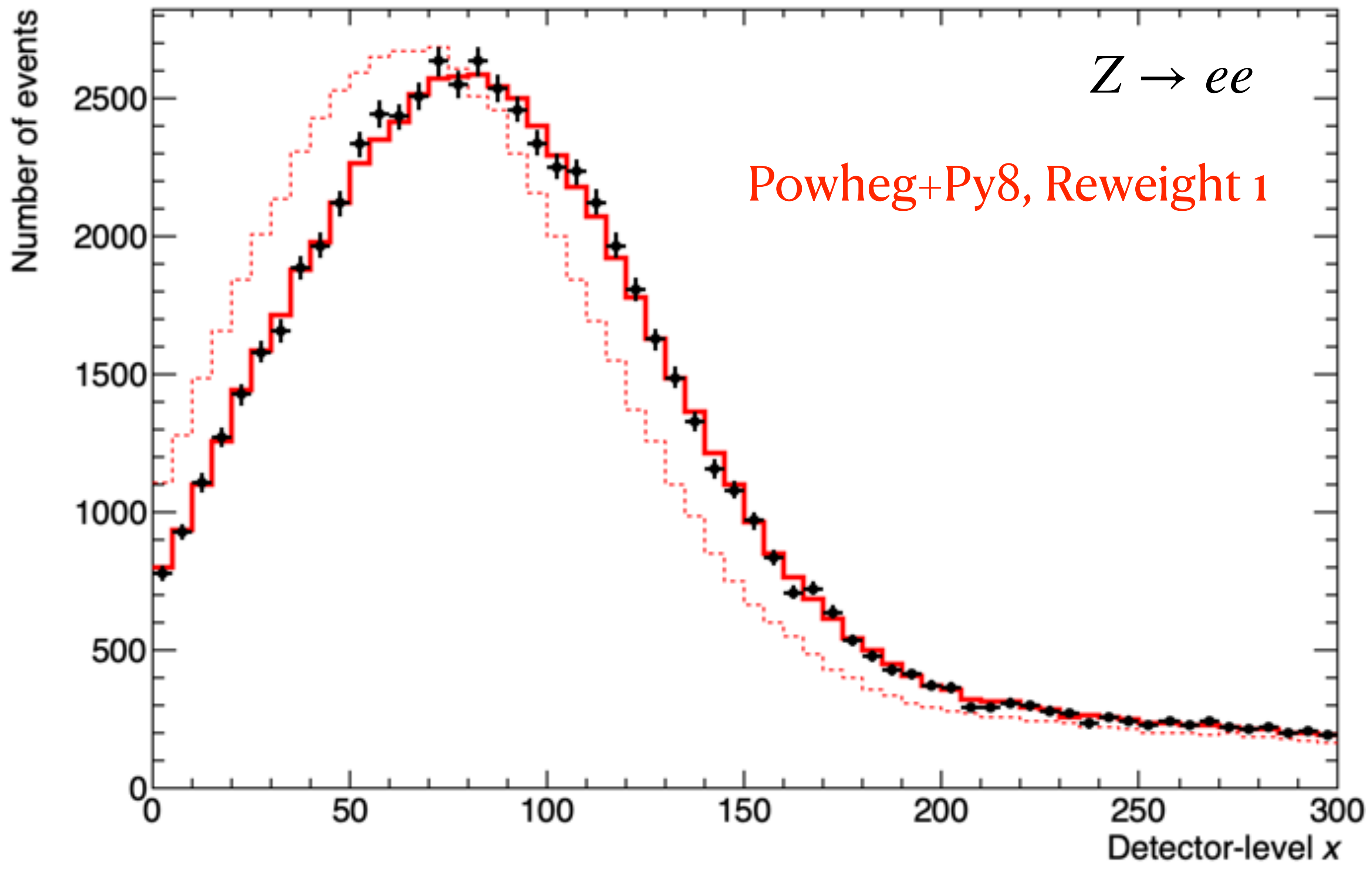


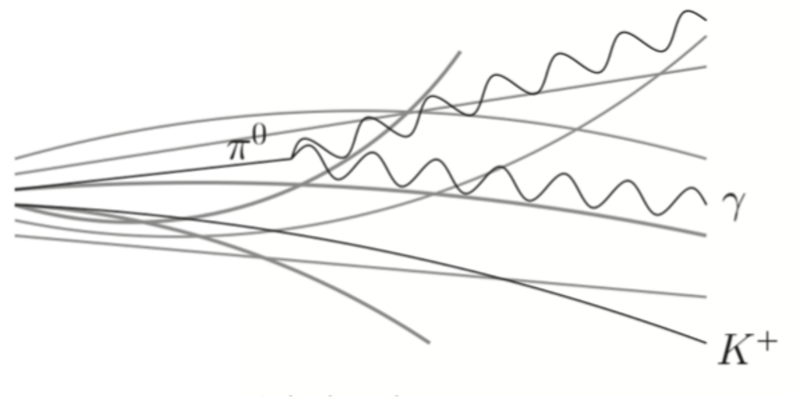
1. Train NN with data as signal, MC bkg Use to reweight!

