MultiFold: 
A user’s perspective

Youqi Song (youqi.song@yale.edu)
2024 RHIC/AGS annual users’ meeting, BNL
ML&AI workshop, 6/11/2024
Outline

• What is MultiFold?
• What are some applications of MultiFold?
• How does MultiFold work?
Outline

• What is MultiFold? → the bare minimum to get started
• What are some applications of MultiFold? → proof that the algorithm works
• How does MultiFold work? → peeking into the black box
Unfolding

- Corrects for detector effects, due to inefficiency, finite resolution, ...
  → Allows for result comparison with theories and other experiments
Unfolding

• **Corrects for detector effects**, due to inefficiency, finite resolution, ...
  → Allows for result comparison with theories and other experiments

• **Ingredients** for unfolding

![Diagram showing detector-level and particle-level data with inputs and outputs](attachment:image.png)
Unfolding

- **Corrects for detector effects**, due to inefficiency, finite resolution, ...
  → Allows for result comparison with theories and other experiments

- **Ingredients** for unfolding

![Diagram showing the process of unfolding with inputs and outputs including Data, PYTHIA+GEANT, Truth, and PYTHIA.]
Unfolding

- Unfolding methods:
  - Iterative Bayesian unfolding \((D'Agostini. \text{arXiv:1010.0632}(2010))\)
  - **MultiFold** \((Andreassen \text{ et al. PRL 124, 182001 (2020)})\)
    - Machine learning driven
    - Unbinned
    - **Simultaneously unfolds many observables**
      \(\rightarrow\) Correlation information is retained!
Unfolding

- Unfolding methods:
  - Iterative Bayesian unfolding (D’Agostini, arXiv:1010.0632 (2010))
  - **MultiFold** (Andreassen et al., PRL 124, 182001 (2020))
    - Machine learning driven
    - Unbinned
    - **Simultaneously unfolds many observables** → Correlation information is retained!
Unfolding

- Unfolding methods:
  - Iterative Bayesian unfolding (D'Agostini. arXiv:1010.0632(2010))
  - **MultiFold** (Andreassen et al. PRL 124, 182001 (2020))
    - Machine learning driven
    - Unbinned
    - **Simultaneously unfolds many observables** → Correlation information is retained!

Unfolding observable 1
observable 2

$p+p \sqrt{s} = 200$ GeV

\[ a < \text{observable 3} < b \]

function of observables 3, 4 and 5

- What is MultiFold
- Applications of MultiFold
- How MultiFold works
- Conclusions
Unfolding

- Unfolding methods:
  - Iterative Bayesian unfolding (D’Agostini. arXiv:1010.0632(2010))
  - MultiFold (Andreassen et al. PRL 124, 182001 (2020))
    - Machine learning driven
    - Unbinned
    - Simultaneously unfolds many observables → Correlation information is retained!

- Variations of the MultiFold/OmniFold algorithm:

<table>
<thead>
<tr>
<th>Variation</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>UniFold</td>
<td>One event observable</td>
</tr>
<tr>
<td>MultiFold</td>
<td>Many event observables</td>
</tr>
<tr>
<td>OmniFold</td>
<td>Full phase space of the event</td>
</tr>
</tbody>
</table>
Unfolding

- Unfolding methods:
  - Iterative Bayesian unfolding (D'Agostini, arXiv:1010.0632(2010))
  - **MultiFold** (Andreassen et al. PRL 124, 182001 (2020))
    - Machine learning driven
    - Unbinned
    - **Simultaneously unfolds many observables** → Correlation information is retained!

- Variations of the MultiFold/OmniFold algorithm:

<table>
<thead>
<tr>
<th>Variation</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>UniFold</td>
<td>One event observable</td>
</tr>
<tr>
<td>MultiFold</td>
<td>Many event observables</td>
</tr>
<tr>
<td>OmniFold</td>
<td>Full phase space of the event</td>
</tr>
</tbody>
</table>

• What is MultiFold
• Applications of MultiFold
• How MultiFold works
• Conclusions
To get started

- `git clone git@github.com:ericmetodiev/OmniFold.git`
  - More updated repo at https://github.com/hep-lbdl/OmniFold

- Run OmniFold Demo.ipynb

- Replace example files with your own trees
Applications

• MultiFold has been applied to several measurements...

