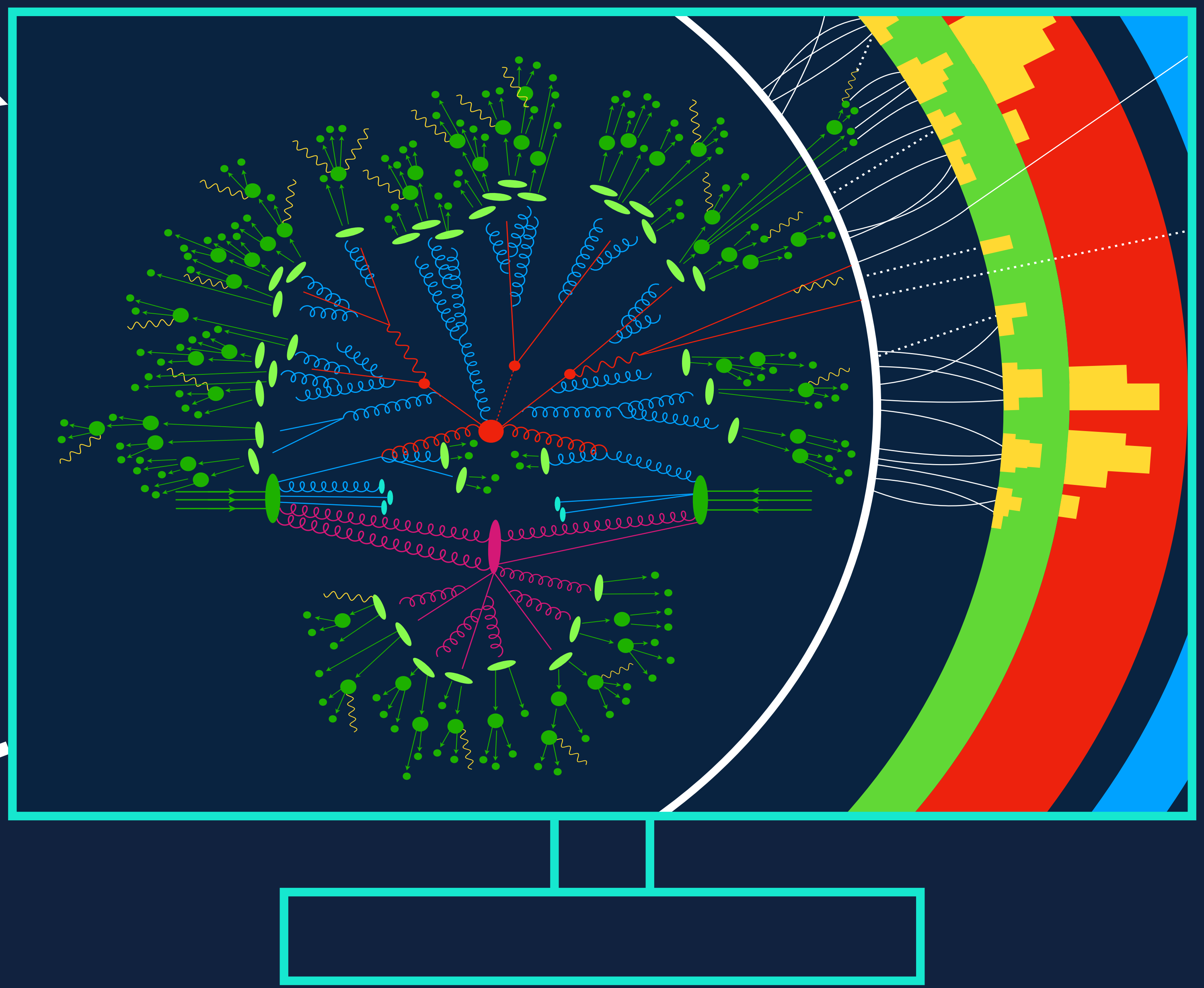
Modern Machine Learning for the LHC Simulation Chain

$$
\begin{aligned}
\mathscr{L} = &-\frac{1}{4}F^{\mu\nu}F_{\mu\nu} \\
&+ i\bar{\psi}\gamma^{\mu}D_{\mu}\psi \\
&+ \bar{\psi}^{i}_{L} y_{ij} \Phi \psi^{j}_{R} + \text{h.c.} \\
&+ |D_{\mu}\Phi|^{2} + V(\Phi) + \text{BSM}
\end{aligned}
$$



UCLouvain

ML4Jets — Hamburg 2023

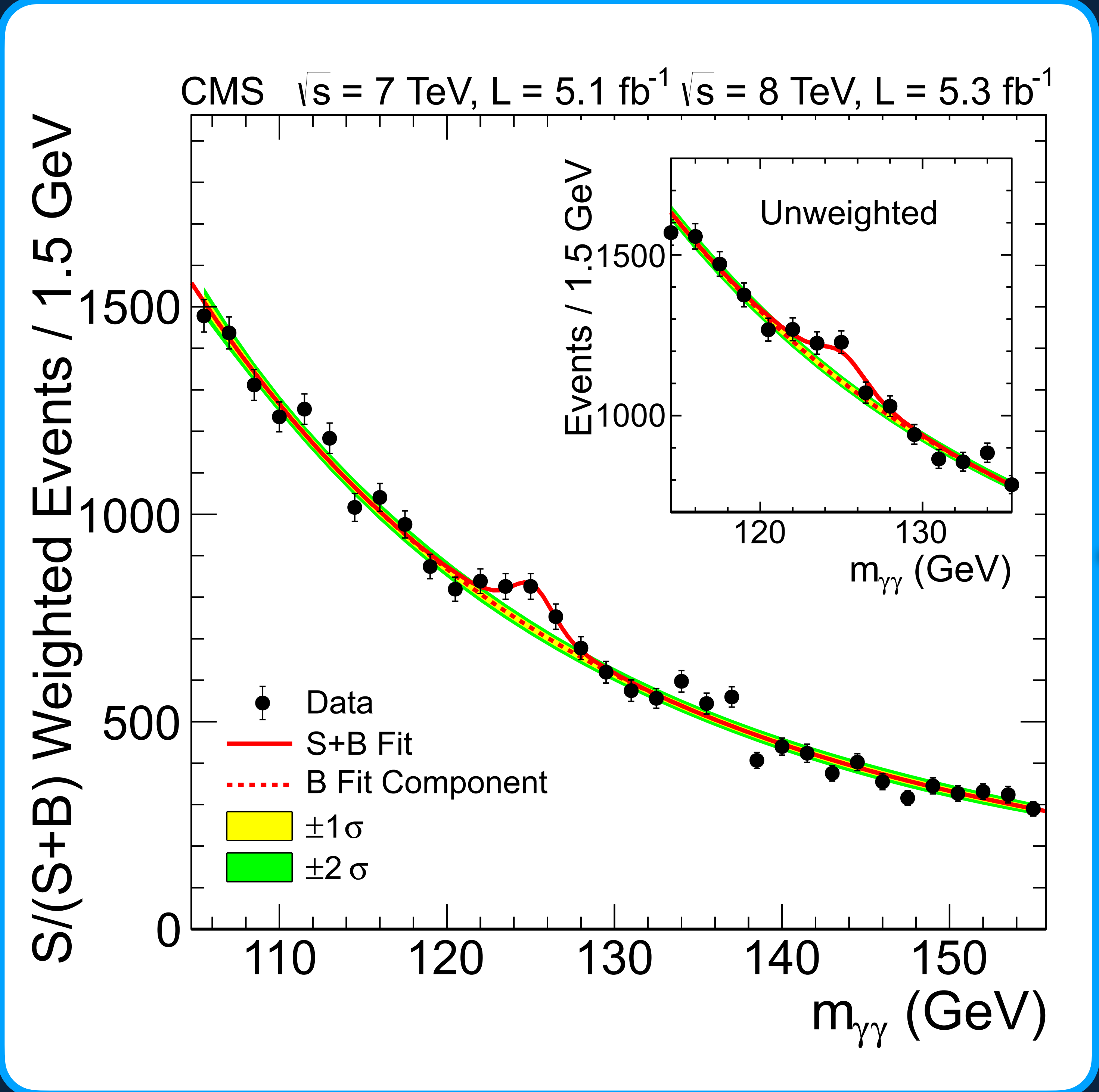
Ramon Winterhalder — UCLouvain

Why do we talk about simulations?

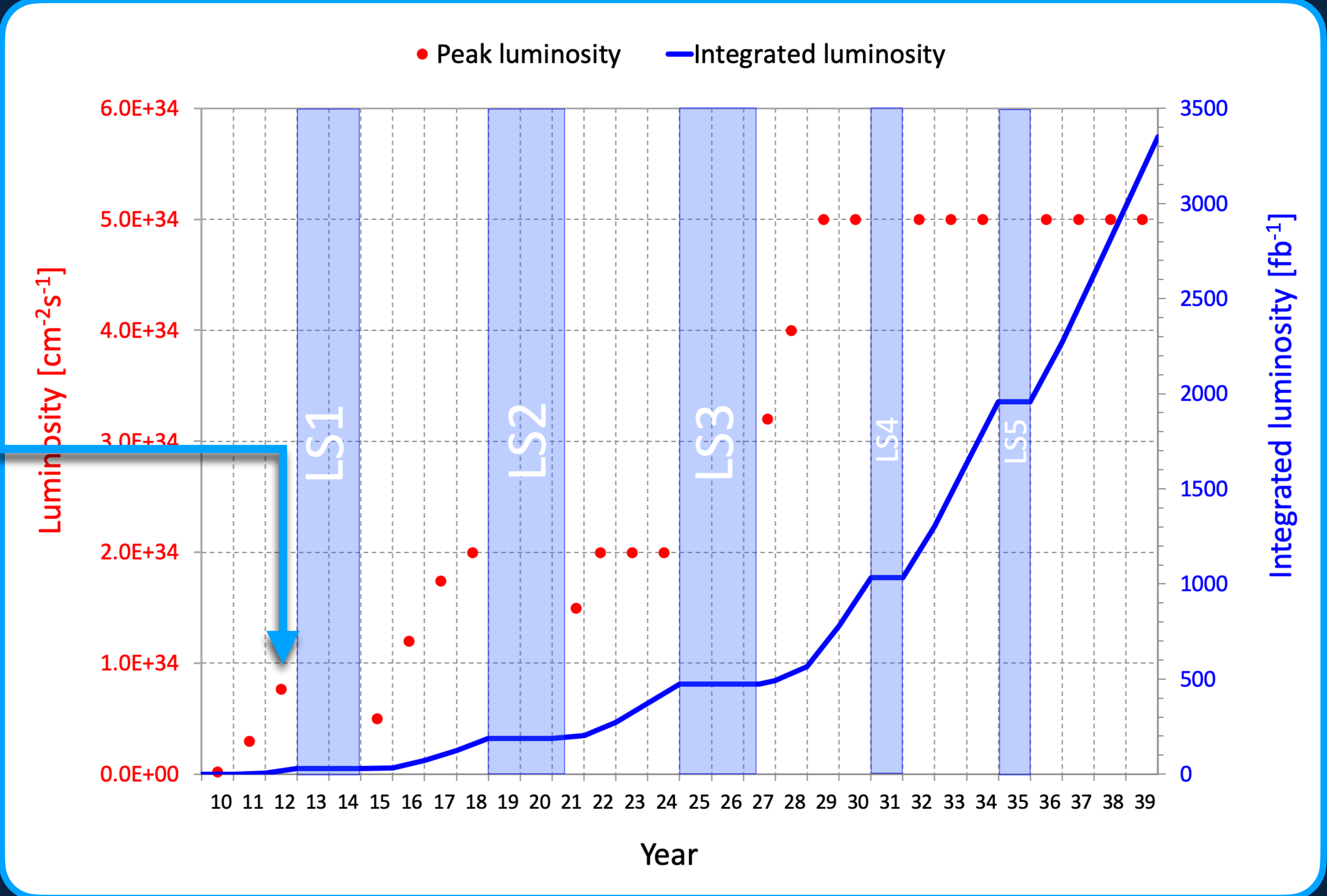
We will have a lot more data soon

CMS Collaboration [arXiv:1207.7235, Phys.Lett.B]

https://lhc-commissioning.web.cern.ch/schedule/HL-LHC-plots.htm



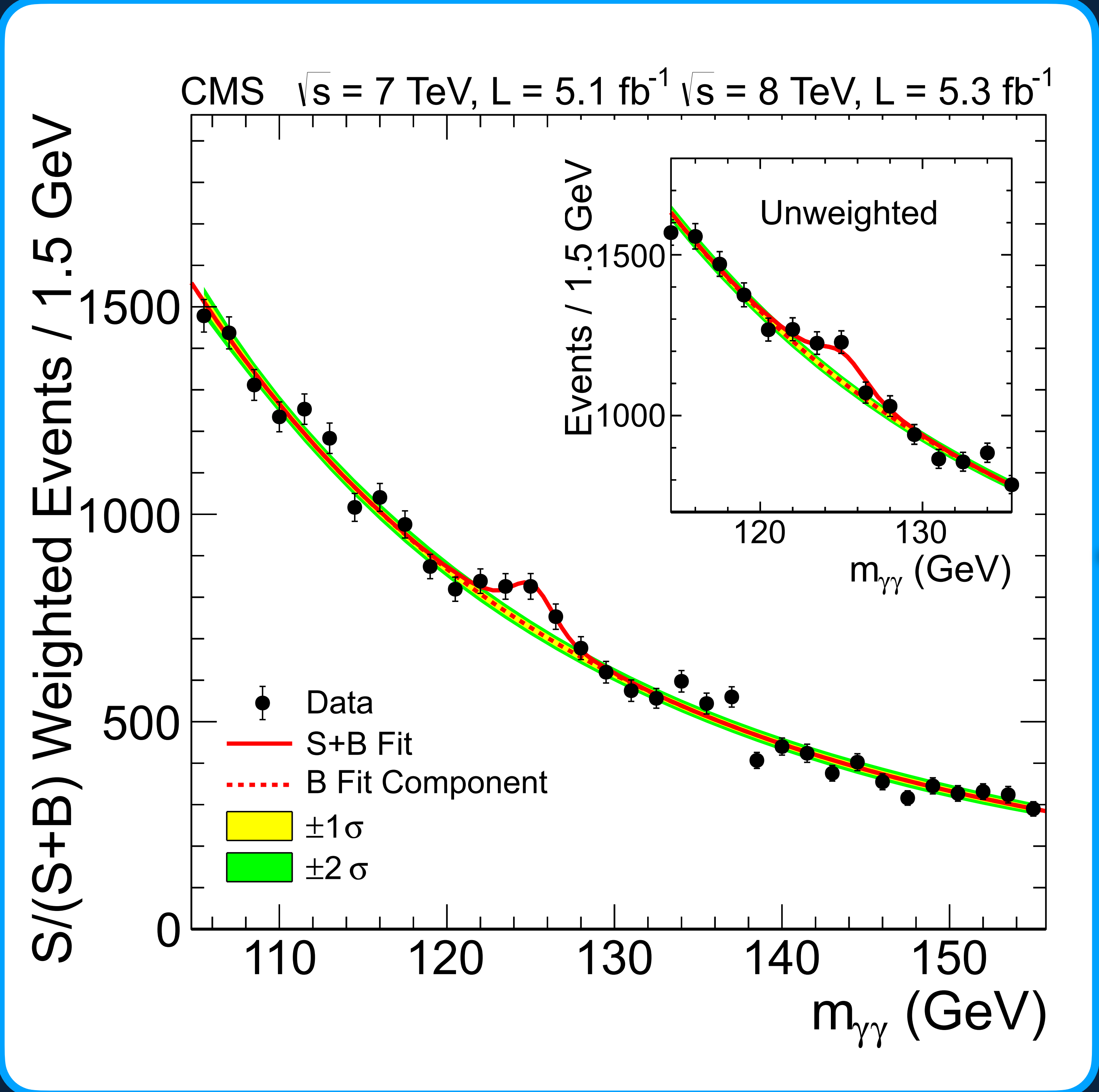
2012



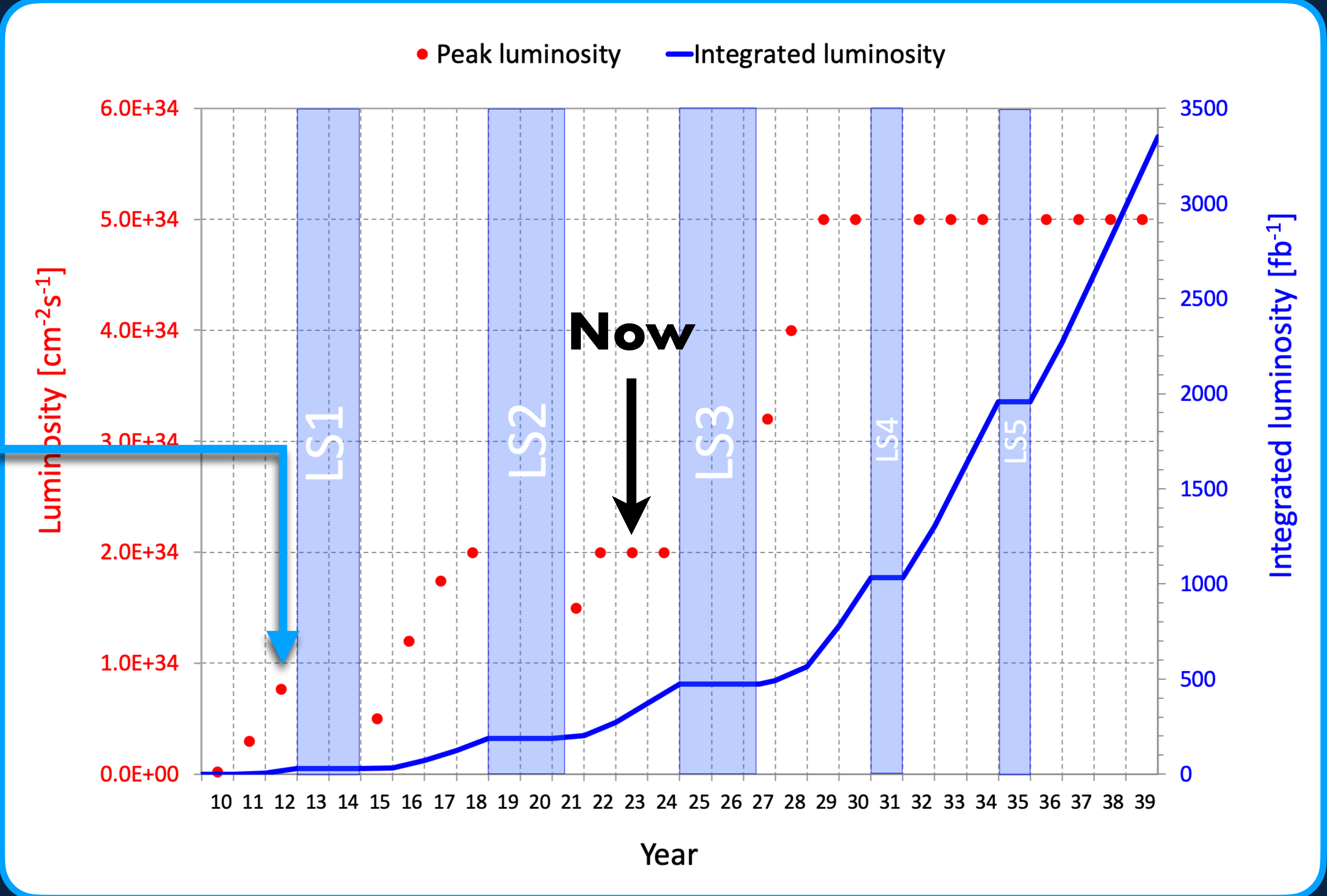
We will have a lot more data soon

CMS Collaboration [arXiv:1207.7235, Phys.Lett.B]