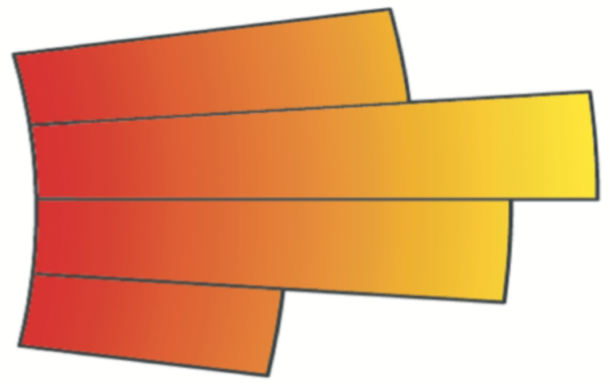
The Omnifold method

- 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



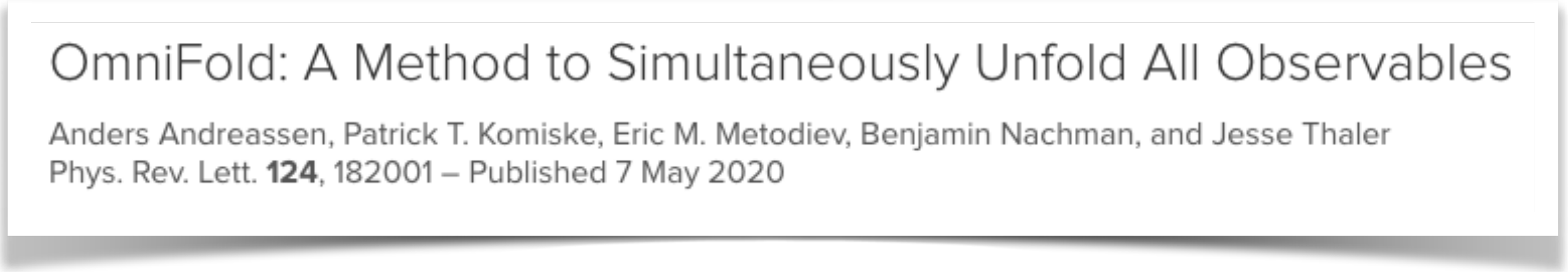






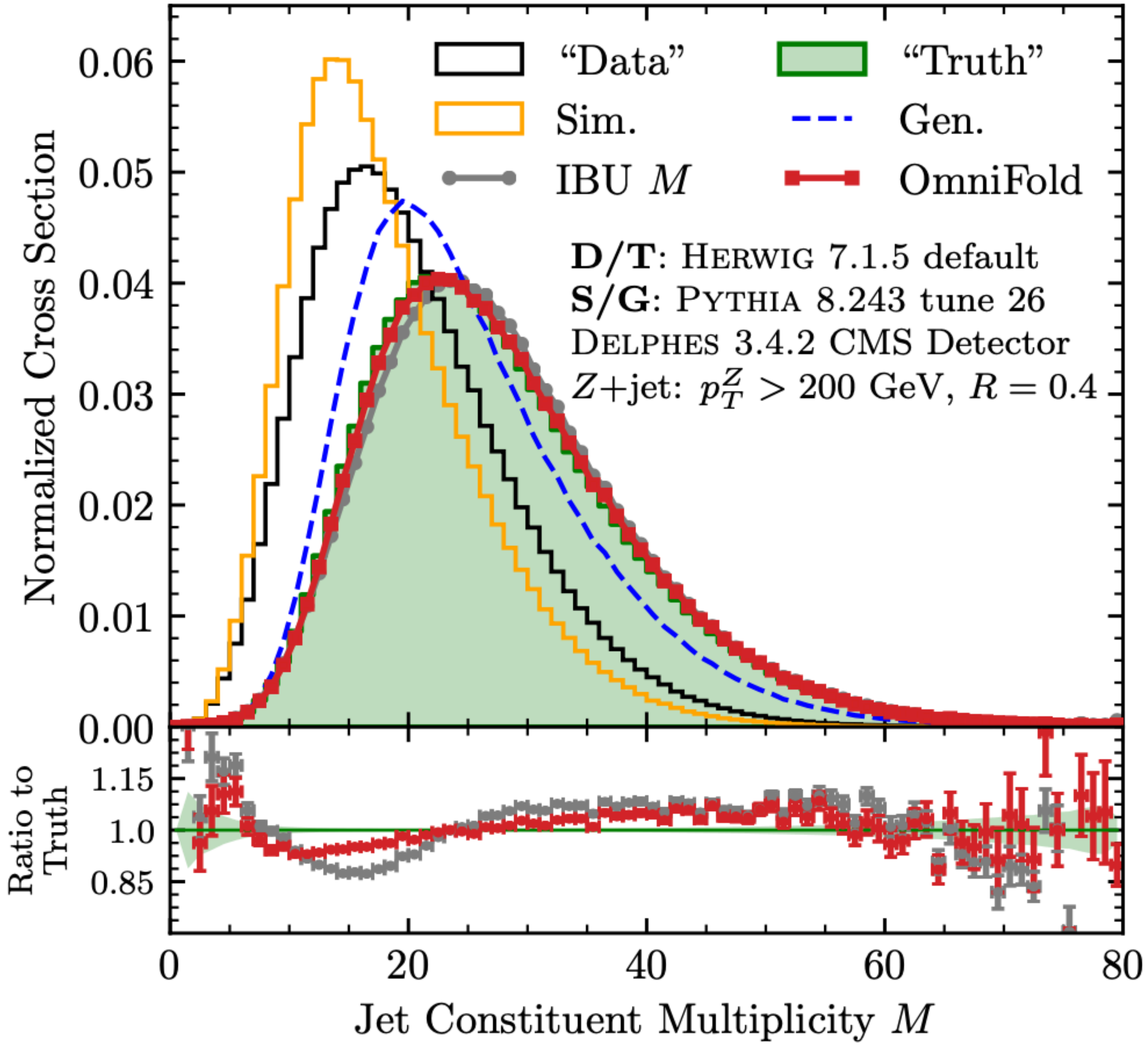
The Omnifold method

- 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’