  - Measurement of Lepton-Jet Correlation in Deep-Inelastic Scattering with the H1 Detector Using Machine Learning for Unfolding
    - H1 Collaboration • V. Andreev (SPb, Moscow (main)) et al. (Aug 27, 2021)
    - 8 observables unfolded

  - Unbinned deep learning jet substructure measurement in high Q2ep collisions at HERA
    - H1 Collaboration • V. Andreev (Lebedev Inst.) et al. (Mar 23, 2023)
    - Published in: Phys.Lett.B 854 (2023) 125101 • e-Print: 2303.13620 [hep-ex]
    - 6 observables unfolded

  - Measurement of CollinearDrop jet mass and its correlation with SoftDrop groomed jet substructure observables in √s = 200 GeV pp collisions by STAR
    - STAR Collaboration • Youqi Song for the collaboration. (Jul 15, 2023)
    - e-Print: 2307.07718 [hep-ex]
    - 6 observables unfolded

  - Generalized angularities measurements from STAR at √s_{NN} = 200 GeV
    - STAR Collaboration • Ramiay Panig (Rutgers U, Piscataway) for the collaboration. (Mar 20, 2023)
    - Contribution to: Quark Matter 2023 • e-Print: 2405.13901 [nucl-ex]
    - 7 observables unfolded

  - Multidifferential study of identified charged hadron distributions in Z-tagged jets in proton-proton collisions at √s =13 TeV
    - LHCb Collaboration • Roel Aaij (Nikhef, Amsterdam) et al. (Aug 24, 2022)
    - Published in: Phys.Rev.D 108 (2023) L031103 • e-Print: 2208.11691 [hep-ex]
    - 4 observables unfolded

  - A simultaneous unbinned differential cross section measurement of twenty-four Z+jets kinematic observables with the ATLAS detector
    - ATLAS Collaboration • Georges Aad (Marseille, CPPM) et al. (May 30, 2024)
    - e-Print: 2405.20041 [hep-ex]
    - 24 observables unfolded

  - Measurement of event shapes in minimum bias events from pp collisions at 13 TeV
    - The CMS Collaboration
    - 8 observables unfolded

in e+p collisions in HERA

in p+p and heavy-ion collisions at RHIC

in p+p collisions at LHC
Applications

- MultiFold has been applied to several measurements...

  - Measurement of Lepton-Jet Correlation in Deep-Inelastic Scattering with the H1 Detector Using Machine Learning for Unfolding
    - 8 observables unfolded

  - Unbinned deep learning jet substructure measurement in high Q2 ep collisions at HERA
    - 6 observables unfolded

  - Measurement of CollinearDrop jet mass and its correlation with SoftDrop groomed jet substructure observables in $\sqrt{s} = 200$ GeV pp collisions by STAR
    - STAR Collaboration • Youqi Song for the collaboration (Jul 15, 2023)
    - e-Print: 2307.07716 [nucl-ex]
    - 6 observables unfolded

  - Generalized angularities measurements from STAR at $\sqrt{s_{NN}} = 200$ GeV
    - STAR Collaboration • Rammy Panj (Rutgers U, Piscataway) for the collaboration (Mar 20, 2024)
    - Contribution to: Quark Matter 2023 - e-Print: 2405.13901 [nucl-ex]
    - 7 observables unfolded

  - Multidifferential study of identified charged hadron distributions in Z-tagged jets in proton-proton collisions at $\sqrt{s} = 13$ TeV
    - LHCB Collaboration • Roel Aaj (Nikhef, Amsterdam) et al (Aug 24, 2022)
    - 4 observables unfolded

  - A simultaneous unbinned differential cross section measurement of twenty-four $Z$+jets kinematic observables with the ATLAS detector
    - ATLAS Collaboration • Georges Aad (Marseille, CPPM) et al (May 30, 2024)
    - e-Print: 2405.20041 [hep-ex]
    - 24 observables unfolded

  - Measurement of event shapes in minimum bias events from pp collisions at 13 TeV
    - The CMS Collaboration
    - 8 observables unfolded