https://lhc-commissioning.web.cern.ch/schedule/HL-LHC-plots.htm

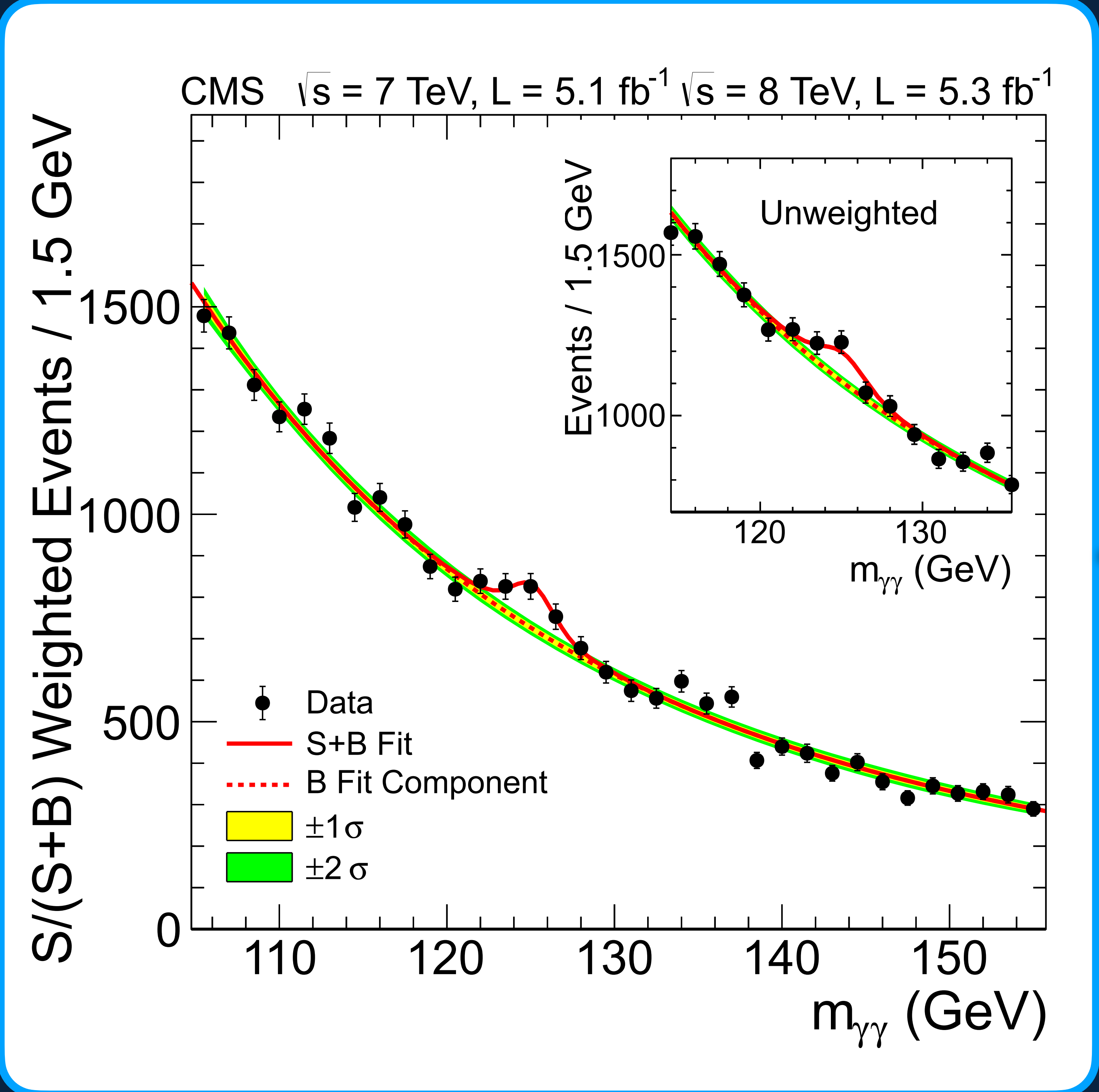


2012

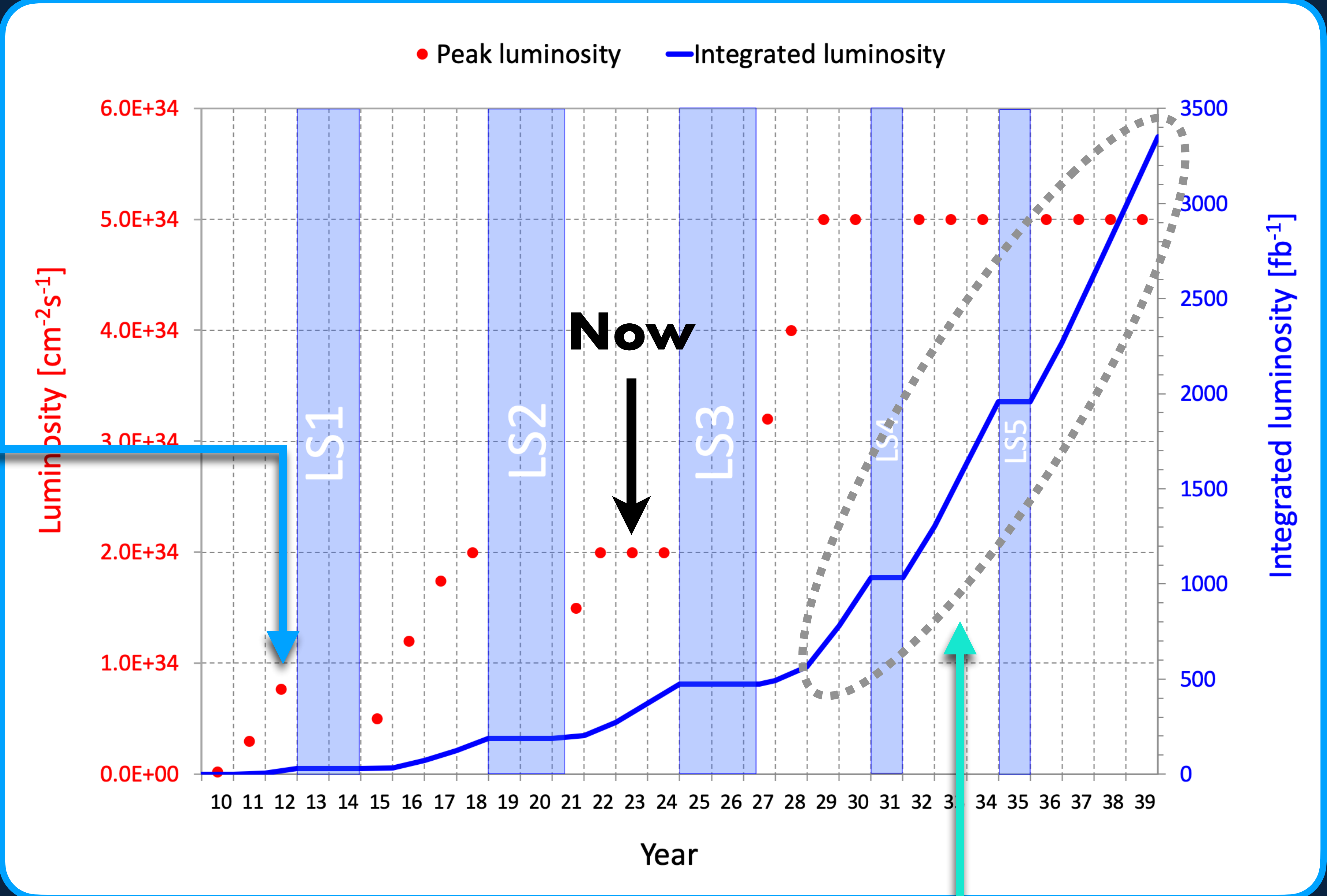


We will have a lot more data soon

CMS Collaboration [arXiv:1207.7235, Phys.Lett.B]



https://lhc-commissioning.web.cern.ch/schedule/HL-LHC-plots.htm



2012

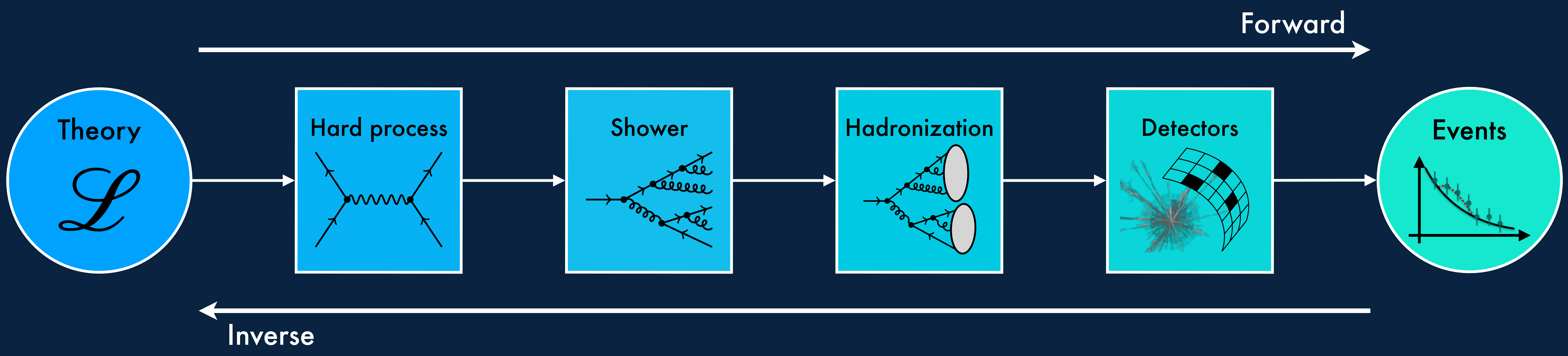
- We will have **20-25x** more data

➡ We want to understand **all aspects** of data based on **first principles!**

Understanding LHC data based on 1^{st} principles

What do we need to understand the data?

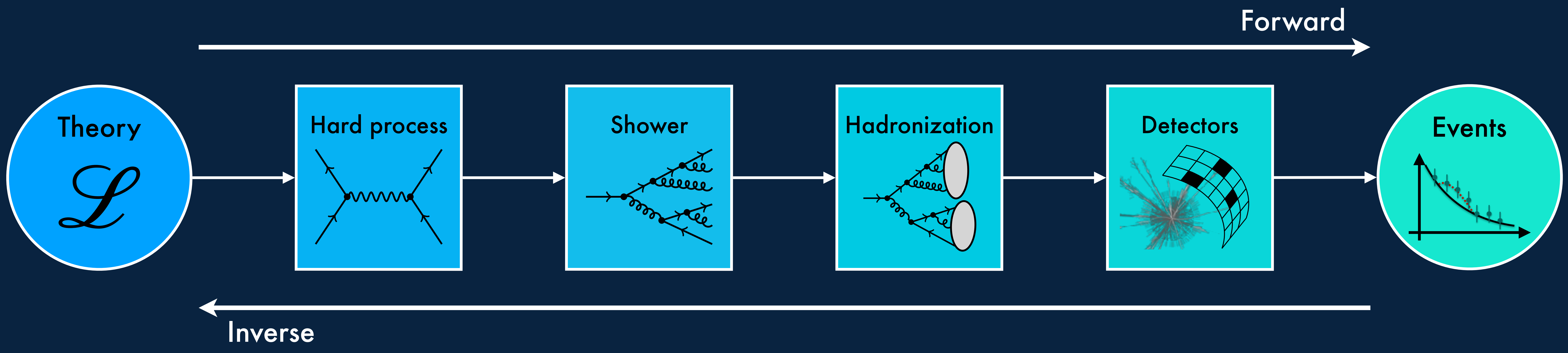
1. Precision simulations (a lot)
2. Optimized analyses for high-dimensional data



Understanding LHC data based on 1^{st} principles

What do we need to understand the data?

1. Precision simulations (a lot)
2. Optimized analyses for high-dimensional data

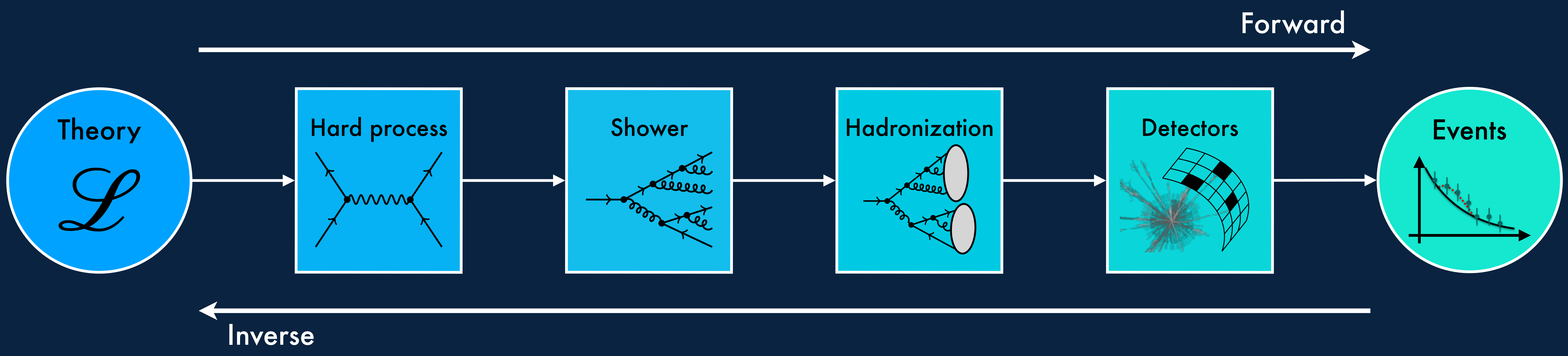


➡ Machine Learning has significant impact on all aspects

Understanding LHC data based on 1^{st} principles

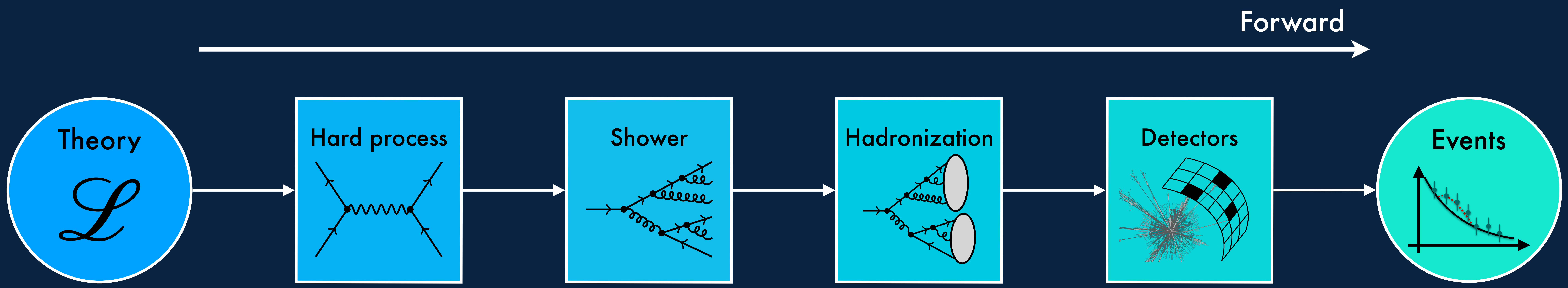
What do we need to understand the data?