Full Omnifold

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

Energy Flow Networks: Deep Sets for Particle Jets

Patrick T. Komiske, Eric M. Metodiev, Jesse Thaler

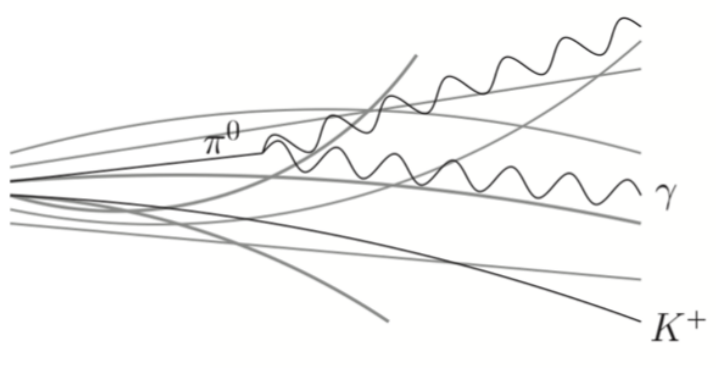
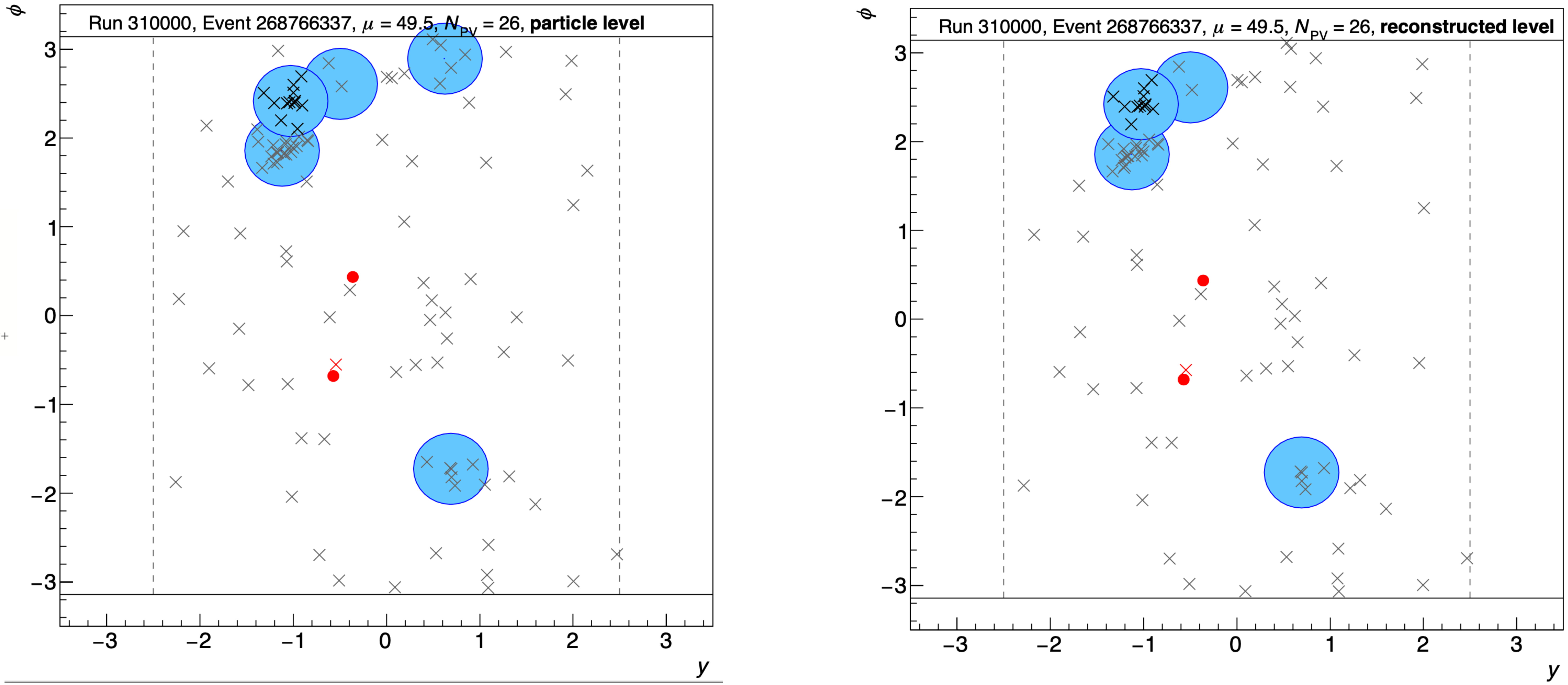