  - 8 observables unfolded

  - 6 observables unfolded

  - 7 observables unfolded

  - 4 observables unfolded

  - 24 observables unfolded

  - 8 observables unfolded

Applications in $e^+p$ collisions in HERA

Applications in $p+p$ and heavy-ion collisions at RHIC

Applications in $p+p$ collisions at LHC
Applications

• Probing the correlation between perturbative and nonperturbative components within jets at STAR

arxiv: 2307.07718

• Simultaneously correct for:
  • $p_T$: transverse momentum
  • $Q^\kappa = \frac{1}{(p_{T,\text{jet}})^\kappa} \sum_{i \in \text{jet}} q_i \cdot (p_{T,i})^\kappa$
  • $M = |\sum_{i \in \text{jet}} p_i| = \sqrt{E^2 - |\vec{p}|^2}$
  • $R_g$: groomed jet radius
  • $M_g$: groomed jet mass
  • $z_g$: shared momentum fraction

$z_g = \frac{\min(p_{T,1}, p_{T,2})}{p_{T,1} + p_{T,2}} > z_{\text{cut}} (R_g/R_{\text{jet}})^\beta$
Applications

- Probing the correlation between perturbative and nonperturbative components within jets at STAR

- Simultaneously correct for:
  - $p_T$: transverse momentum
  - $Q^\kappa$: groomed jet quark
  - $M_{\text{jet}}$: groomed jet mass
  - $z_g$: shared momentum fraction

Good agreement between MultiFold and IBU verified:

$M = \sqrt{E^2 - \vec{p}^2}$
How does MultiFold work?
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

Detector-level

\[ \left( \frac{1}{N} \frac{dN}{dx} \right) \]

Data

\( d(x) \)

PYTHIA+GEANT

\( g(x) \)

\[ \left( \frac{1}{N} \frac{dN}{dx} \right) \]

\( w_1(x) \)
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

Ok for the binned case
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

\( \approx \frac{f(x)}{1 - f(x)} \)

where \( f(x) \) is a neural network and trained with the binary cross-entropy loss function.

Ok for the binned case

Using Bayes’ Theorem; See derivation in backup
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

\[ \approx \frac{f(x)}{1 - f(x)} \]

where \( f(x) \) is a neural network and trained with the binary cross-entropy loss function.

Ok for the binned case

Using Bayes’ Theorem; See derivation in backup

to distinguish events coming from data vs from PYTHIA+GEANT
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

\[ \approx \frac{f(x)}{1 - f(x)} \]

where \( f(x) \) is a neural network and trained with the binary cross-entropy loss function.

Using Bayes’ Theorem; See derivation in backup.

Ok for the binned case.

Want to distinguish events coming from data vs from PYTHIA+GEANT.

Density estimation → Classification!
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

Ok for the binned case

Using Bayes’ Theorem; See derivation in backup

where \( f(x) \) is a neural network and trained with the binary cross-entropy loss function

to distinguish events coming from data vs from PYTHIA+GEANT

Density estimation → Classification!
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

Using Bayes’ Theorem; See derivation in backup to distinguish events coming from data vs from PYTHIA+GEANT

where \( f(x) \) is a neural network and trained with the binary cross-entropy loss function

Ok for the binned case

Density estimation \( \rightarrow \) Classification!
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

Ok for the binned case

Using Bayes’ Theorem; See derivation in backup

where \( f(x) \) is a neural network and trained with the binary cross-entropy loss function

to distinguish events coming from data vs from PYTHIA+GEANT

\[
\begin{align*}
\hat{x} = \begin{bmatrix}
event \text{ observable 1} \\
\ldots \\
event \text{ observable 6}
\end{bmatrix}
\end{align*}
\]

\[
\begin{align*}
f(\hat{x}) &= \text{probability that event is from data} \\
1 - f(\hat{x}) &= \text{probability event is from GEANT}
\end{align*}
\]

Density estimation → Classification!
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

\[ \approx \frac{f(x)}{1 - f(x)} \]

where \( f(x) \) is a neural network and trained with the binary cross-entropy loss function.

Ok for the binned case

Using Bayes’ Theorem; See derivation in backup

to distinguish events coming from data vs from PYTHIA+GEANT

Density estimation → Classification!