1. Precision simulations (a lot) ← this talk
2. Optimized analyses for high-dimensional data

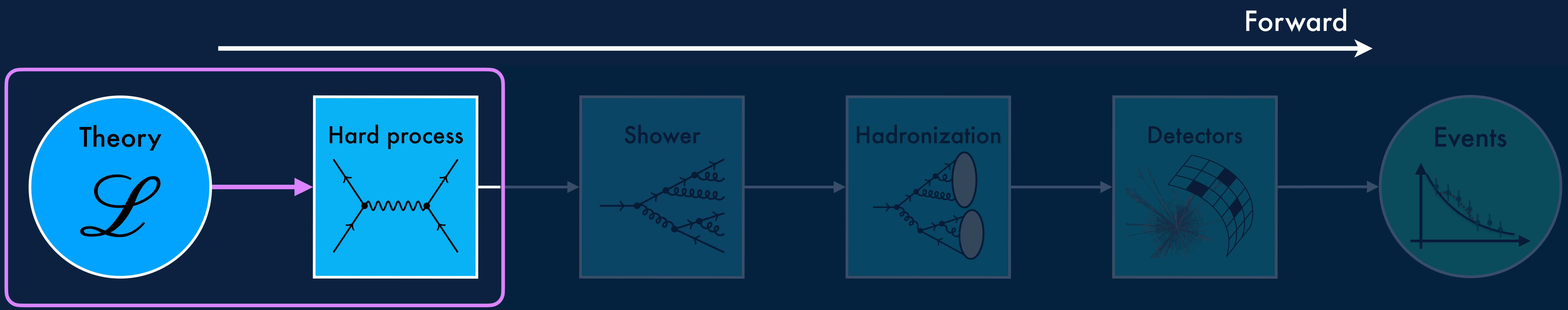


➡ Machine Learning has significant impact on all aspects

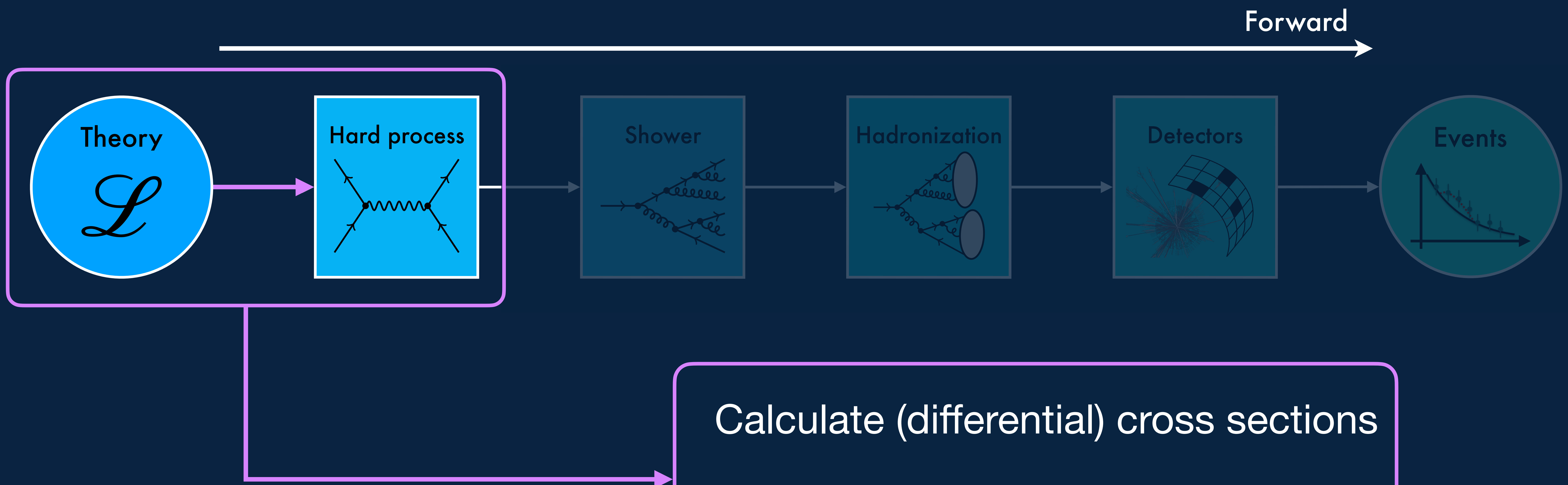
ML for forward simulations



ML for forward simulations



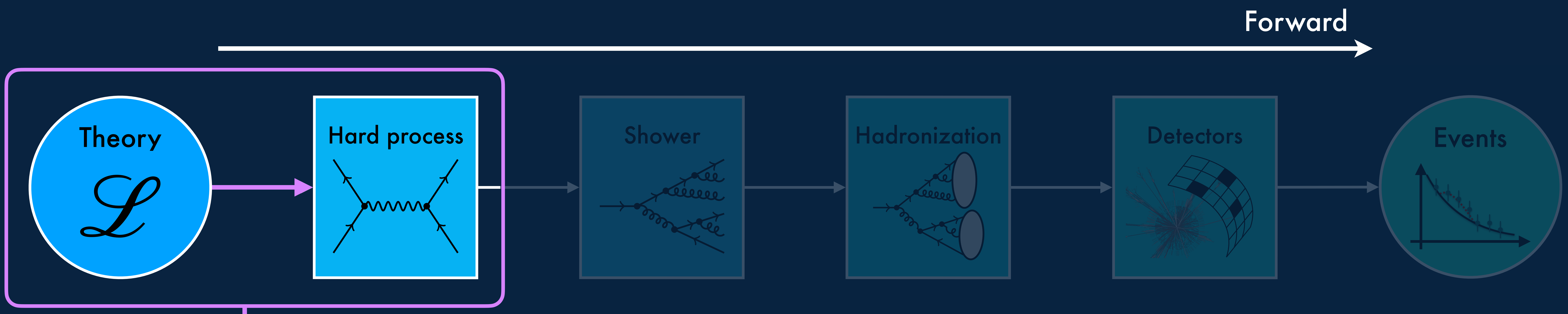
ML for forward simulations



Calculate (differential) cross sections

$$d\sigma \sim \text{pdf} \times |M(x)|^2 \times d\Phi$$

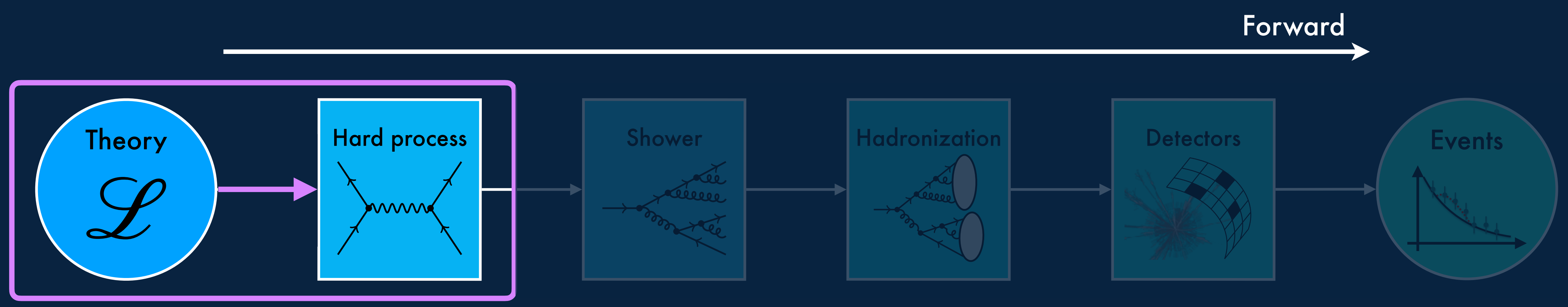
ML for forward simulations



Calculate (differential) cross sections

$$d\sigma \sim \text{pdf} \times \underbrace{|M(x)|^2}_{1} \times \underbrace{d\Phi}_{2}$$

ML for forward simulations



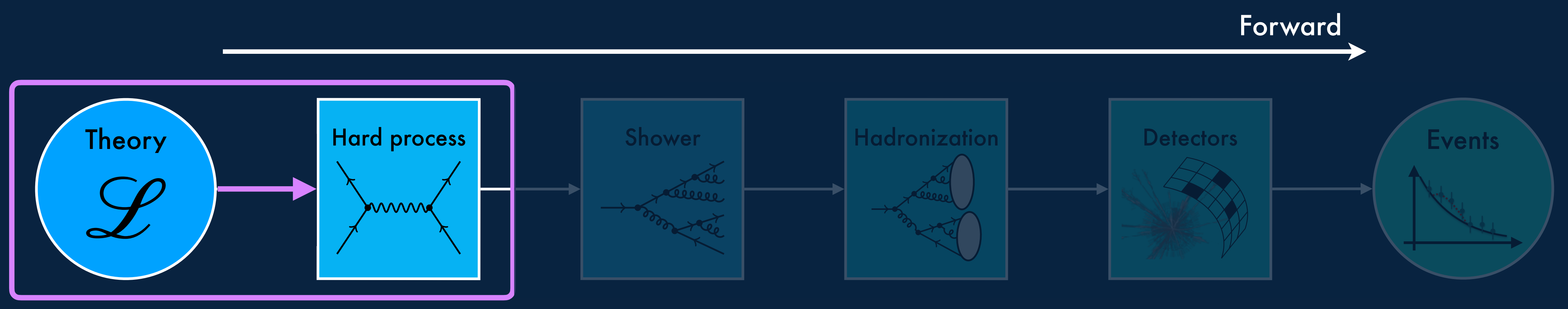
$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

Amplitudes: avoid expensive matrix element

- As "simple" **regression** task

[1912.11055, 2002.07516, 2006.16273, 2008.10949, 2104.14182, 2105.04898, 2106.09474, 2107.06625, 2109.11964, 2112.09145, 2201.04523, 2206.08901, 2206.04115, 2206.14831, 2301.13562, 2302.04005, 2306.07726,…..]

ML for forward simulations

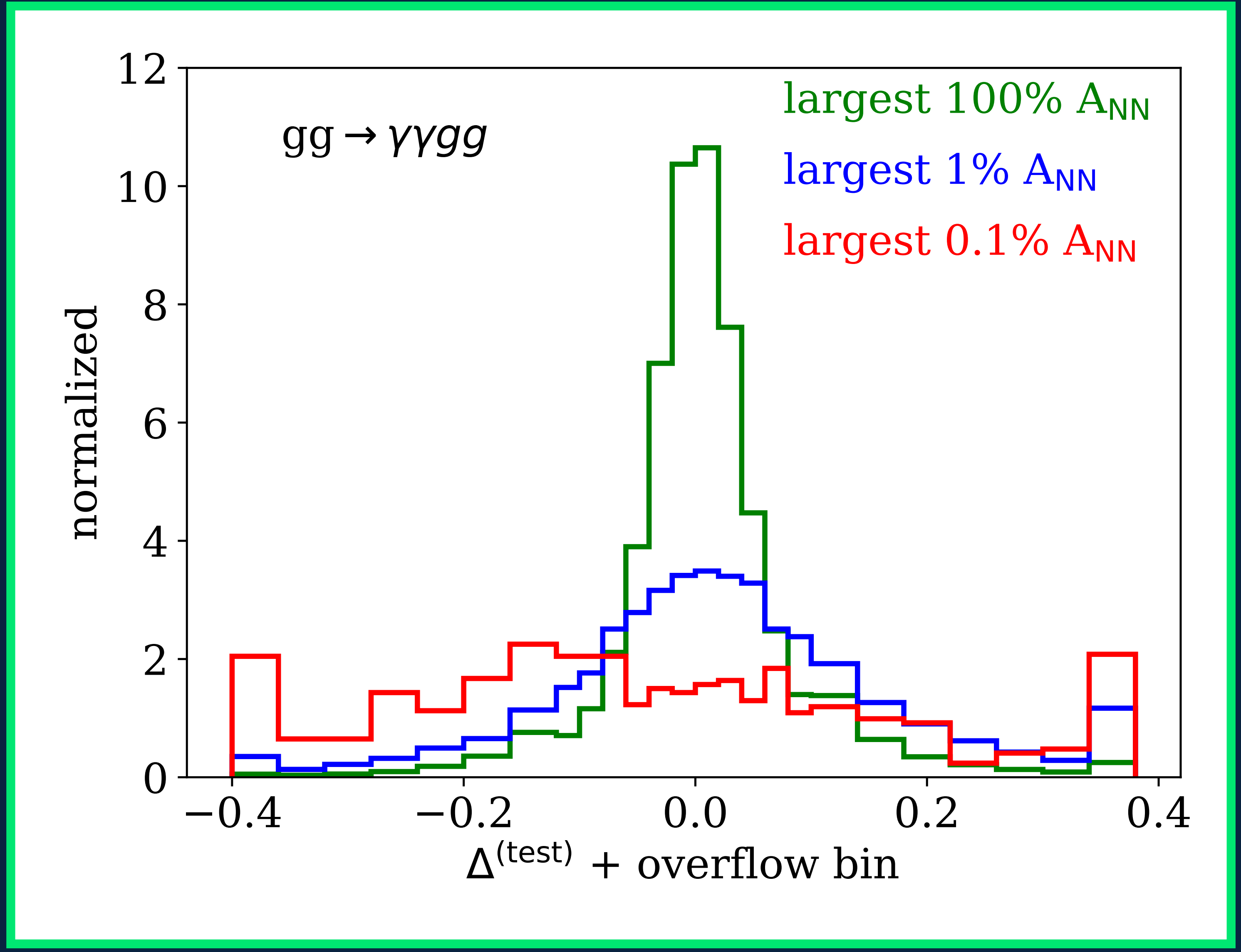


$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

Amplitudes: avoid expensive matrix element

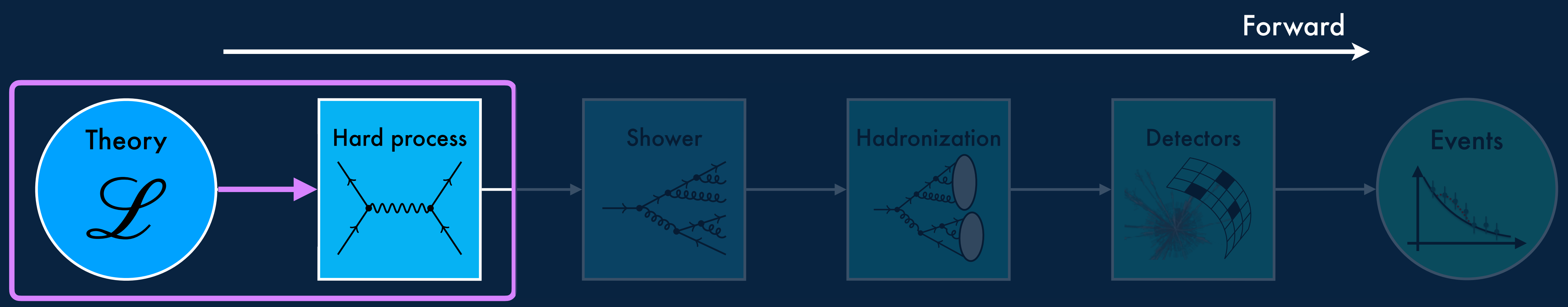
- As "simple" **regression** task
- With uncertainties/boosting using **Bayesian NN**

Badger, Butter, Luchman, Pitz, Plehn [2206.14831]



[1912.11055, 2002.07516, 2006.16273, 2008.10949, 2104.14182, 2105.04898, 2106.09474, 2107.06625, 2109.11964, 2112.09145, 2201.04523, 2206.08901, 2206.04115, 2206.14831, 2301.13562, 2302.04005, 2306.07726,…..]

ML for forward simulations

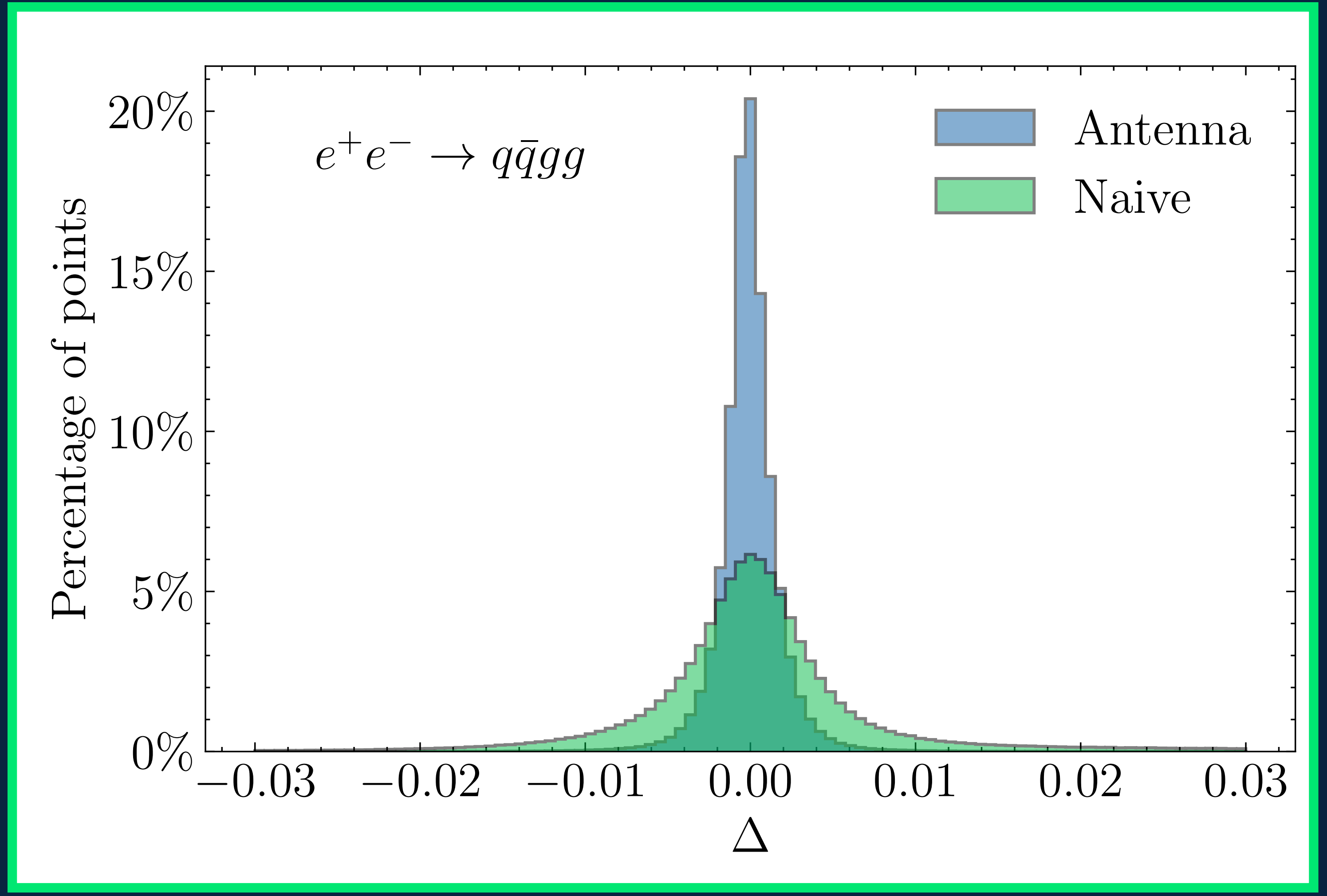


$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

Amplitudes: avoid expensive matrix element

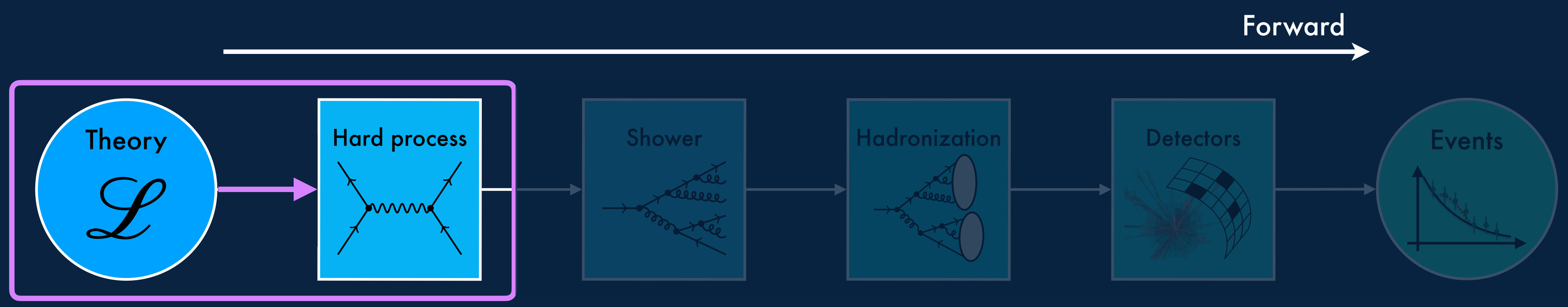
- As "simple" **regression** task
- With uncertainties/boosting using **Bayesian NN**
- Using factorisation ansatz to reach **‰ level** accuracy

Maître, Truong [2302.04005]



[1912.11055, 2002.07516, 2006.16273, 2008.10949, 2104.14182, 2105.04898, 2106.09474, 2107.06625, 2109.11964, 2112.09145, 2201.04523, 2206.08901, 2206.04115, 2206.14831, 2301.13562, 2302.04005, 2306.07726,…..]