Illustration of
All changed particles
Produced in high pT
$Z \rightarrow \mu\mu$ events



Input and output of the unfolding (1/3)

- When we unfold, we have real data as input and MC simulated input
- The data only has reconstructed level quantiles, we can write the data as:

Data sample: $\vec{X}^{\text{reco}}_{\text{data}} = \{X^{\text{reco}}_{\text{data},1}, X^{\text{reco}}_{\text{data},2}, \ldots, X^{\text{reco}}_{\text{data},N_{\text{data}}}\}$

First data event

Last data event

- Each event X_i contains is defined by a weight w_i and some variables $\vec{x}$:
  $X_i = (w_i, \vec{x}_i)$
- For data, all weights are unity: $w_i = 1$.
- For MC the final weight tells us “the importance of the event”
  - $w_i = 1$ would mean “equally important as a data event”.
  - We want small MC weights (less than 1) to avoid large MC statistical errors
  - MC can have negative weights corresponding to NLO corrections (negative interference)

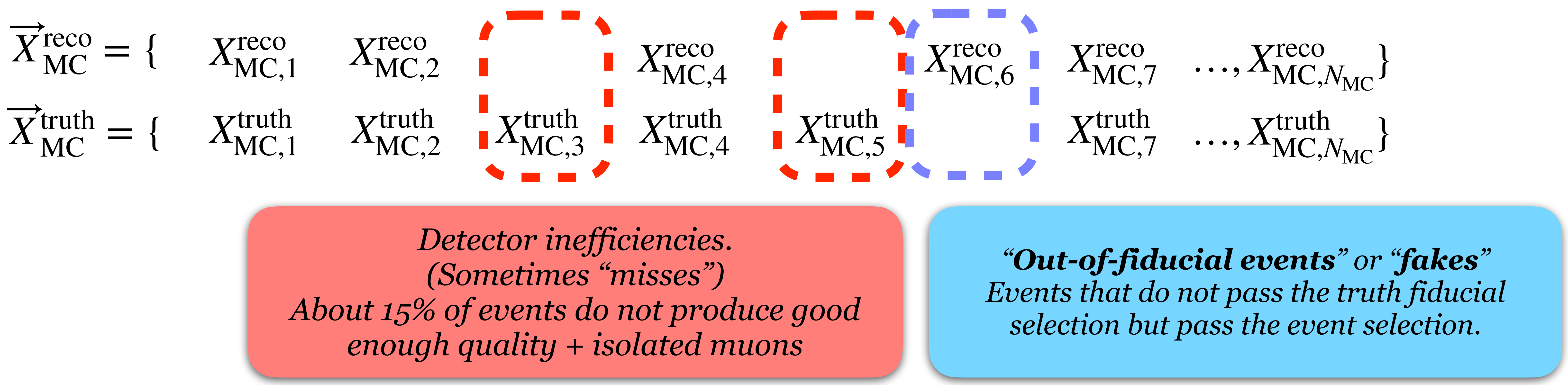
Input and output of the unfolding (2/3)

- When we unfold, we have real data as input and MC simulated input
- For MC, we have both truth and reco information for each event