• What is MultiFold
• Applications of MultiFold
• How MultiFold works
• Conclusions

RHIC/AGS Users Meeting, 6/11/2024
Youqi Song (Yale)
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

\( \approx \frac{f(x)}{1 - f(x)} \)

Using Bayes’ Theorem; See derivation in backup to distinguish events coming from data vs from PYTHIA+GEANT

where \( f(x) \) is a neural network and trained with the binary cross-entropy loss function

\[
\text{loss}(f(x)) = - \sum_{x \in \text{data}} \log(f(x)) - \sum_{x \in \text{geant}} \log(1 - f(x))
\]

\( f(\hat{x}) = \text{probability that event is from data} \)

\( 1 - f(\hat{x}) = \text{probability event is from GEANT} \)

Density estimation → Classification!
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

Ok for the binned case

Using Bayes’ Theorem; See derivation in backup

to distinguish events coming from data vs from PYTHIA+GEANT

Where \( f(x) \) is a neural network and trained with the binary cross-entropy loss function

\[
\text{loss}(f(x)) = - \sum_{x \in \text{data}} \log(f(x)) - \sum_{x \in \text{geant}} \log(1 - f(x))
\]

Density estimation \( \rightarrow \) Classification!

- What is MultiFold
- Applications of MultiFold
- How MultiFold works
- Conclusions

RHIC/AGS Users Meeting, 6/11/2024

Youqi Song (Yale)
Iterative reweighting: Step 1, iteration 1

Weights: \( w_1(x) = \frac{d(x)}{g(x)} \)

where \( f(x) \) is a neural network and trained with the binary cross-entropy loss function

\[
\text{loss}(f(x)) = - \sum_{x \in \text{data}} \log(f(x)) - \sum_{x \in \text{geant}} \log(1 - f(x))
\]

Sanity checks:
- Correct classification minimizes the loss function
- Unlikely events get weighted down

Density estimation \( \rightarrow \) Classification!
Iterative reweighting: Step 1, iteration 1

Detector-level

\[
\frac{1}{N} \frac{dN}{dx} = d(x)
\]

\[
\frac{1}{N} \frac{dN}{dx} = g(x)
\]

PYTHIA+GEANT, reweighted

PYTHIA+GEANT

\[
w_1(x)\]

\[
\frac{1}{N} \frac{dN}{dx}
\]
Iterative reweighting: Step 1, iteration 1

Detector-level

\[
\frac{1}{N} \frac{dN}{dx} \quad d(x)
\]

\[
\frac{1}{N} \frac{dN}{dx} \quad g(x)
\]

\[
\text{event 1} = x_1
\]
\[
\text{event 2}
\]
\[
... 
\]
\[
\text{event N}
\]

\[
\text{event 1 observable 1}
\]
\[
... 
\]
\[
\text{event 1 observable 6}
\]

\[
\text{PYTHIA+GEANT, reweighted}
\]

\[
\frac{1}{N} \frac{dN}{dx} \quad x
\]

\[
\text{PYTHIA+GEANT, reweighted}
\]

\[
\text{event 1, } x_1 \times w_1(x_1)
\]
\[
\text{event 2, } x_2 \times w_1(x_2)
\]
\[
... 
\]
\[
\text{event N, } x_N \times w_1(x_N)
\]

• What is MultiFold
• Applications of MultiFold
• How MultiFold works
• Conclusions
Iterative reweighting: Step 1, iteration 1

Detector-level

\[ \frac{1}{N} \frac{dN}{dx} \]

Data \( d(x) \)

\[ w_1(x) \]

\[ g(x) \]

event 1 = \( x_1 \)

event 2

...

event N

\[ \text{event 1 observable 1} \]

\[ \text{event 1 observable 2} \]

\[ \text{event 1 observable 6} \]

\[ \text{event 2, } x_1 w_1(\tilde{x}_1) \]

\[ \text{event 2, } x_2 w_1(\tilde{x}_2) \]

...

\[ \text{event N, } x_N w_1(\tilde{x}_N) \]

PYTHIA+GEANT, reweighted
Iterative reweighting: Step 1, iteration 1

Detector-level

\[ \frac{1}{N} dN/dx \]

Data \( d(x) \)

\[ x \]

\[ w_1(x) \]