ML for forward simulations

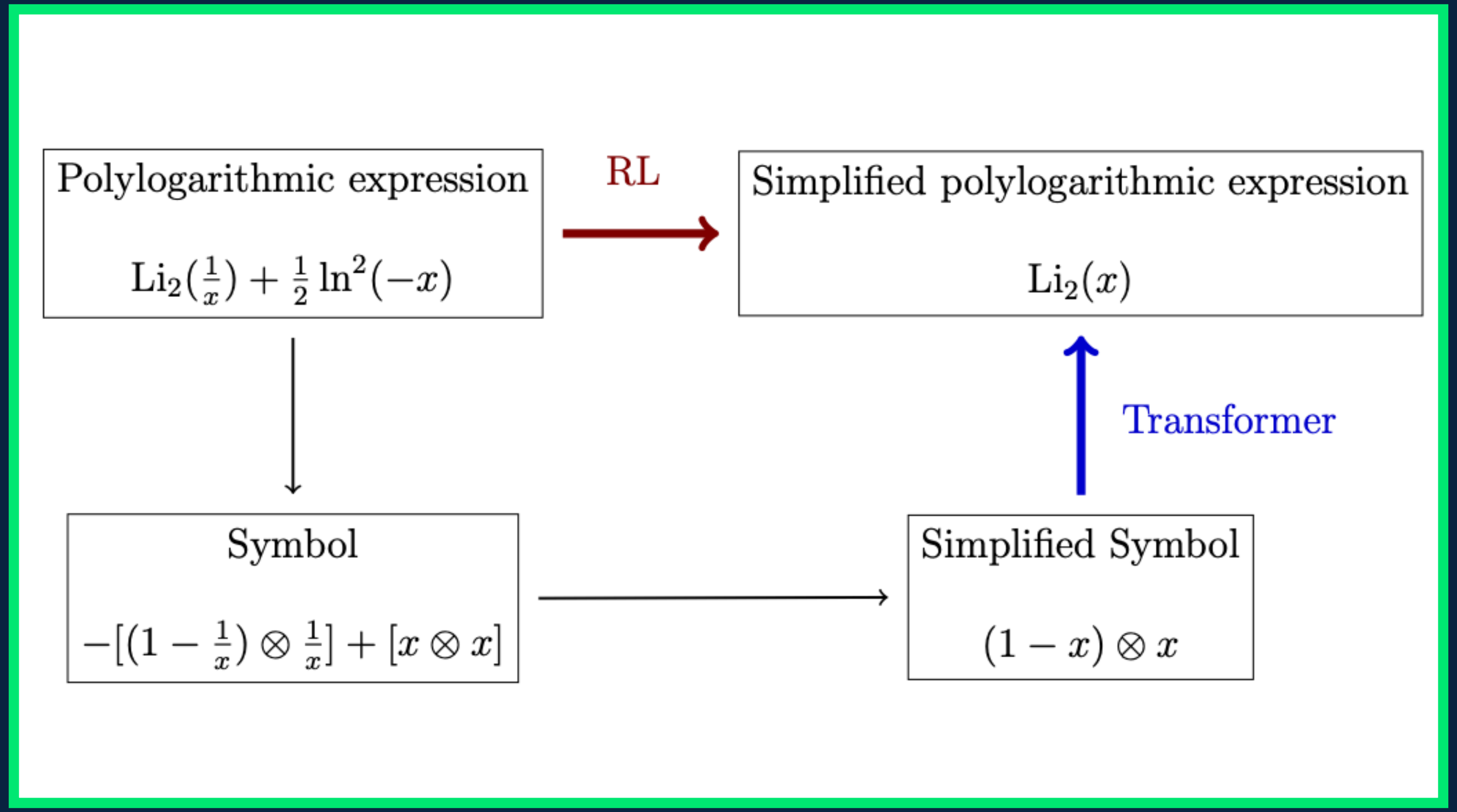


$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

Amplitudes: avoid expensive matrix element

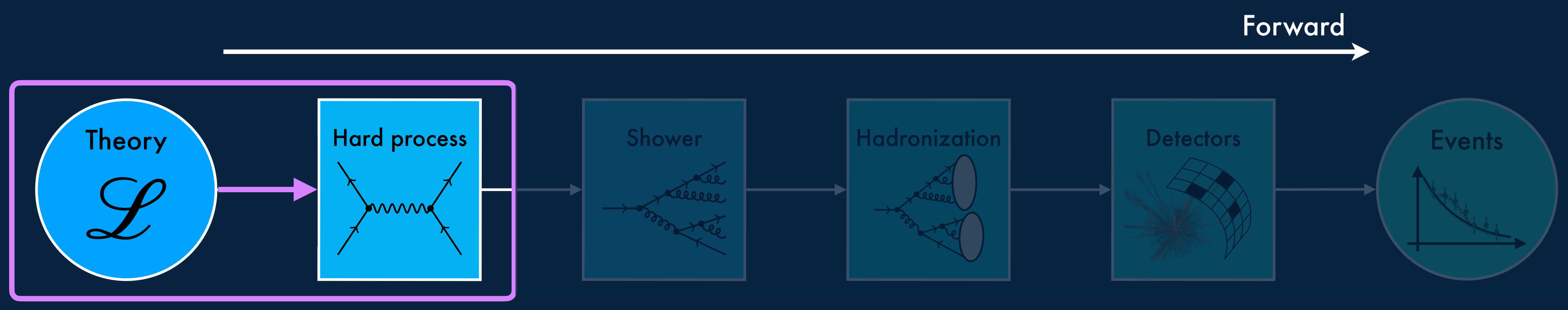
- As "simple" **regression** task
- With uncertainties/boosting using **Bayesian NN**
- Using factorisation ansatz to reach **‰ level** accuracy
- **RL and/or Transformer** for simplifications of Polylogarithms

Dersey, Schwartz, Zhang [2206.04115]



[1912.11055, 2002.07516, 2006.16273, 2008.10949, 2104.14182, 2105.04898, 2106.09474, 2107.06625, 2109.11964, 2112.09145, 2201.04523, 2206.08901, 2206.04115, 2206.14831, 2301.13562, 2302.04005, 2306.07726,…..]

ML for forward simulations



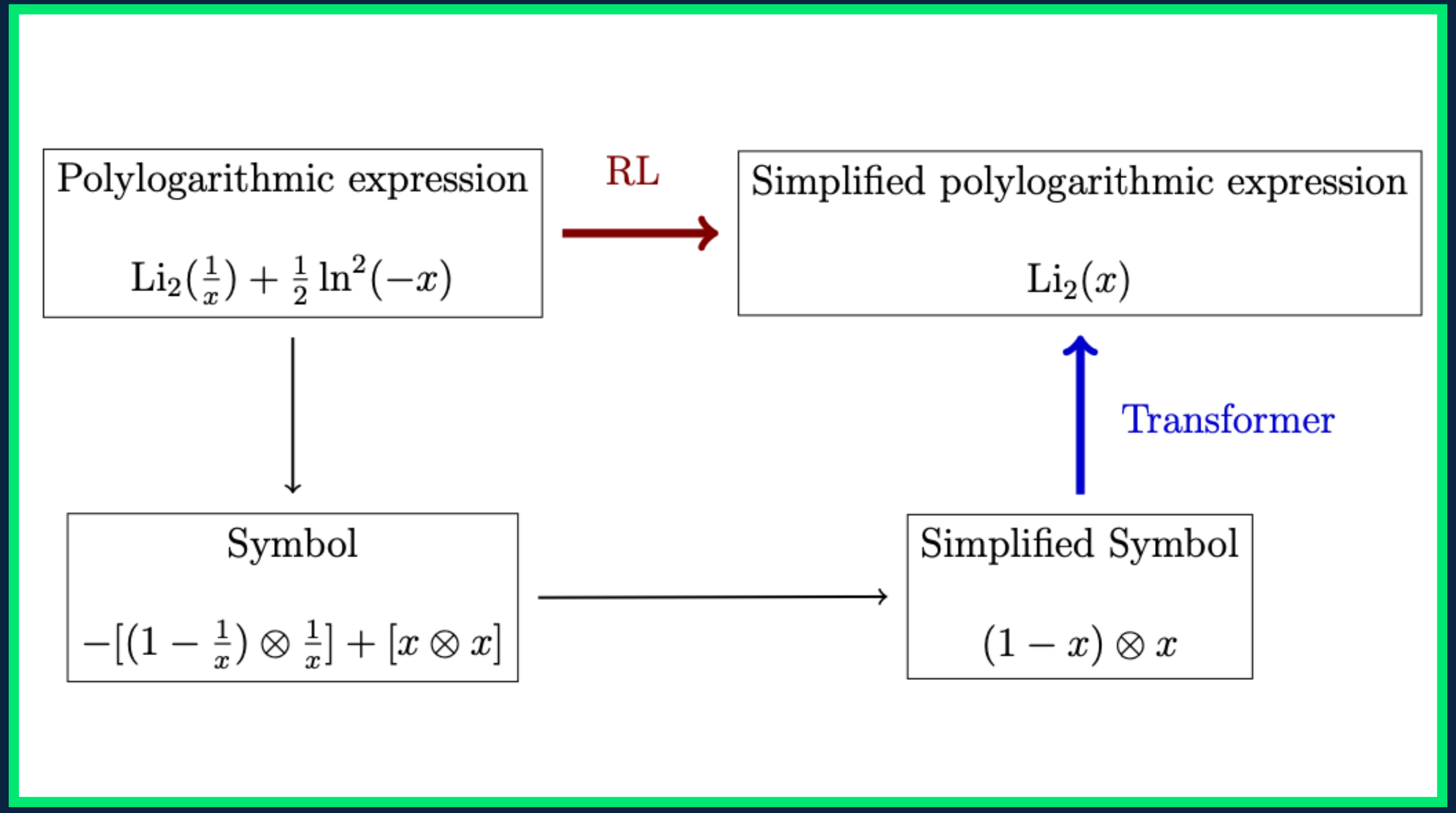
$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

Amplitudes: avoid expensive matrix element

- As "simple" **regression** task
- With uncertainties/boosting using **Bayesian NN**
- Using factorisation ansatz to reach **‰ level** accuracy
- **RL and/or Transformer** for simplifications of Polylogarithms
- NN-assisted **contour deformation** (Loop integrals)

[1912.11055, 2002.07516, 2006.16273, 2008.10949, 2104.14182, 2105.04898, 2106.09474, 2107.06625, 2109.11964, 2112.09145, 2201.04523, 2206.08901, 2206.04115, 2206.14831, 2301.13562, 2302.04005, 2306.07726,…..]

Dersey, Schwartz, Zhang [2206.04115]



NNContour

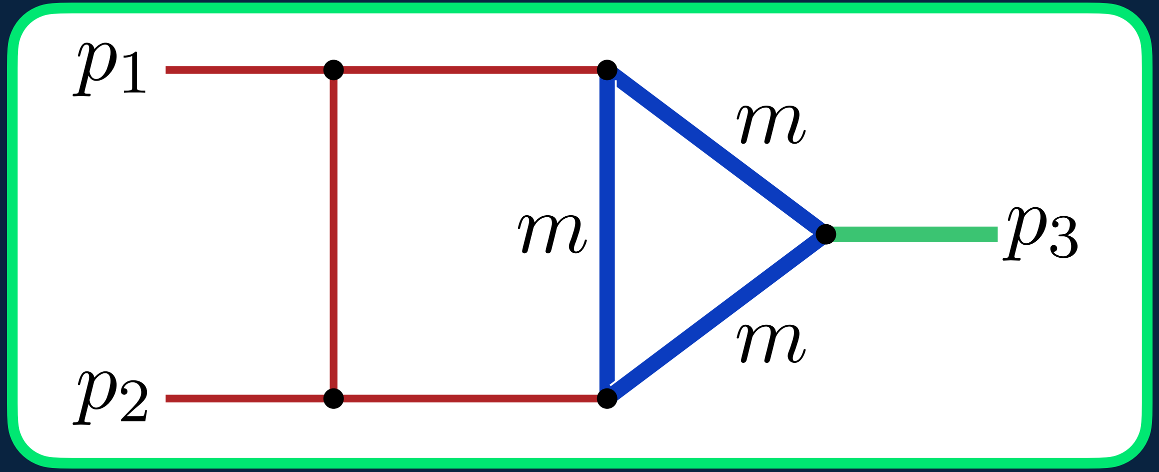
ML for loop integrals

RW, Magerya, Villa, Jones, Kerner, Butter, Heinrich, Plehn [2112.09145]

NNContour — ML for loop integrals

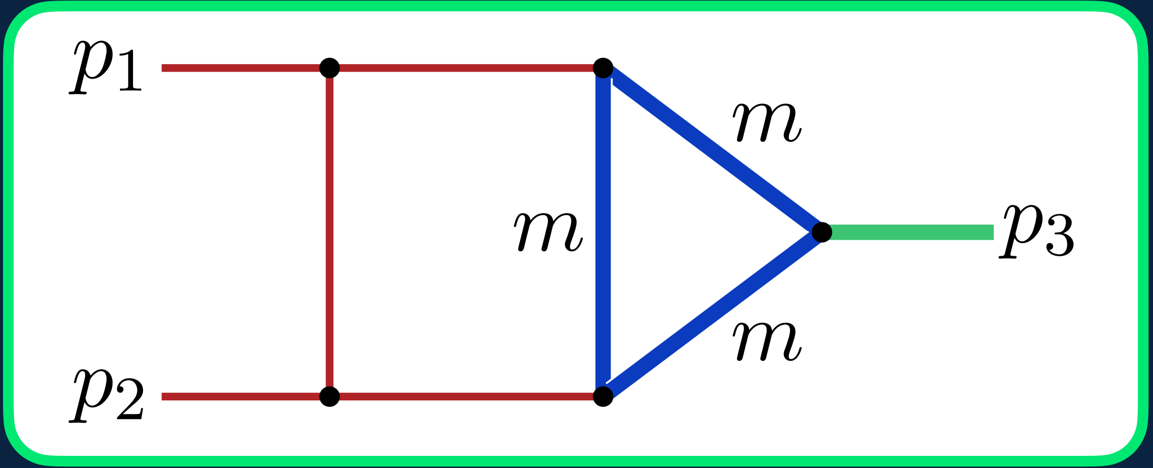
$$G \propto \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu-LD/2}}$$

Contains singularities



NNContour — ML for loop integrals

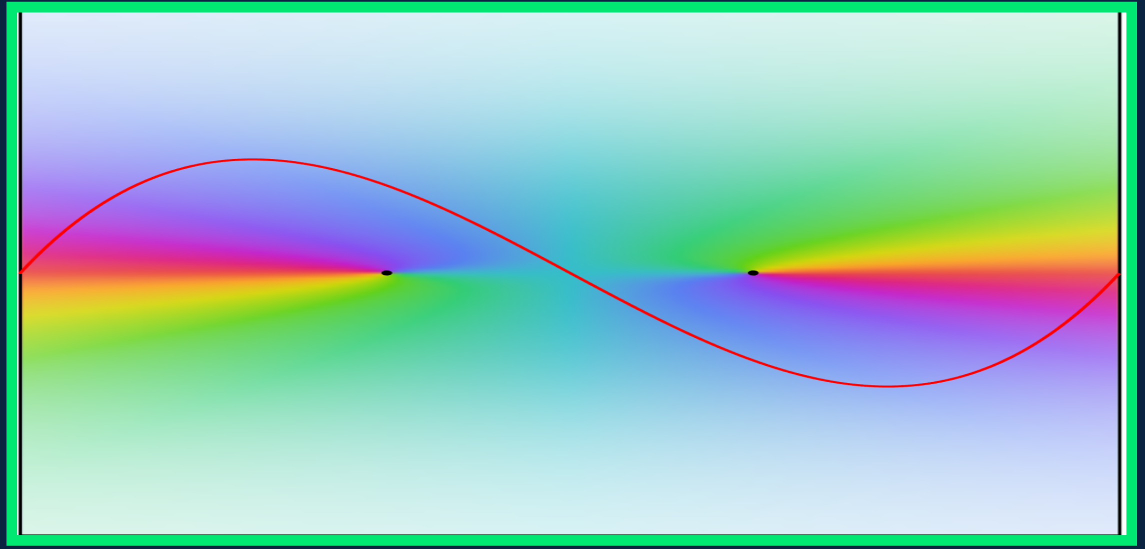
$$G \propto \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j\, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu-LD/2}}$$