MC sample:

$$\vec{X}^{\text{reco}}_{\text{MC}} = \{X^{\text{reco}}_{\text{MC},1}, X^{\text{reco}}_{\text{MC},2}, \ldots, X^{\text{reco}}_{\text{MC},N_{\text{MC}}}\}$$
$$\vec{X}^{\text{truth}}_{\text{MC}} = \{X^{\text{truth}}_{\text{MC},1}, X^{\text{truth}}_{\text{MC},2}, \ldots, X^{\text{truth}}_{\text{MC},N_{\text{MC}}}\}$$

- After applying selection, events will be removed:



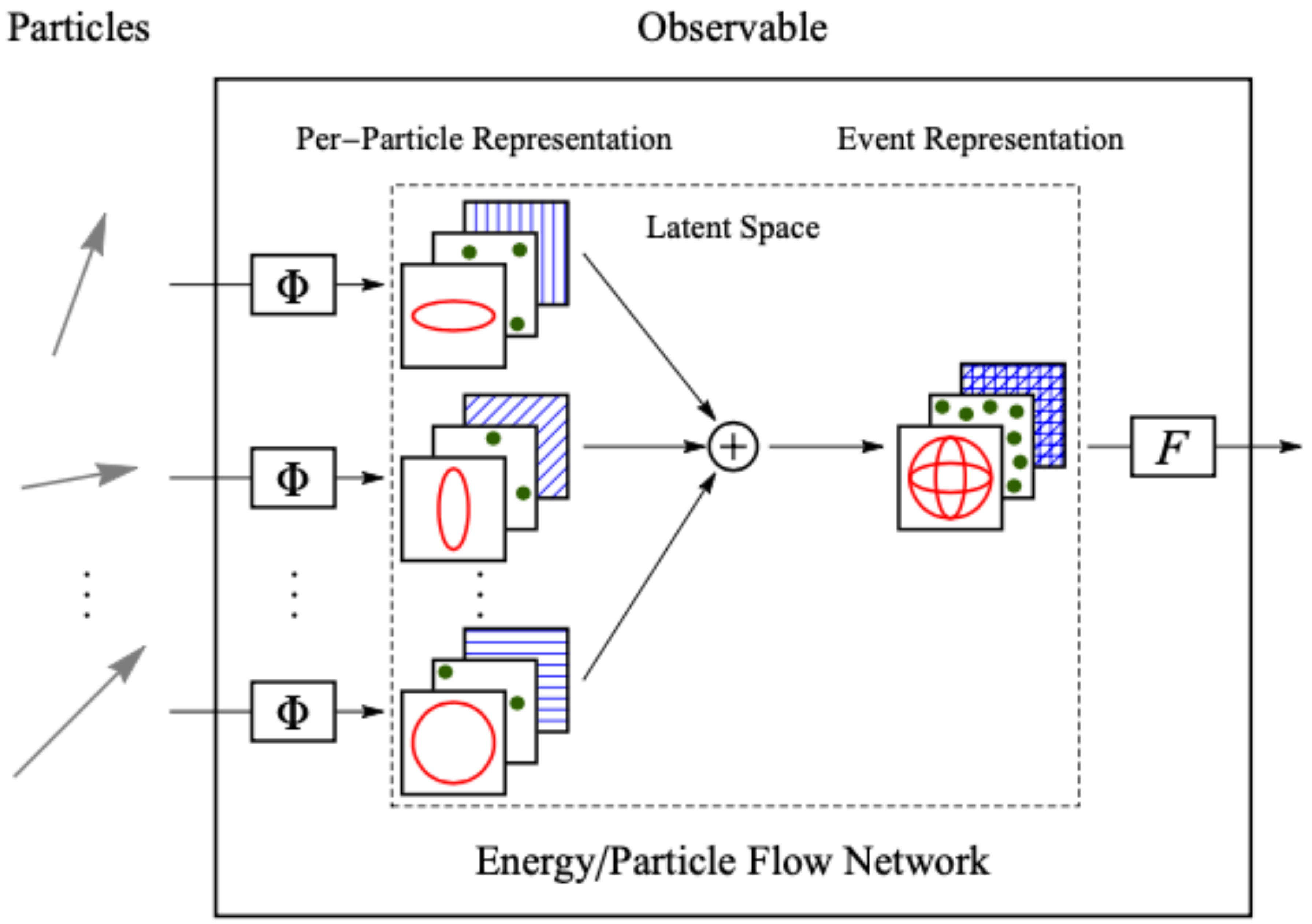
$$\begin{array}{lcccccccc} \vec{X}^{\text{reco}}_{\text{MC}} = \{ & X^{\text{reco}}_{\text{MC},1} & X^{\text{reco}}_{\text{MC},2} & & X^{\text{reco}}_{\text{MC},4} & & X^{\text{reco}}_{\text{MC},6} & X^{\text{reco}}_{\text{MC},7} & \ldots, X^{\text{reco}}_{\text{MC},N_{\text{MC}}}\} \\ \vec{X}^{\text{truth}}_{\text{MC}} = \{ & X^{\text{truth}}_{\text{MC},1} & X^{\text{truth}}_{\text{MC},2} & X^{\text{truth}}_{\text{MC},3} & X^{\text{truth}}_{\text{MC},4} & X^{\text{truth}}_{\text{MC},5} & & X^{\text{truth}}_{\text{MC},7} & \ldots, X^{\text{truth}}_{\text{MC},N_{\text{MC}}}\} \end{array}$$