PYTHIA+GEANT \( g(x) \)

\[ x \]

event 1 = \( x_1 \)

event 2

...

event N

{ event 1 observable 1
  ...
  event 1 observable 6 }

reweight

PYTHIA+GEANT, reweighted

PYTHIA+GEANT, reweighted

event matching

PYTHIA+GEANT, reweighted

PYTHIA, reweighted

\[ \begin{array}{l}
\text{event } 1, \times w_1(\hat{x}_1) \\
\text{event } 2, \times w_1(\hat{x}_2) \\
\text{...} \\
\text{event } N, \times w_1(\hat{x}_N) \\
\end{array} \]

\[ \begin{array}{l}
\text{event } 1, \times w_1(\hat{x}_1) \\
\text{event } 2, \times w_1(\hat{x}_2) \\
\text{...} \\
\text{event } N, \times w_1(\hat{x}_N) \\
\end{array} \]
Iterative reweighting: Step 2, iteration 1

- Detector response is stochastic
  - Two identical \textbf{particle-level} events might not get mapped to identical \textbf{detector-level} events

<table>
<thead>
<tr>
<th>PYTHIA+GEANT, reweighted</th>
<th>PYTHIA, reweighted</th>
</tr>
</thead>
<tbody>
<tr>
<td>event 1, $\times w_1(x_1)$</td>
<td>event 1, $\times w_1(x_1)$</td>
</tr>
<tr>
<td>event 2, $\times w_1(x_2)$</td>
<td>event 2, $\times w_1(x_2)$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>event $N$, $\times w_1(x_N)$</td>
<td>event $N$, $\times w_1(x_N)$</td>
</tr>
</tbody>
</table>
Iterative reweighting: Step 2, iteration 1

- Detector response is stochastic
  - Two identical particle-level events might not get mapped to identical detector-level events

- $w_1(x)$ is a weighting function of detector-level events
- Want $v_1(y)$, a weighting function of particle-level events

<table>
<thead>
<tr>
<th>PYTHIA+GEANT, reweighted</th>
<th>PYTHIA, reweighted</th>
</tr>
</thead>
<tbody>
<tr>
<td>event 1, $\times w_1(x_1)$</td>
<td>event 1, $\times w_1(x_1)$</td>
</tr>
<tr>
<td>event 2, $\times w_1(x_2)$</td>
<td>event 2, $\times w_1(x_2)$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>event N, $\times w_1(x_N)$</td>
<td>event N, $\times w_1(x_N)$</td>
</tr>
</tbody>
</table>
Iterative reweighting: Step 2, iteration 1

- Detector response is stochastic
  - Two identical particle-level events might not get mapped to identical detector-level events

- $w_1(x)$ is a weighting function of detector-level events
- Want $v_1(y)$, a weighting function of particle-level events

$v_1(y)$

<table>
<thead>
<tr>
<th>PYTHIA</th>
<th>PYTHIA+GEANT, reweighted</th>
<th>PYTHIA, reweighted</th>
</tr>
</thead>
<tbody>
<tr>
<td>event 1 = $\tilde{y}_1$</td>
<td>event 1, $\times w_1(\tilde{x}_1)$</td>
<td>event 1, $\times w_1(\tilde{x}_1)$</td>
</tr>
<tr>
<td>event 2</td>
<td>event 2, $\times w_1(\tilde{x}_2)$</td>
<td>event 2, $\times w_1(\tilde{x}_2)$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>event N</td>
<td>event N, $\times w_1(\tilde{x}_N)$</td>
<td>event N, $\times w_1(\tilde{x}_N)$</td>
</tr>
</tbody>
</table>

What is MultiFold
- Applications of MultiFold
- How MultiFold works
- Conclusions
Iterative reweighting: Step 2, iteration 1

- Detector response is stochastic
  - Two identical particle-level events might not get mapped to identical detector-level events

- \( w_1(x) \) is a weighting function of detector-level events
- Want \( v_1(y) \), a weighting function of particle-level events
  \[ \approx h(y)/(1 - h(y)) \]
  where \( h(y) \) is a neural network and trained with the binary cross-entropy loss function

\[ v_1(y) \]