Contains singularities

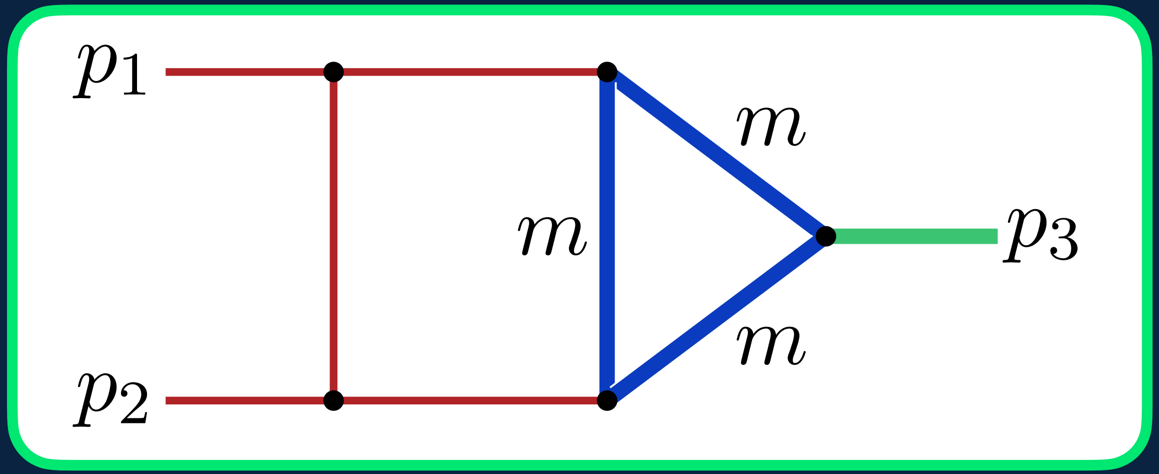
Cauchy-Theorem → Contour deformation

$$\int_0^1 \prod_{j=1}^{N} \mathrm{d}x_j\, I(\vec{x}) = \int_\gamma \prod_{j=1}^{N} \mathrm{d}z_j\, I(\vec{z})$$



NNContour — ML for loop integrals

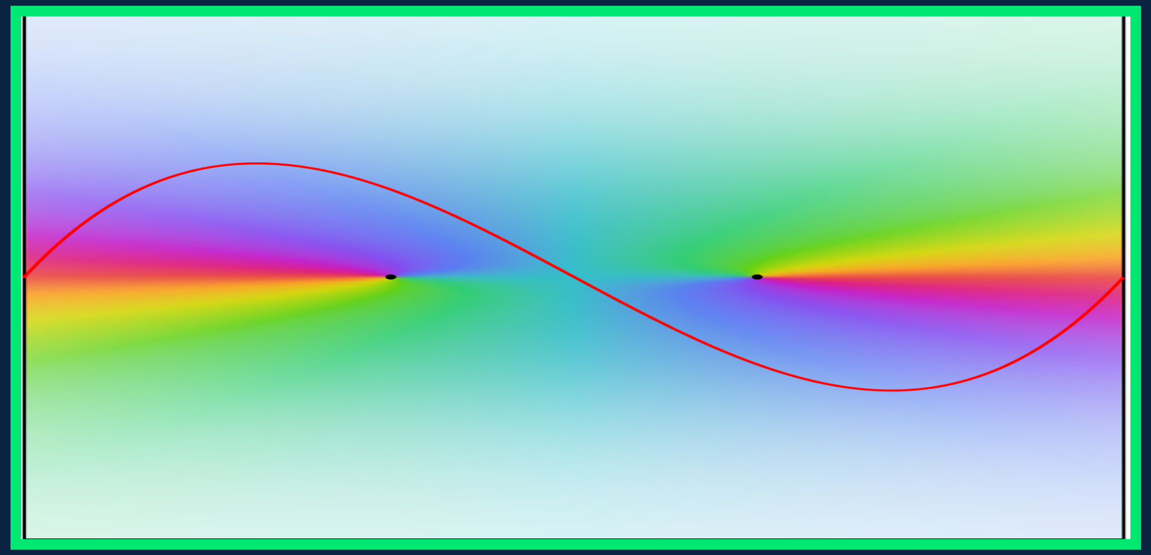
$$G \propto \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j\, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu-LD/2}}$$



Contains singularities

Cauchy-Theorem → Contour deformation

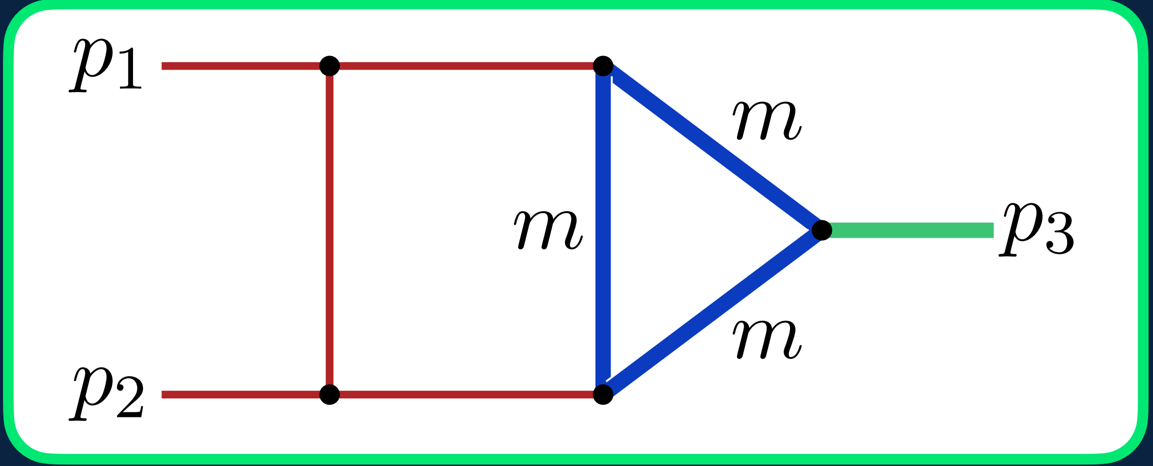
$$\int_0^1 \prod_{j=1}^{N} \mathrm{d}x_j\, I(\vec{x}) = \int_\gamma \prod_{j=1}^{N} \mathrm{d}z_j\, I(\vec{z})$$



Parametrize with **NF + NN**

NNContour — ML for loop integrals

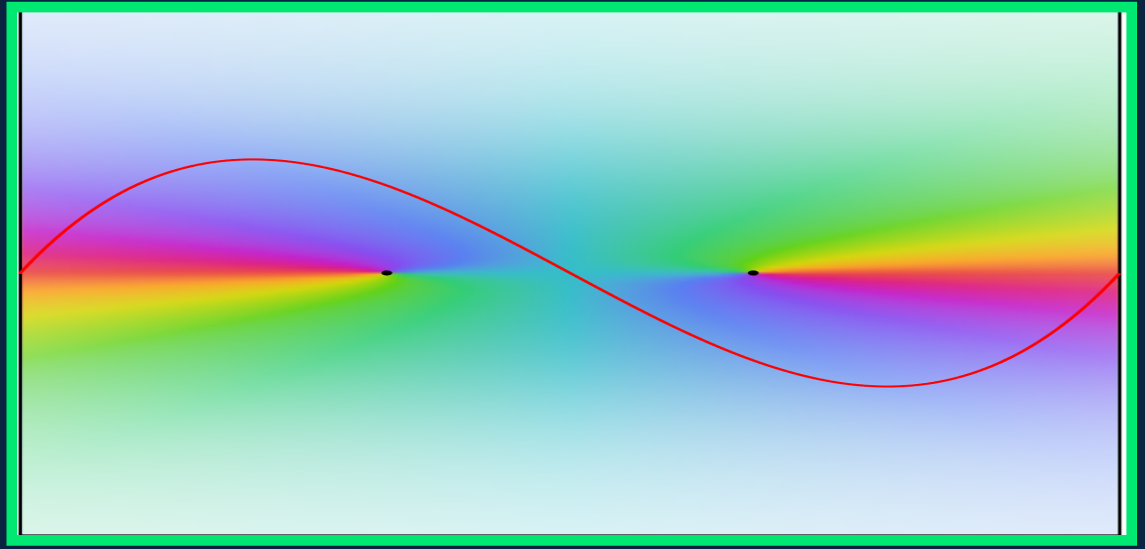
$$G \propto \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu-LD/2}}$$



Contains singularities

Cauchy-Theorem → Contour deformation

$$\int_0^1 \prod_{j=1}^{N} \mathrm{d}x_j \, I(\vec{x}) = \int_\gamma \prod_{j=1}^{N} \mathrm{d}z_j \, I(\vec{z})$$



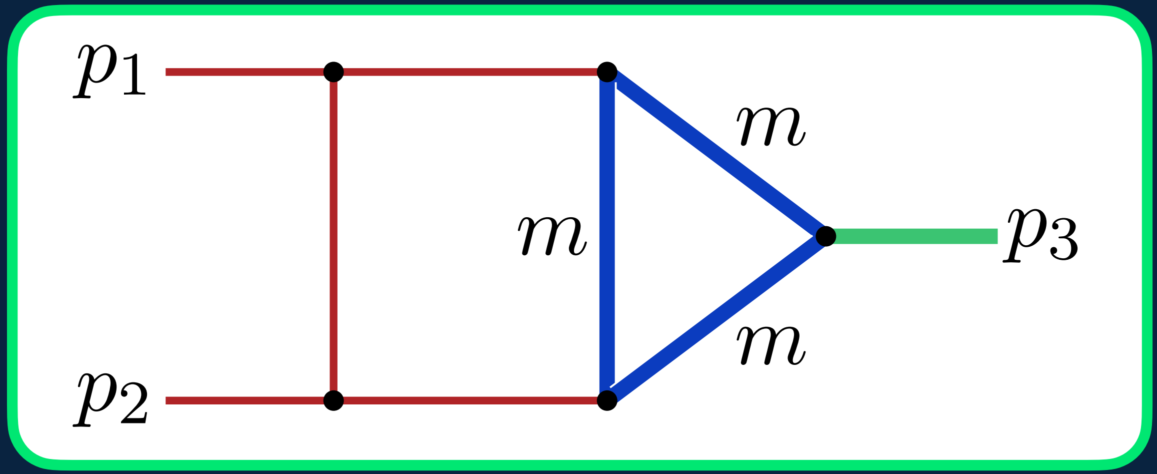
Parametrize with **NF + NN**

→ Optimal parametrization = small variance

NNContour — ML for loop integrals

RW, Magerya, Villa, Jones, Kerner, Butter, Heinrich, Plehn [2112.09145]

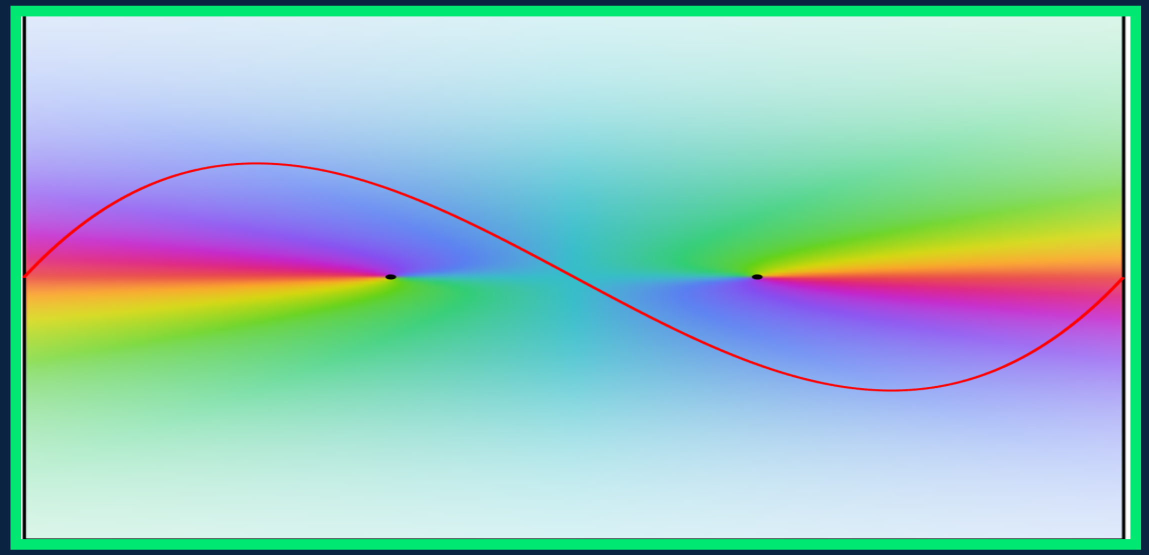
$$G \propto \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j\, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu-LD/2}}$$



Contains singularities

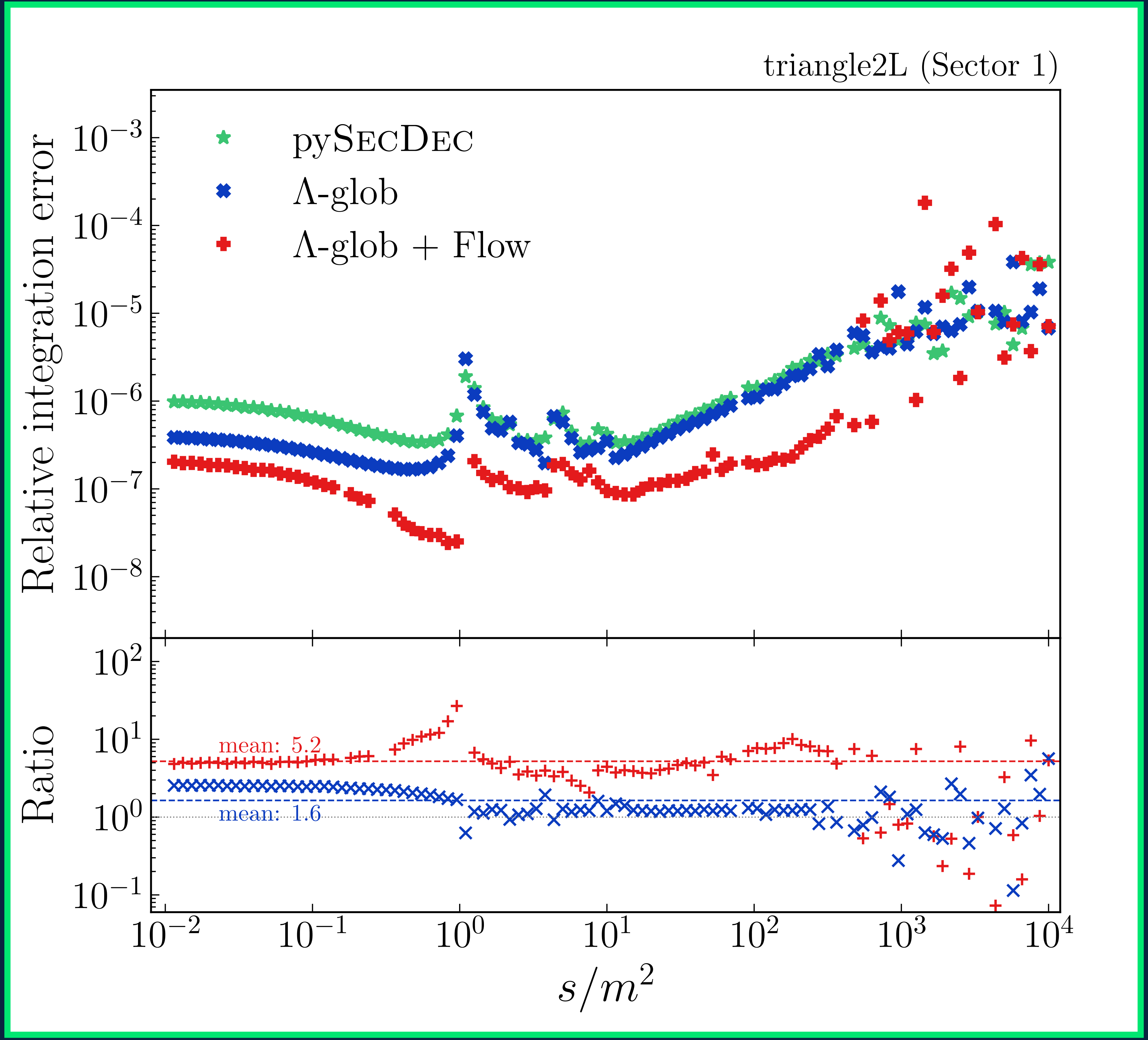
Cauchy-Theorem → Contour deformation

$$\int_0^1 \prod_{j=1}^{N} \mathrm{d}x_j\, I(\vec{x}) = \int_\gamma \prod_{j=1}^{N} \mathrm{d}z_j\, I(\vec{z})$$