Detector inefficiencies.
(Sometimes "misses")
About 15% of events do not produce good enough quality + isolated muons

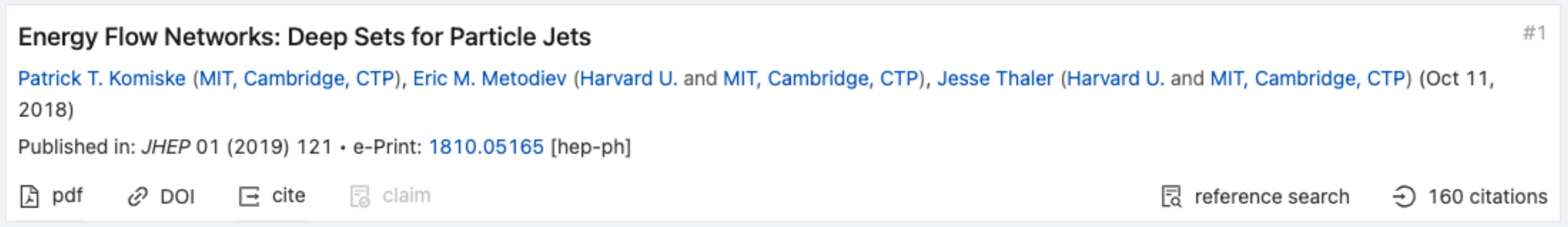
*"**Out-of-fiducial events**" or "**fakes**"*
Events that do not pass the truth fiducial selection but pass the event selection.

*Factorize: **Efficiency correction**, **fiducial correction** + preform unfolding on **(truth+reco) subset***

Particle flow networks



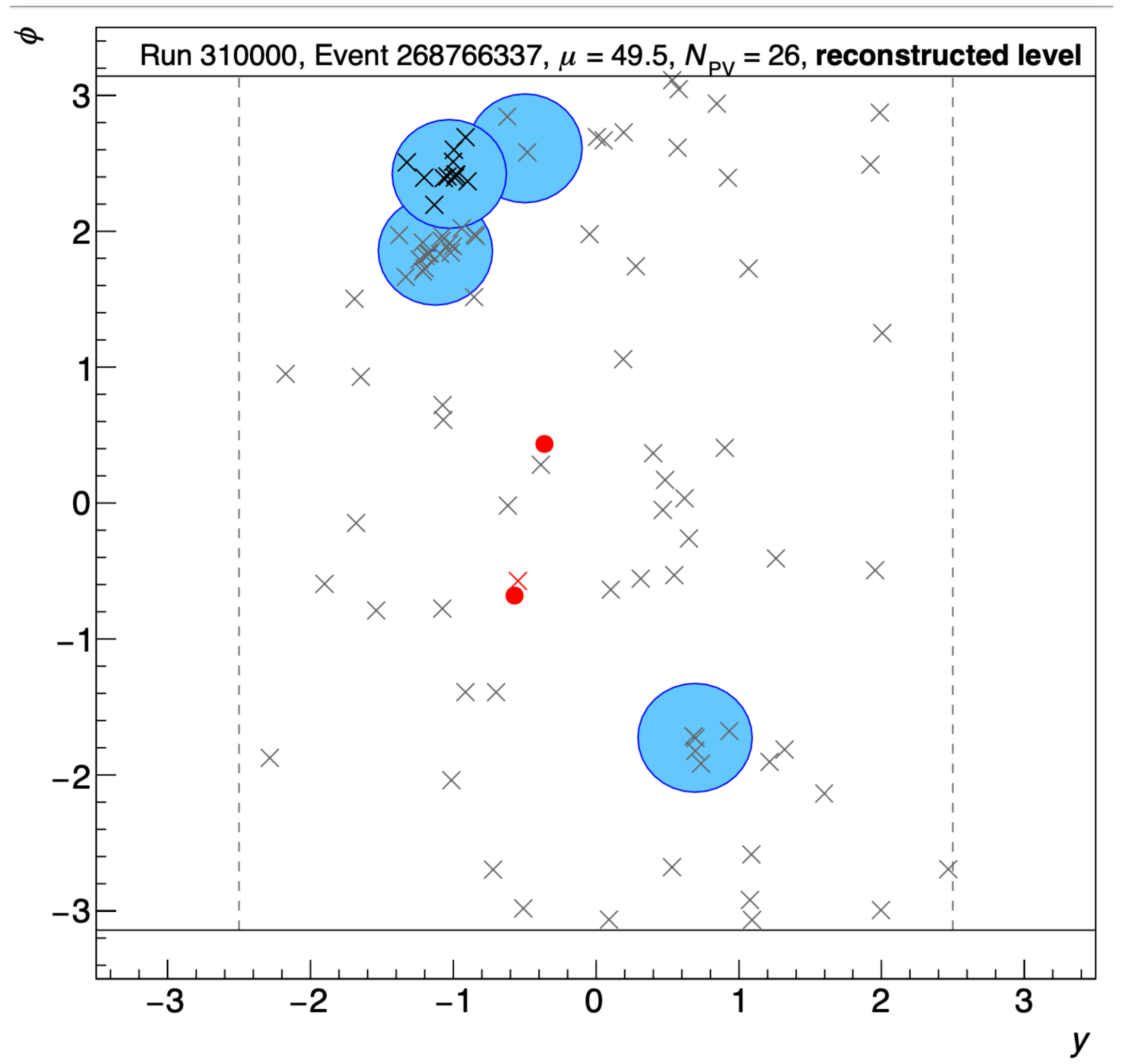
$$\mathcal{O}(\{p_1,\dots,p_M\}) = F\left(\sum_{i=1}^{M}\Phi(p_i)\right)$$