PYTHIA

<table>
<thead>
<tr>
<th>event 1 = ( \hat{y}_1 )</th>
<th>PYTHIA+GEANT, reweighted</th>
<th>PYTHIA, reweighted</th>
</tr>
</thead>
<tbody>
<tr>
<td>event 1, ( \times w_1(\hat{x}_1) )</td>
<td>event 1, ( \times w_1(\hat{x}_1) )</td>
<td></td>
</tr>
<tr>
<td>event 2, ( \times w_1(\hat{x}_2) )</td>
<td>event 2, ( \times w_1(\hat{x}_2) )</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>event N, ( \times w_1(\hat{x}_N) )</td>
<td>event N, ( \times w_1(\hat{x}_N) )</td>
<td></td>
</tr>
</tbody>
</table>

\( w_1(x) \) is a weighting function of detector-level events

\( v_1(y) \) is a weighting function of particle-level events

\( h(y) \) is a neural network and trained with the binary cross-entropy loss function
Iterative reweighting: Step 2, iteration 1

PYTHIA, w/ weights from step 1

<table>
<thead>
<tr>
<th>event 1, × w₁((x_1))</th>
<th>event 2, × w₁((x_2))</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>event N, × w₁((x_N))</td>
<td></td>
</tr>
</tbody>
</table>

PYTHIA

<table>
<thead>
<tr>
<th>event 1 = (\hat{y}_1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>event 2</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>event N</td>
</tr>
</tbody>
</table>

\(v_1(y)\) →

PYTHIA, w/ proper weighting function

| event 1, × \(v_1(\hat{y}_1)\) |
| event 2, × \(v_1(\hat{y}_2)\) |
| ...                            |
| event N, × \(v_1(\hat{y}_N)\) |

Unfolding result after 1 iteration

• What is MultiFold
• Applications of MultiFold
• How MultiFold works
• Conclusions
Iterative reweighting: Step 2, iteration 1

**PYTHIA, w/ weights from step 1**

- $\text{event 1, } x_1$ $\times$ $w_1(x_1)$
- $\text{event 2, } x_2$ $\times$ $w_1(x_2)$
- $\ldots$
- $\ldots$
- $\text{event N, } x_N$ $\times$ $w_1(x_N)$

**PYTHIA, w/ proper weighting function**

- $\text{event 1, } y_1$ $\times$ $v_1(y_1)$
- $\text{event 2, } y_2$ $\times$ $v_1(y_2)$
- $\ldots$
- $\ldots$
- $\text{event N, } y_N$ $\times$ $v_1(y_N)$

Unfolding result after 1 iteration

- $v_1(y)$ used to reweight both particle and detector-level events in iteration 2
- $v_1(y)$, $w_2(x)$, $v_2(y)$, ...

**What is MultiFold**
- Applications of MultiFold
- How MultiFold works
- Conclusions
Iterative reweighting: Result

- Result: **Particle-level** events, reweighted by $v_N(y)$ – step 2 output of the last iteration
Iterative reweighting: Result

- **Result:** Particle-level events, reweighted by $v_N(y)$ – step 2 output of the last iteration
- **Unfolding methods:**
  - Iterative Bayesian unfolding (D'Agostini. arXiv:1010.0632(2010))
  - **MultiFold** (Andreassen et al. PRL 124, 182001 (2020))
    - Machine learning driven
    - Unbinned
    - *Simultaneously unfolds many observables*
Iterative reweighting: Result

• Result: **Particle-level** events, reweighted by $\nu_N(y)$ – step 2 output of the last iteration

• Unfolding methods:
  • Iterative Bayesian unfolding *(D’Agostini. arXiv:1010.0632(2010))*
  • **MultiFold** *(Andreassen et al. PRL 124, 182001 (2020))*
    ✓ Machine learning driven
    • Unbinned
    • **Simultaneously unfolds many observables**
Iterative reweighting: Result

- Result: **Particle-level** events, reweighted by $v_N(y)$ – step 2 output of the last iteration
- Unfolding methods:
  - Iterative Bayesian unfolding ([D'Agostini. arXiv:1010.0632(2010)])
  - **MultiFold** ([Andreassen et al. PRL 124, 182001 (2020)])
    - ✓ Machine learning driven
    - ✓ Unbinned $\rightarrow$ reweighting is done event-by-event
  - Simultaneously unfolds many observables
Iterative reweighting: Result