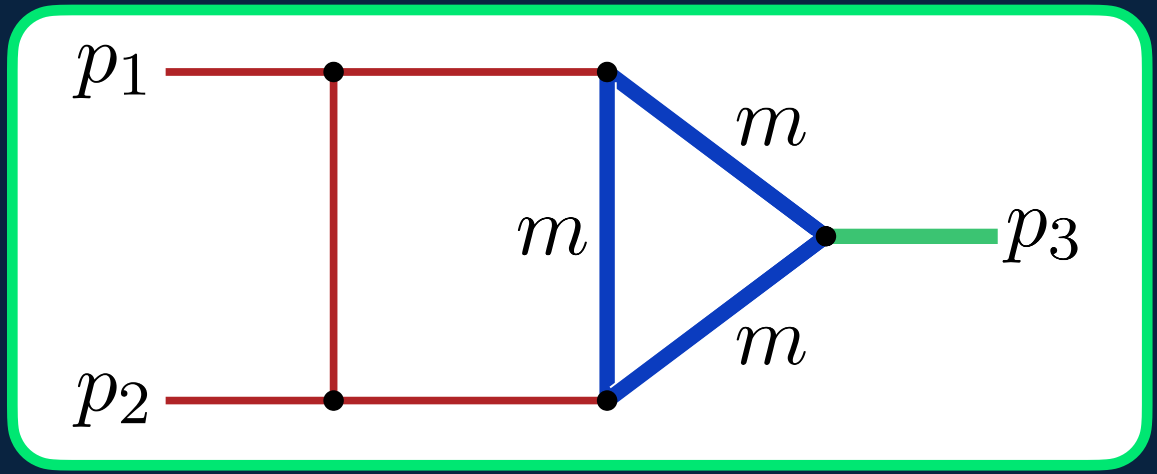
Parametrize with **NF + NN**

→ Optimal parametrization = small variance



NNContour — ML for loop integrals

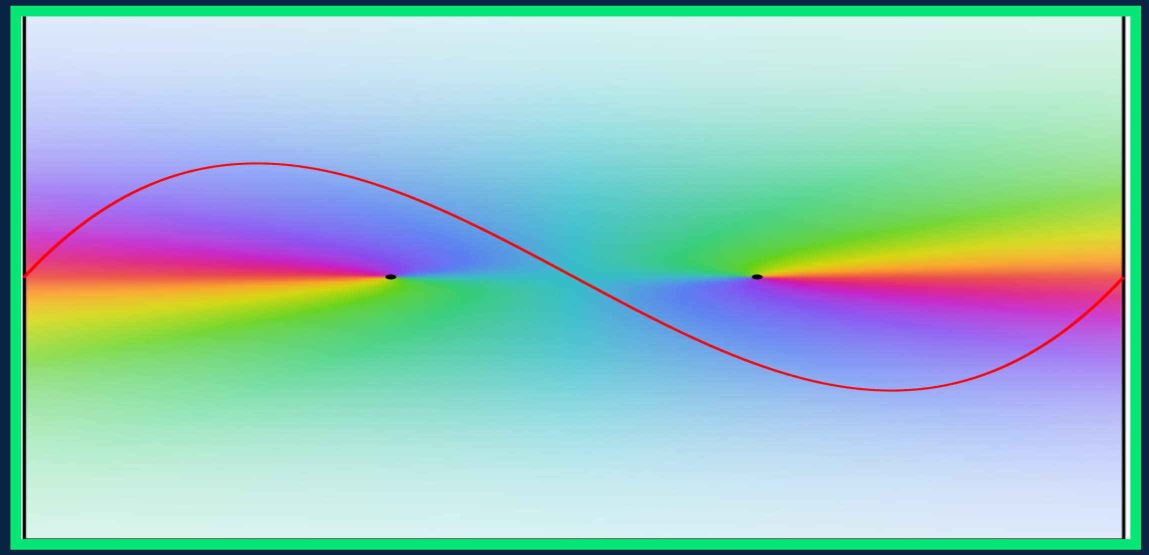
$$G \propto \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j\, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu-LD/2}}$$



Contains singularities

Cauchy-Theorem → Contour deformation

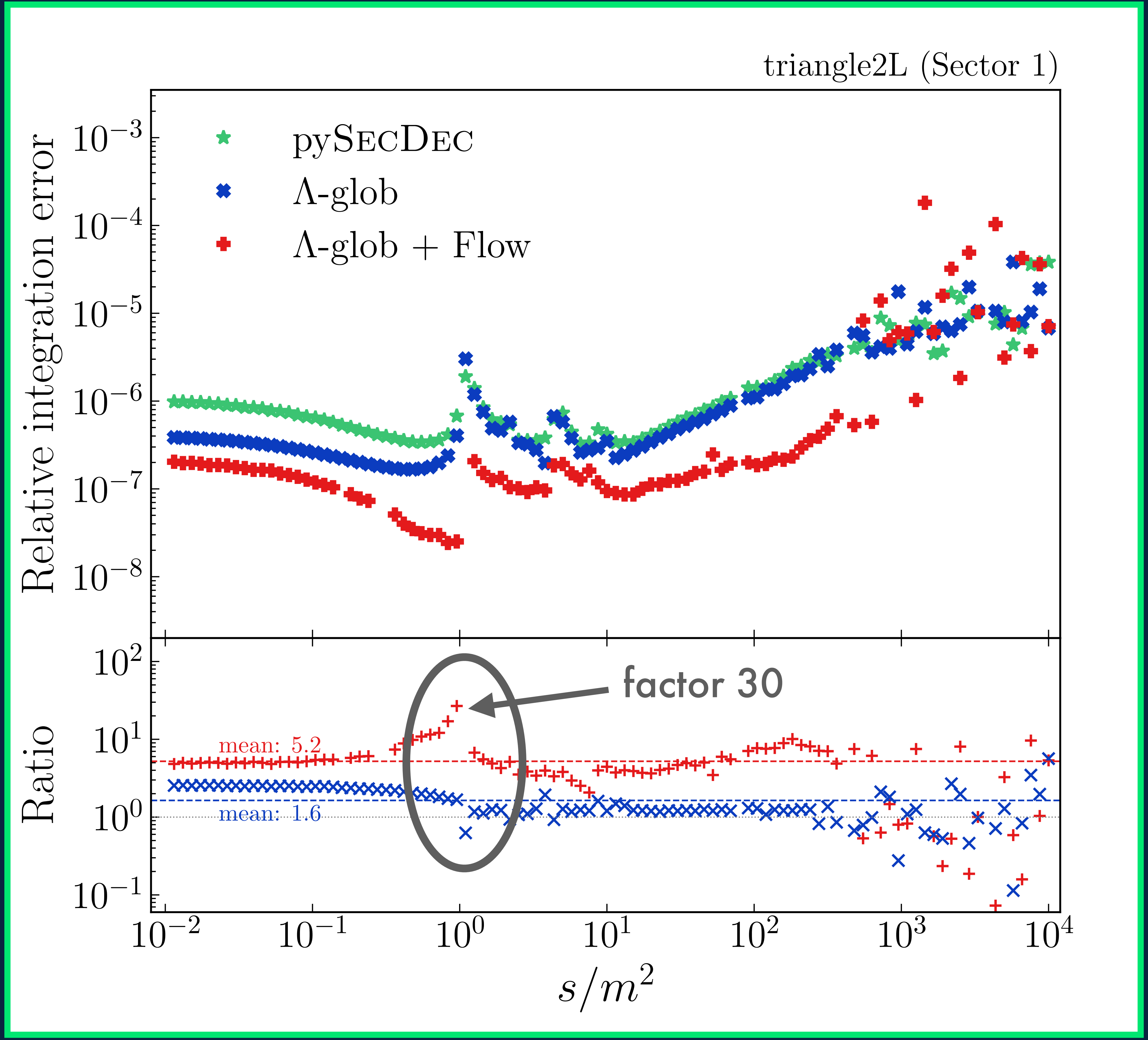
$$\int_0^1 \prod_{j=1}^{N} \mathrm{d}x_j\, I(\vec{x}) = \int_\gamma \prod_{j=1}^{N} \mathrm{d}z_j\, I(\vec{z})$$



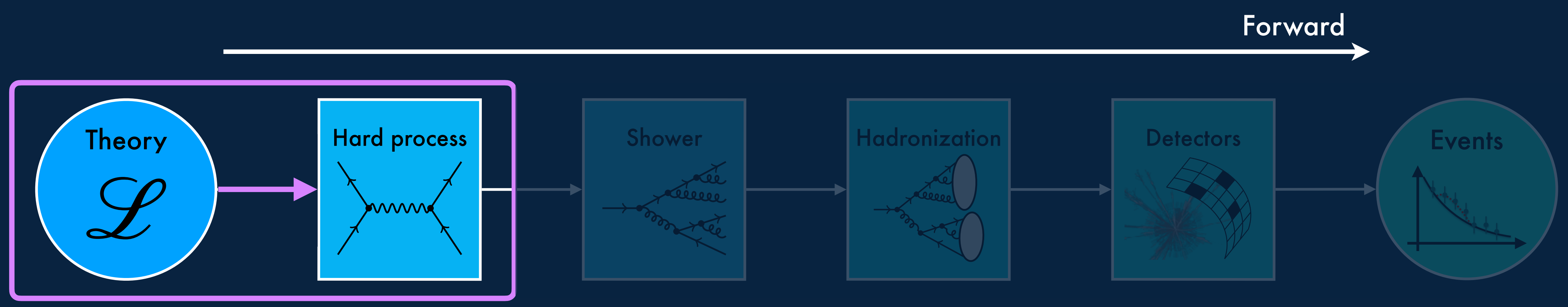
Parametrize with **NF + NN**

→ Optimal parametrization = small variance

RW, Magerya, Villa, Jones, Kerner, Butter, Heinrich, Plehn [2112.09145]

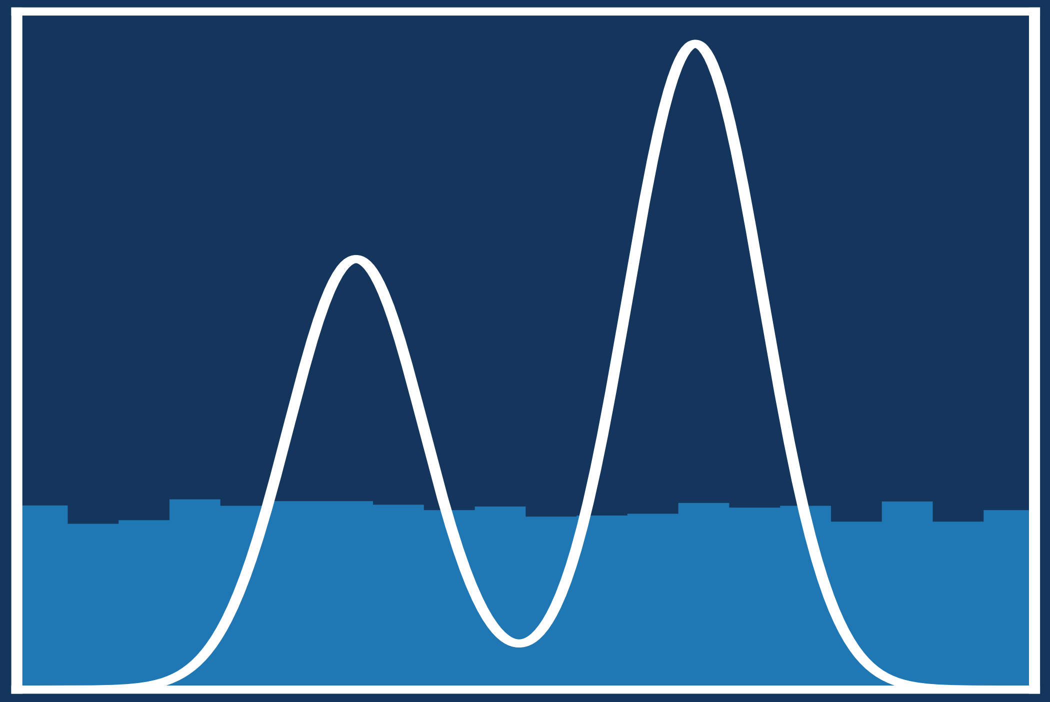


ML for forward simulations



$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

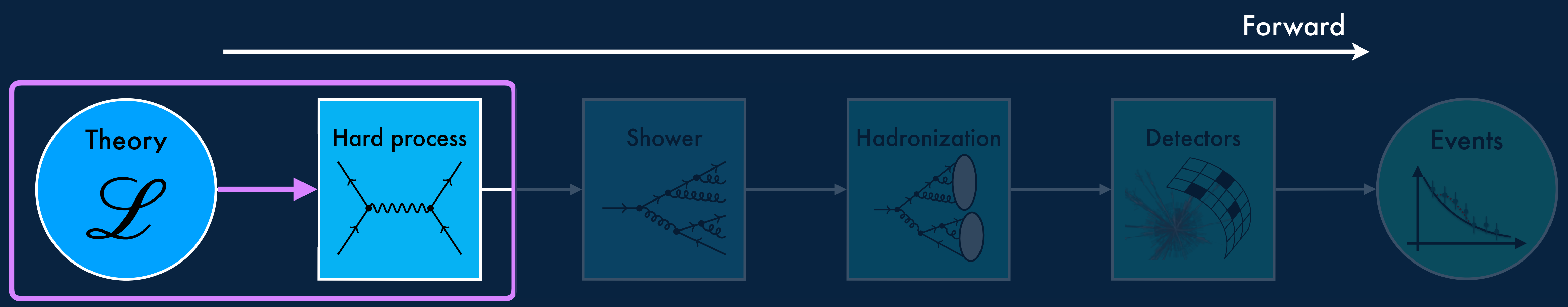
Phase space: increase unweighting efficiency



Flat sampling:
inefficient

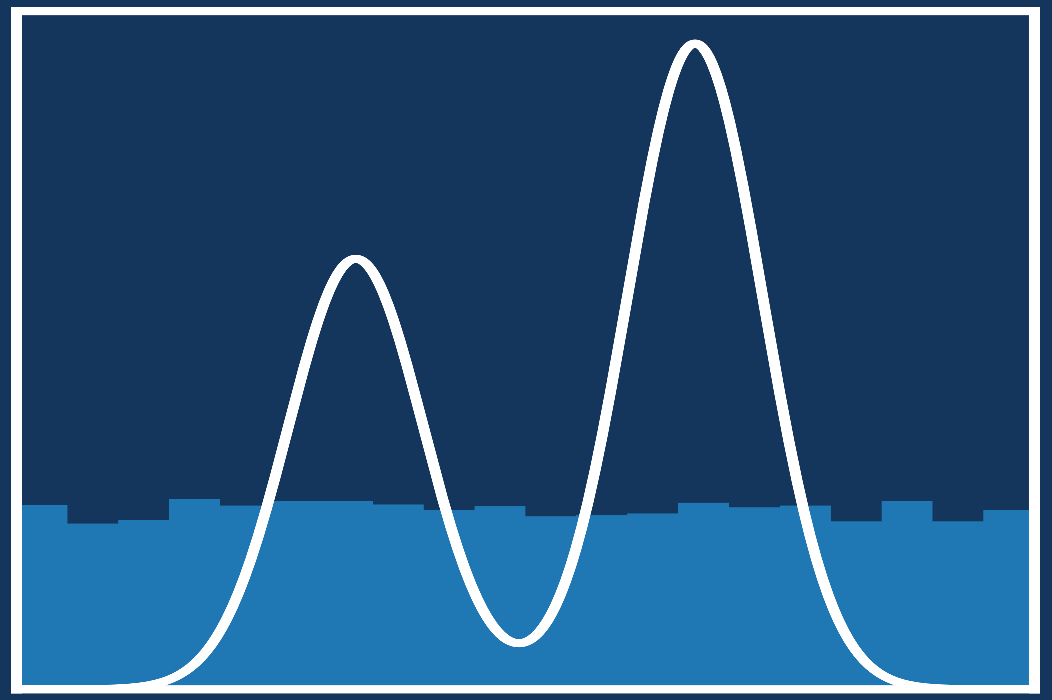
$$I = \langle f(x) \rangle_{x \sim \text{unif}}$$

ML for forward simulations



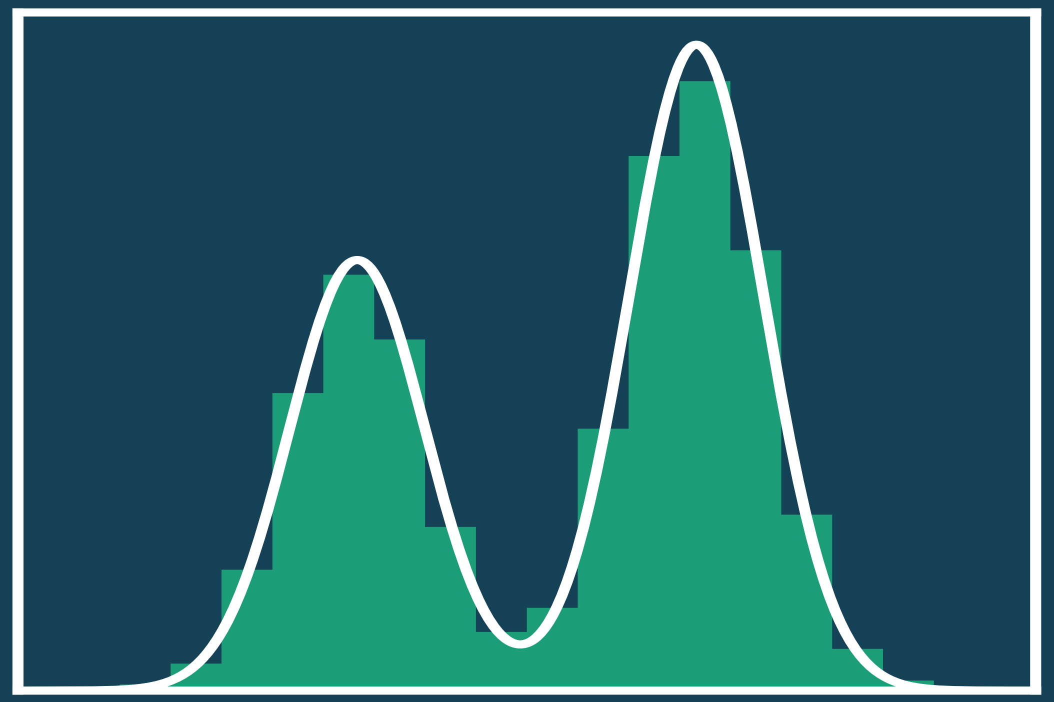
$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