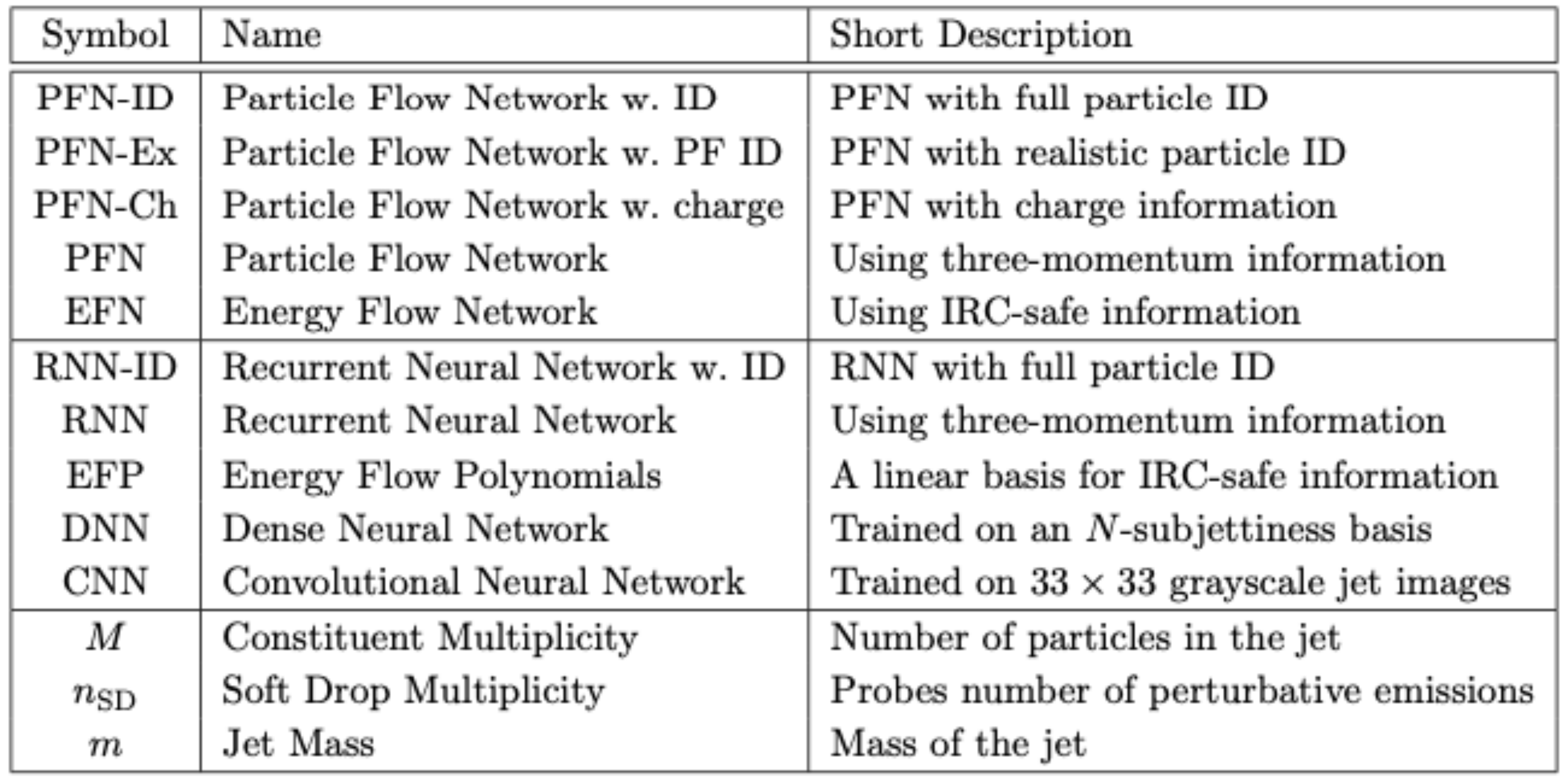


Energy Flow Networks: Deep Sets for Particle Jets

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]