• Result: **Particle-level** events, reweighted by $v_N(y)$ – step 2 output of the last iteration
• Unfolding methods:
  • Iterative Bayesian unfolding (D'Agostini. arXiv:1010.0632(2010))
  • **MultiFold** (Andreassen et al. PRL 124, 182001 (2020))
    ✓ Machine learning driven
    ✓ Unbinned $\rightarrow$ reweighting is done event-by-event
    ✓ **Simultaneously unfolds many observables** $\rightarrow$ can adjust the input dimension of neural networks
Iterative reweighting: Result

• Result: **Particle-level** events, reweighted by $v_N(y)$ – step 2 output of the last iteration
• Unfolding methods:
  • Iterative Bayesian unfolding (D'Agostini. arXiv:1010.0632(2010))
  • **MultiFold** (Andreassen et al. PRL 124, 182001 (2020))
    ✓ Machine learning driven
    ✓ Unbinned $\rightarrow$ reweighting is done event-by-event
    ✓ **Simultaneously unfolds many observables** $\rightarrow$ can adjust the input dimension of neural networks

**Appendix: OMNIFold as a Maximum Likelihood Estimate**

In this Appendix, we review the statistical underpinnings of Iterative Bayesian Unfolding (IBU) [6] as well as OMNIFold and confirm that they converge to the maximum likelihood estimate of the true particle-level distribution.
Iterative reweighting: Result

- Result: **Particle-level** events, reweighted by $v_N(y)$ – step 2 output of the last iteration
- Unfolding methods:
  - Iterative Bayesian unfolding (D'Agostini. arXiv:1010.0632(2010))
  - **MultiFold** (Andreassen et al. PRL 124, 182001 (2020))
    - ✓ Machine learning driven
    - ✓ Unbinned $\rightarrow$ reweighting is done event-by-event
    - ✓ **Simultaneously unfolds many observables** $\rightarrow$ can adjust the input dimension of neural networks

Good agreement between MultiFold and RooUnfold verified with data.

Arxiv: 2307.07718
Conclusions

• **MultiFold** *(Andreassen et al. PRL 124, 182001 (2020))*

  ✓ Machine learning driven
  ✓ Unbinned → reweighting is done event-by-event
  ✓ **Simultaneously unfolds many observables** → can adjust the input dimension of neural networks
Conclusions

• **MultiFold** (Andreassen et al. PRL 124, 182001 (2020))
  ✓ Machine learning driven
  ✓ Unbinned → reweighting is done event-by-event
  ✓ **Simultaneously unfolds many observables** → can adjust the input dimension of neural networks

• Resources readily available, e.g., https://github.com/ericmetodiev/OmniFold and https://github.com/hep-lbdl/OmniFold
• Successful applications in H1, STAR, LHCb, ATLAS and CMS
• Easy access to correlation information among observables
• Promising potential for multi-differential measurements
Backup
Applications

- Probing transverse-momentum dependent (TMD) parton distribution functions at H1

\[ \text{Jet } p_T \cdot \text{Jet } \eta \cdot \text{Jet } \phi \cdot \text{Electron } p_T \cdot \text{Electron } p_z \cdot \text{Electron-jet imbalance} \cdot \text{Electron-jet azimuthal angle correlation} \]

- Probing TMD jet fragmentation functions at LHCb

\[ \text{Jet } p_T \cdot \text{Jet } \eta \cdot \text{Hadron in jet longitudinal momentum fraction} \cdot \text{Hadron momentum wrt jet axis} \]

Simultaneously correct for:
- Jet \( p_T \) • Jet \( \eta \) • Jet \( \phi \) • Electron \( p_T \) • Electron \( p_z \) • Electron-jet imbalance • Electron-jet azimuthal angle correlation
- Similar measurement ongoing at STAR, see talk by Hannah Harrison-Smith

Andreev et al. PRL 128, 132002 (2022)
Youqi Song (Yale)
Iterative reweighting: Step 1, iteration 1

\[ w_1(x) = \frac{d(x)}{g(x)} \]