Phase space: increase unweighting efficiency



Flat sampling:
inefficient

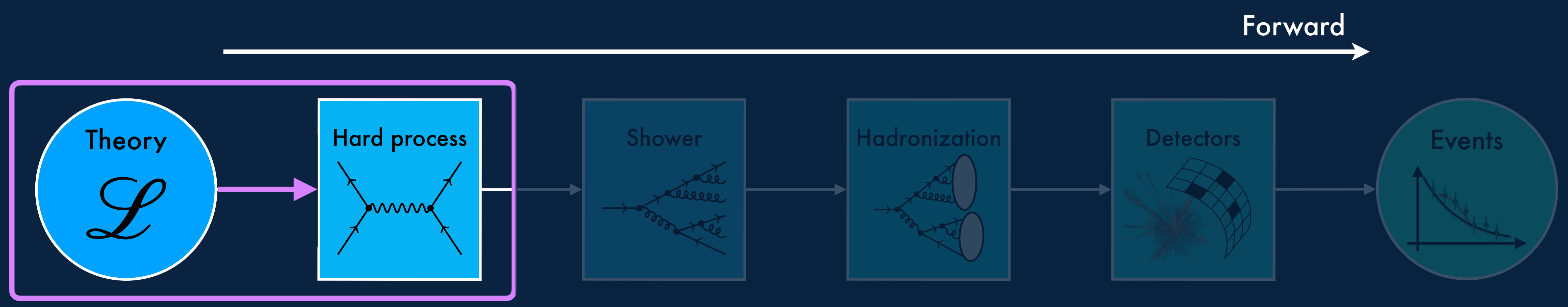
$$I = \langle f(x) \rangle_{x \sim \text{unif}}$$



Importance sampling:
find g close to f

$$I = \left\langle \frac{f(x)}{g(x)} \right\rangle_{x \sim g(x)}$$

ML for forward simulations



$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

Phase space: increase unweighting efficiency

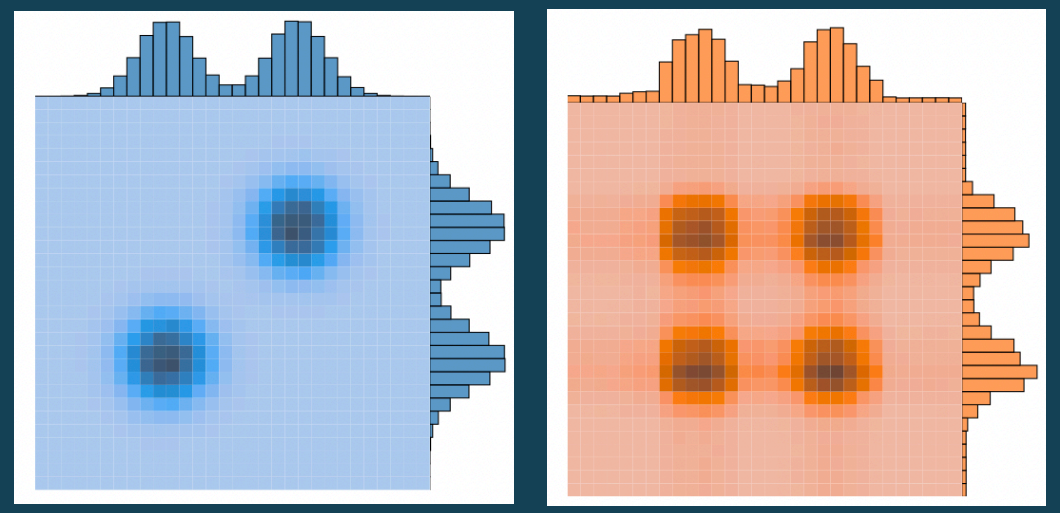
- Standard **VEGAS** approach → **fast** but **no correlations**

[G. P. Lepage, 1978]

⊕ Computationally cheap

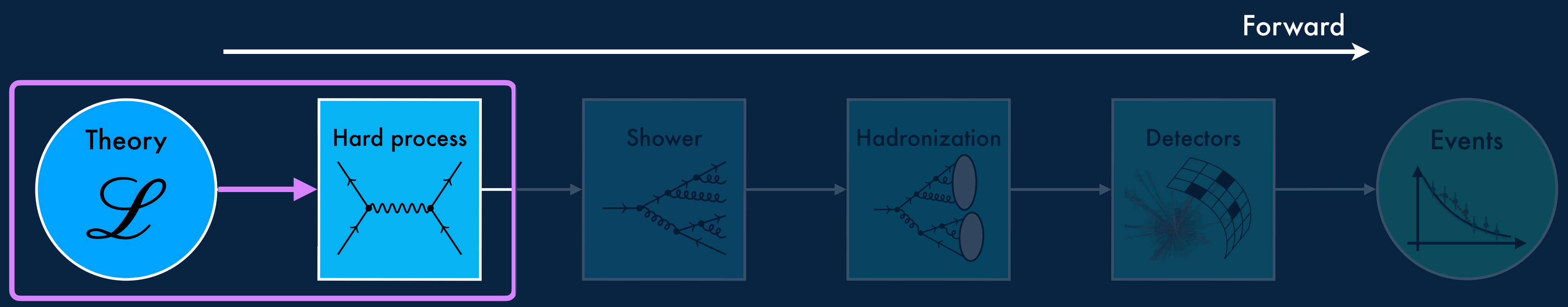
⊖ High-dim and rich peaking functions
→ **slow convergence**

⊖ Peaks not aligned with grid axes
→ **phantom peaks**



[1707.00028, 1810.11509, 2001.05478, 2001.05486, 2001.10028, 2005.12719, 2009.07819, 2011.13445, 2109.11964, 2112.09145, 2212.06172, 2301.13562, 2309.12369, 2311.01548…..]

ML for forward simulations



$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

Phase space: increase unweighting efficiency

- Standard **VEGAS** approach → **fast** but **no correlations**
- Improve with **NN** → **correlations** but **unstable**

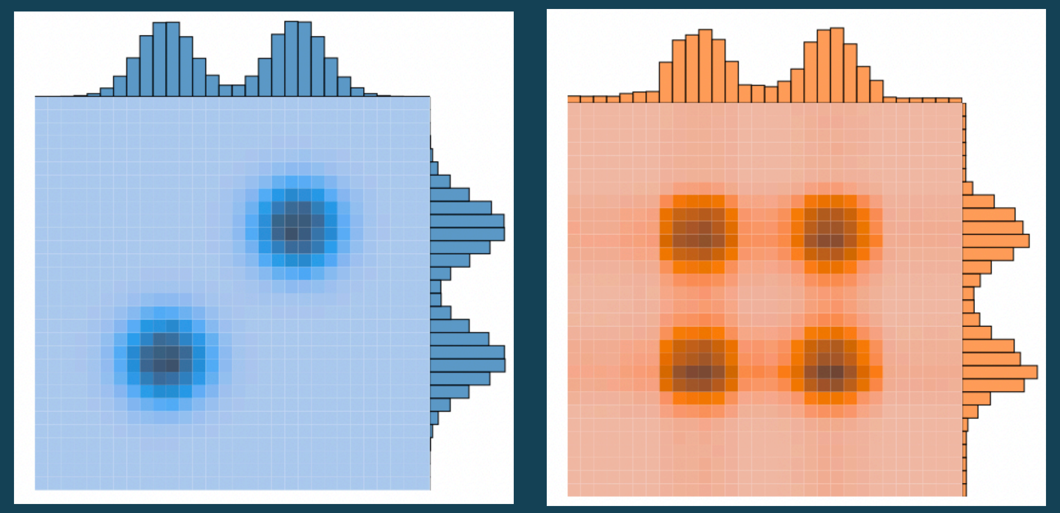
[1707.00028, 1810.11509, 2001.05478, 2001.05486, 2001.10028, 2005.12719, 2009.07819, 2011.13445, 2109.11964, 2112.09145, 2212.06172, 2301.13562, 2309.12369, 2311.01548…..]

[G. P. Lepage, 1978]

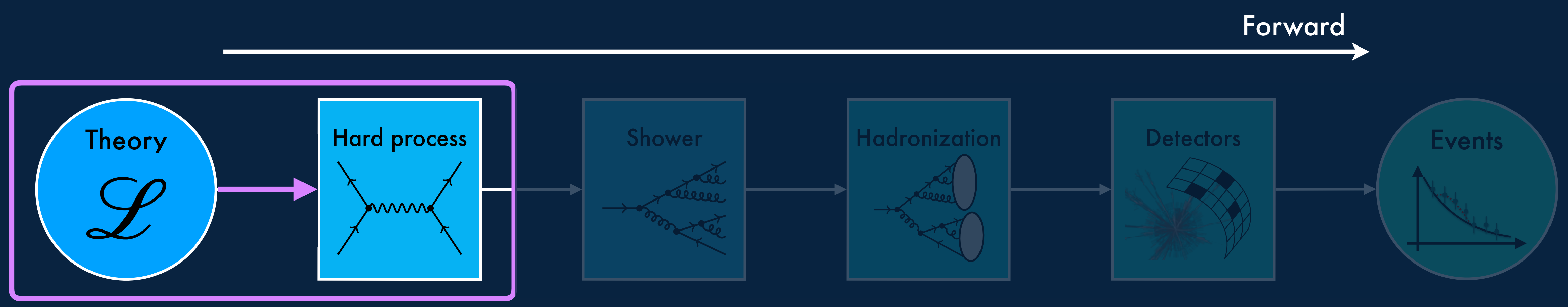
⊕ Computationally cheap

⊖ High-dim and rich peaking functions → **slow convergence**

⊖ Peaks not aligned with grid axes → **phantom peaks**



ML for forward simulations

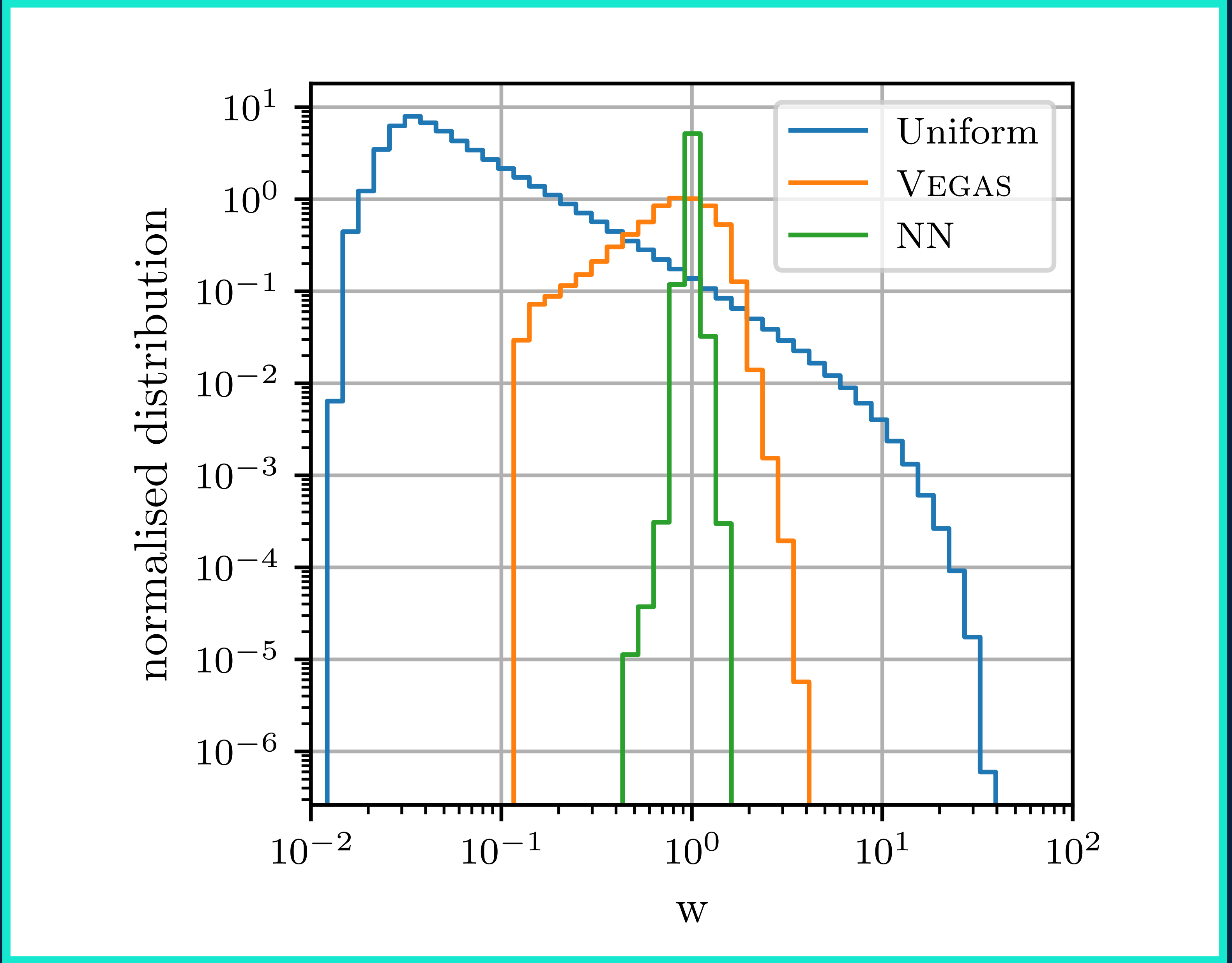


$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

Phase space: increase unweighting efficiency

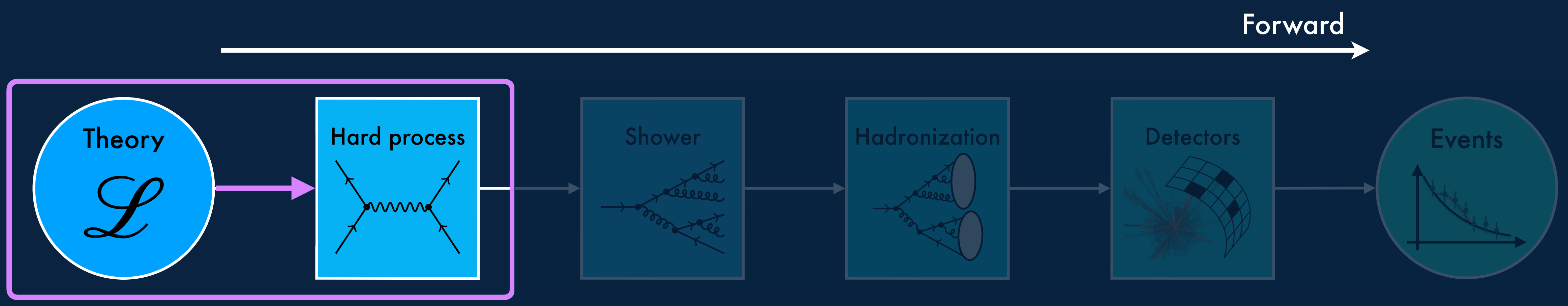
- Standard **VEGAS** approach → **fast** but **no correlations**
- Improve with **NN** → **correlations** but **unstable**
- Use **normalizing flows** → **correlations** and **stable**

Bothmann, Janßen, Knobbe, Schmale, Schumann [2001.05478]



[1707.00028, 1810.11509, 2001.05478, 2001.05486, 2001.10028, 2005.12719, 2009.07819, 2011.13445, 2109.11964, 2112.09145, 2212.06172, 2301.13562, 2309.12369, 2311.01548…..]

ML for forward simulations



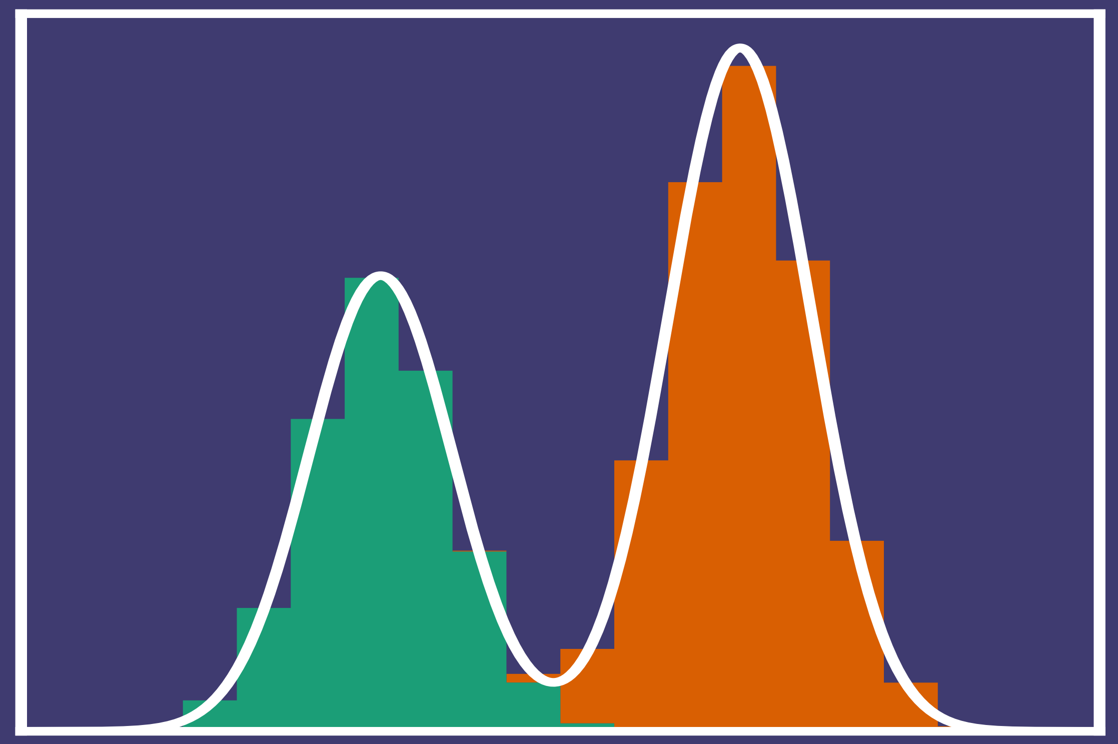
$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

Phase space: increase unweighting efficiency

- Standard **VEGAS** approach → **fast** but **no correlations**
- Improve with **NN** → **correlations** but **unstable**
- Use **normalizing flows** → **correlations** and **stable**
- **Multi-channel** approach → split the integral

[1707.00028, 1810.11509, 2001.05478, 2001.05486, 2001.10028, 2005.12719, 2009.07819, 2011.13445, 2109.11964, 2112.09145, 2212.06172, 2301.13562, 2309.12369, 2311.01548…..]