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

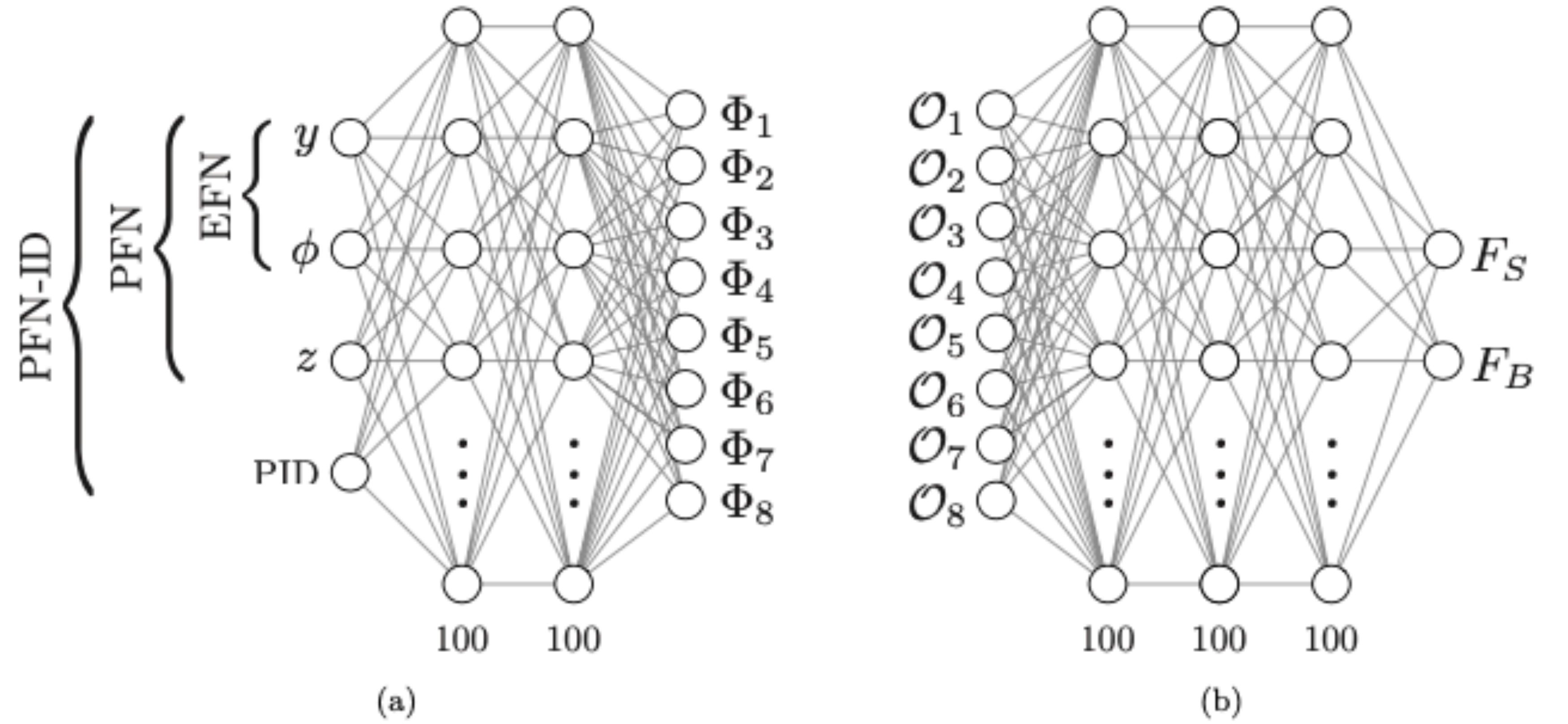


Figure 4: The particular dense networks used here to parametrize (a) the per-particle