Using Bayes’ Theorem

\[ P(A|B) = \frac{P(B|A)P(A)}{P(B)} \]


\[ \approx \frac{f(x)}{1 - f(x)} \]

where \( f(x) \) is a neural network and trained with the binary cross-entropy loss function.
Iterative reweighting: Step 1, iteration 2

Data

- event 1
- event 2
- ...
- ...
- event N

PYTHIA+GEANT, w/ weights from iteration 1

- event 1, × \( v_1(y_1) \)
- event 2, × \( v_1(y_2) \)
- ...
- ...
- event N, × \( v_1(y_N) \)

PYTHIA+GEANT, reweighted to data

- event 1, × \( w_2(x_1) \)
- event 2, × \( w_2(x_2) \)
- ...
- ...
- event N, × \( w_2(x_N) \)
Iterative reweighting: Step 2, iteration 2

**PYTHIA, w/ weights from step 1, iteration 2**
- event 1, \(w_2(x_1)\)
- event 2, \(w_2(x_2)\)
  ...
- ...
- event N, \(w_2(x_N)\)

\[ v_2(y) \]

**PYTHIA, w/ proper weighting function**
- event 1, \(v_2(y_1)\)
- event 2, \(v_2(y_2)\)
  ...
- ...
- event N, \(v_2(y_N)\)

Unfolding result after 2 iterations
Iterative reweighting: Iteration n

- Iteration n, Step 1:
  - Run model.fit() on Data. Obtain weights.
  - Weights corresponding to Embedding are pulled to step 2.
  - Decides when training should stop.

- Iteration n, Step 2:
  - Run model.fit() on Pythia*. Obtain weights.
  - Weights corresponding to Embedding are pushed to step 1.
  - Decides when training should stop.

*: (With weights pulled from step 1 of iteration n)
**: (With weights pushed from step 2 of iteration (n-1))
Iterative reweighting: Toy example

- Adapted from slides by Ben Nachman

**Initial**

- Simulation
- Generation

<table>
<thead>
<tr>
<th>Generation</th>
<th>Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0% 50%</td>
</tr>
<tr>
<td>2</td>
<td>100% 50%</td>
</tr>
</tbody>
</table>

**Ideal**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measured</td>
<td>0%</td>
<td>50%</td>
</tr>
<tr>
<td></td>
<td>100%</td>
<td>50%</td>
</tr>
</tbody>
</table>

**Starting point of iteration 1**

- Sim: 2
- Gen: 1

**Result from iteration 1**

- Sim: 2
- Gen: 1

**New weights**

- $w_i = \frac{2}{3}$
- $w_i = \frac{4}{3}$

**To match data distribution**

- Sim: 2

**To match 50-50 from response matrix**

- Sim: 2

**Next iteration**

- Sim: 2
- Gen: 1

**Applications of MultiFold**

- What is MultiFold
- How MultiFold works
- Conclusions
Iterative reweighting: Toy example

- Adapted from [slides](#) by Ben Nachman

### Initial

<table>
<thead>
<tr>
<th>Generation</th>
<th>Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0% 50%</td>
</tr>
<tr>
<td>2</td>
<td>100% 50%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ideal</th>
<th>Measured</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>100% 50%</td>
</tr>
</tbody>
</table>

### Starting point of iteration 2

<table>
<thead>
<tr>
<th>Sim.</th>
<th>Gen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4/5</td>
<td>4/3</td>
</tr>
</tbody>
</table>

### Result from iteration 2

<table>
<thead>
<tr>
<th>Sim.</th>
<th>Gen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3/2</td>
<td>3/2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sim.</th>
<th>Gen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3/4</td>
<td>3/4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sim.</th>
<th>Gen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3/4</td>
<td>3/4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sim.</th>
<th>Gen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3/4</td>
<td>3/4</td>
</tr>
</tbody>
</table>

### How MultiFold works

- Iterative reweighting
- Response matrix
- Data distribution

### Conclusions
Iterative reweighting: Toy example

- Adapted from [slides](#) by Ben Nachman

Initial

Result from iteration 1

Result from iteration 2

Result from iteration ∞
Iterative reweighting

- Why do we iterate?

- What is MultiFold
- Applications of MultiFold
- How MultiFold works
- Conclusions
Challenges

- Computationally expensive
- How to publish an unbinned result? arxiv:2109.13243