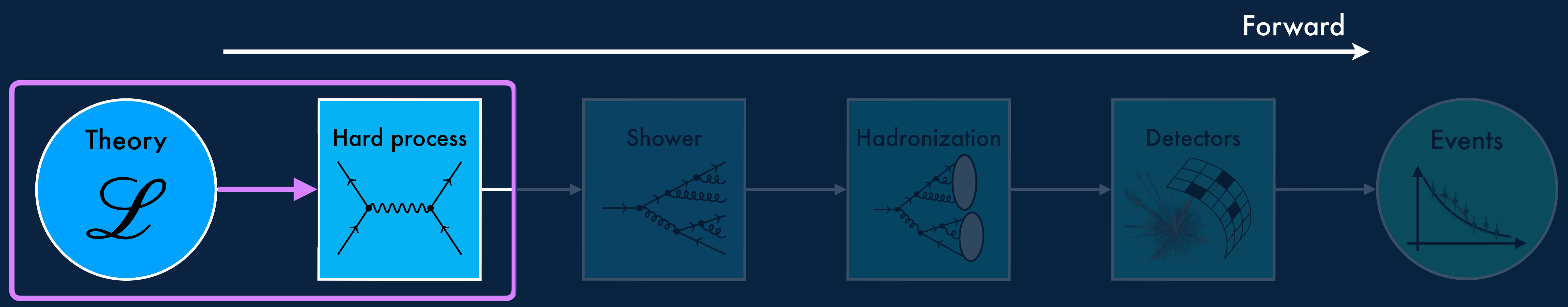
Kleiss, Pittau [hep-ph/9405257], Maltoni, Stelzer [hep-ph/0208156]



Multi-channel:
one map for each channel

$$I = \sum_i \left\langle \alpha_i(x) \frac{f(x)}{g_i(x)} \right\rangle_{x \sim g_i(x)}$$

ML for forward simulations



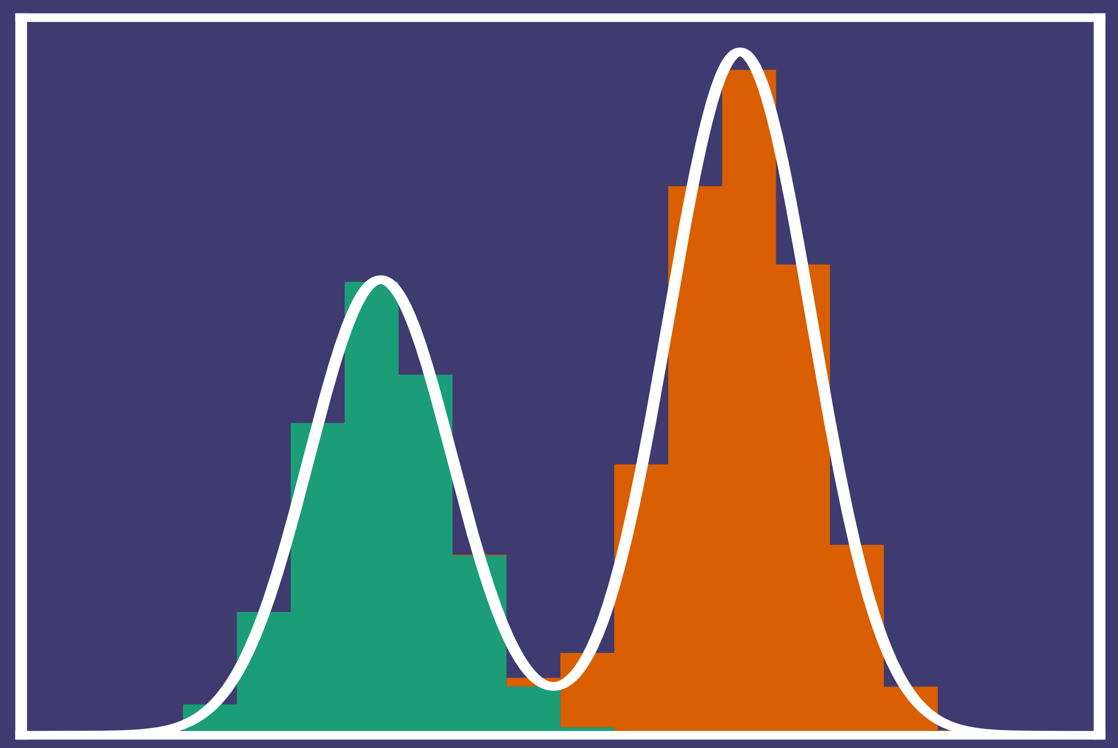
$$d\sigma \sim \text{pdf} \quad \times \quad |M(x)|^2 \quad \times \quad \text{phase space}$$

Phase space: increase unweighting efficiency

- Standard **VEGAS** approach → **fast** but **no correlations**
- Improve with **NN** → **correlations** but **unstable**
- Use **normalizing flows** → **correlations** and **stable**
- **Multi-channel** approach → split the integral
- Combine all (VEGAS, learned α_i, NF, symmetries,..) → **MadNIS** framework

[1707.00028, 1810.11509, 2001.05478, 2001.05486, 2001.10028, 2005.12719, 2009.07819, 2011.13445, 2109.11964, 2112.09145, 2212.06172, 2301.13562, 2309.12369, 2311.01548…..]

Kleiss, Pittau [hep-ph/9405257], Maltoni, Stelzer [hep-ph/0208156]



Multi-channel:
one map for each channel

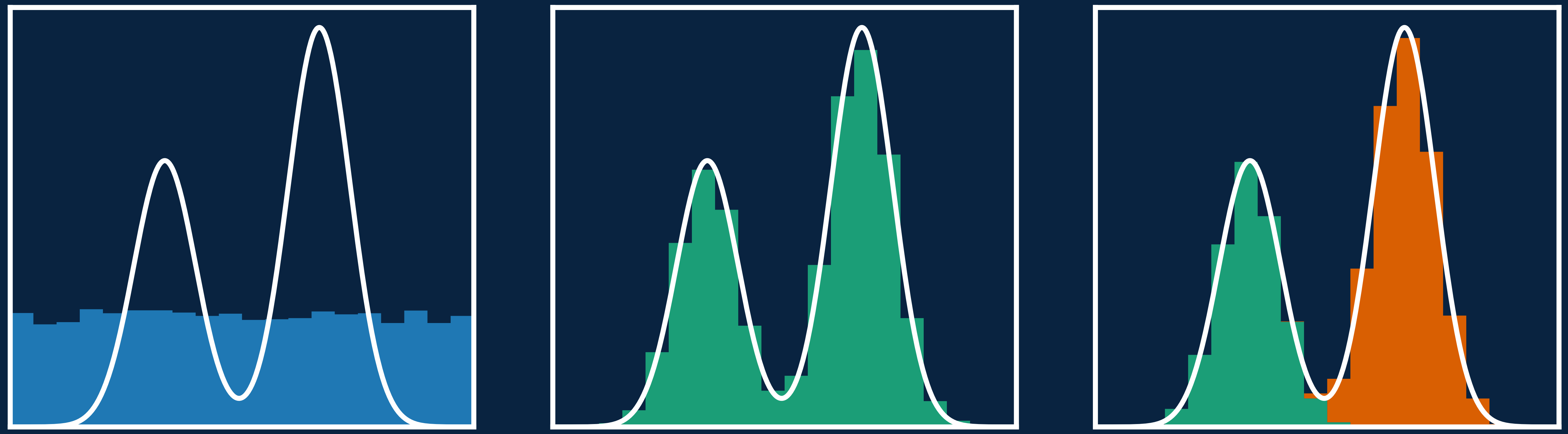
$$I = \sum_i \left\langle \alpha_i(x) \frac{f(x)}{g_i(x)} \right\rangle_{x \sim g_i(x)}$$

MadNIS

Neural Importance Sampling

Heimel, Huetsch, Maltoni, Mattelaer, Plehn, RW [2311.01548]
Heimel, RW, Butter, Isaacson, Krause, Maltoni, Mattelaer, Plehn [2212.06172]

MadNIS — Neural importance sampling



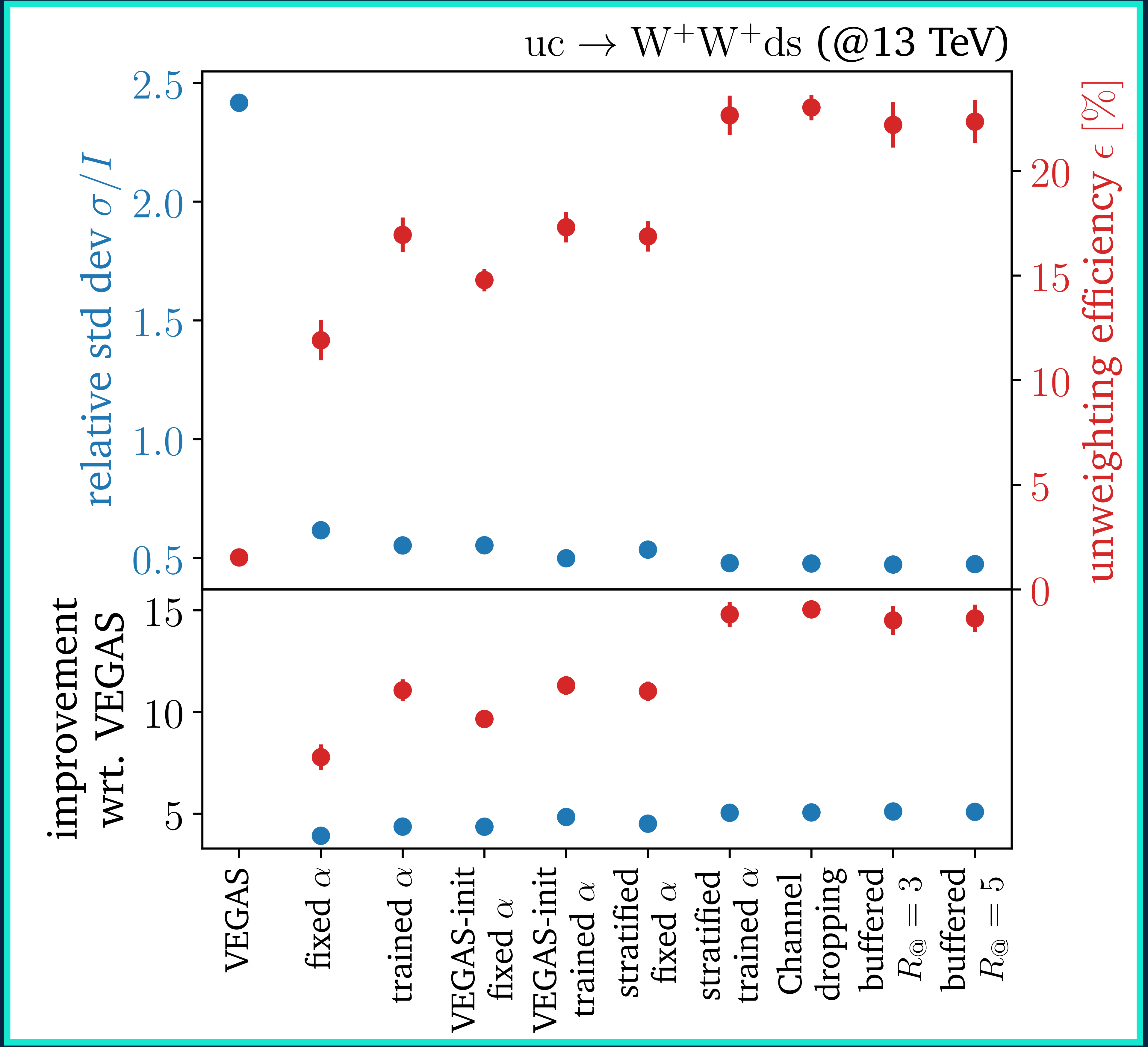
Flat sampling | Importance Sampling | Multi-channel

$$I = \sum_i \left\langle \alpha_i(x) \frac{f(x)}{g_i(x)} \right\rangle_{x \sim g_i(x)}$$

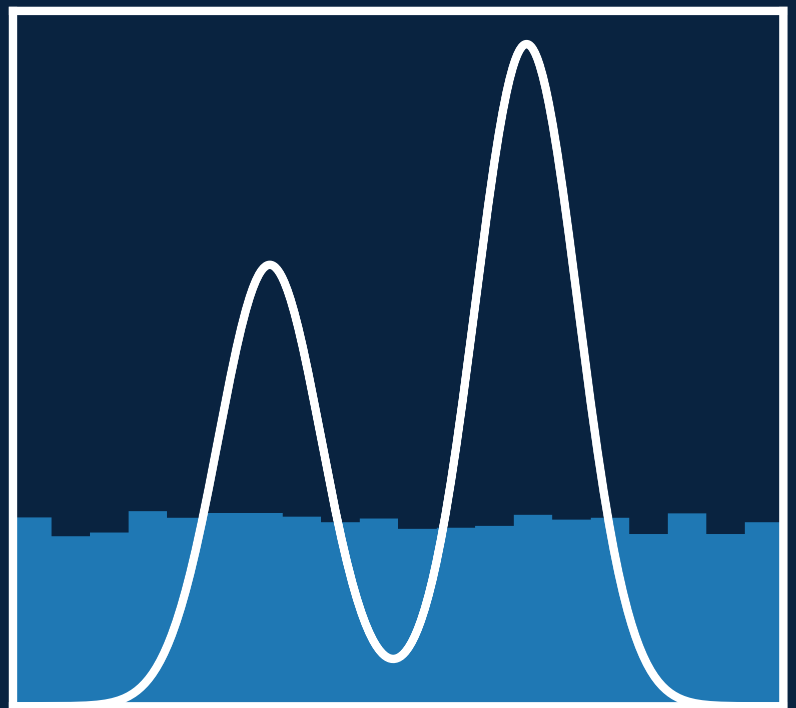
Parametrize with **NN** (for $\alpha_i(x)$)

Parametrize with **NF** (for $g_i(x)$)

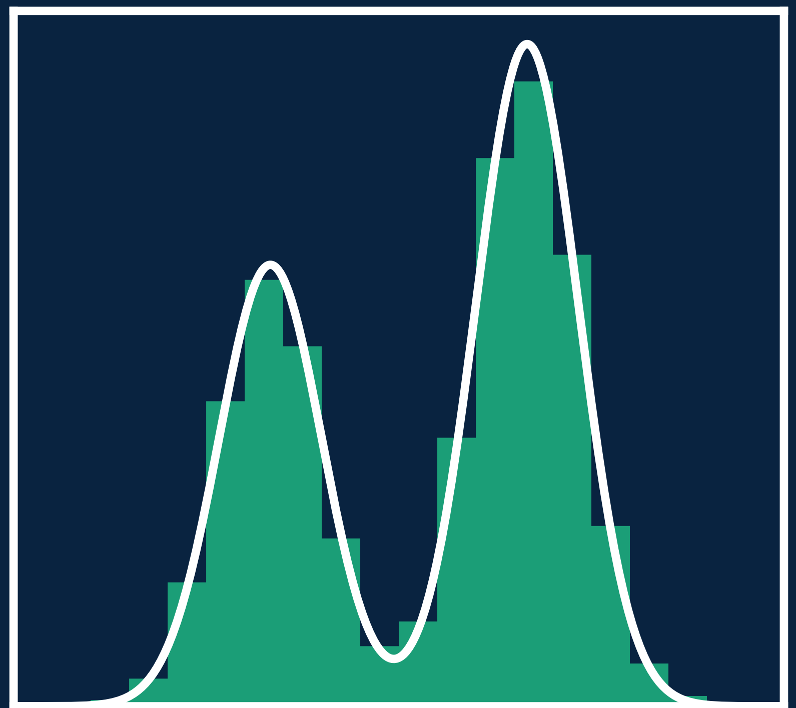
Heimel, Huetsch, Maltoni, Mattelear, Plehn, RW [2311.01548]



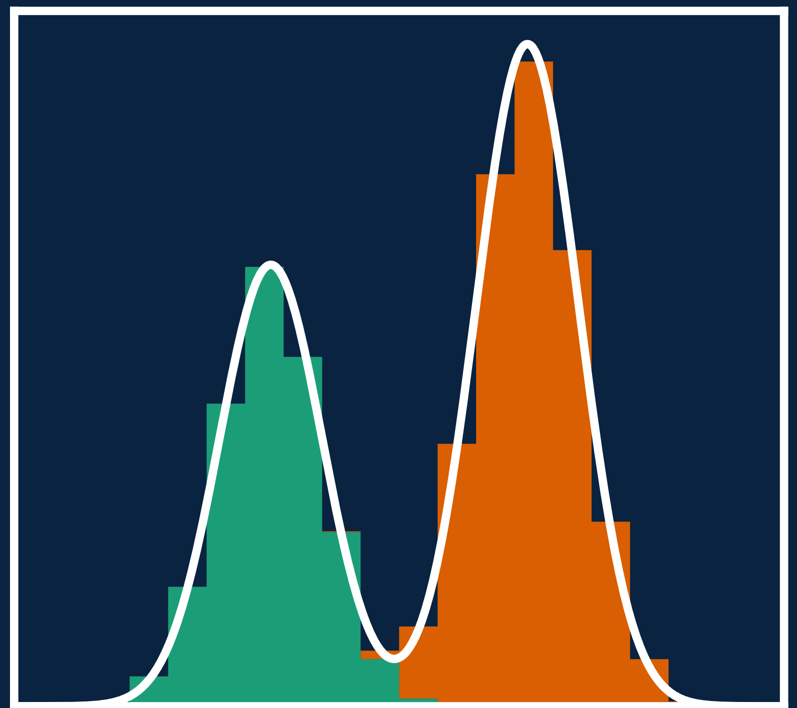
MadNIS — Neural importance sampling



Flat sampling



Importance Sampling



Multi-channel

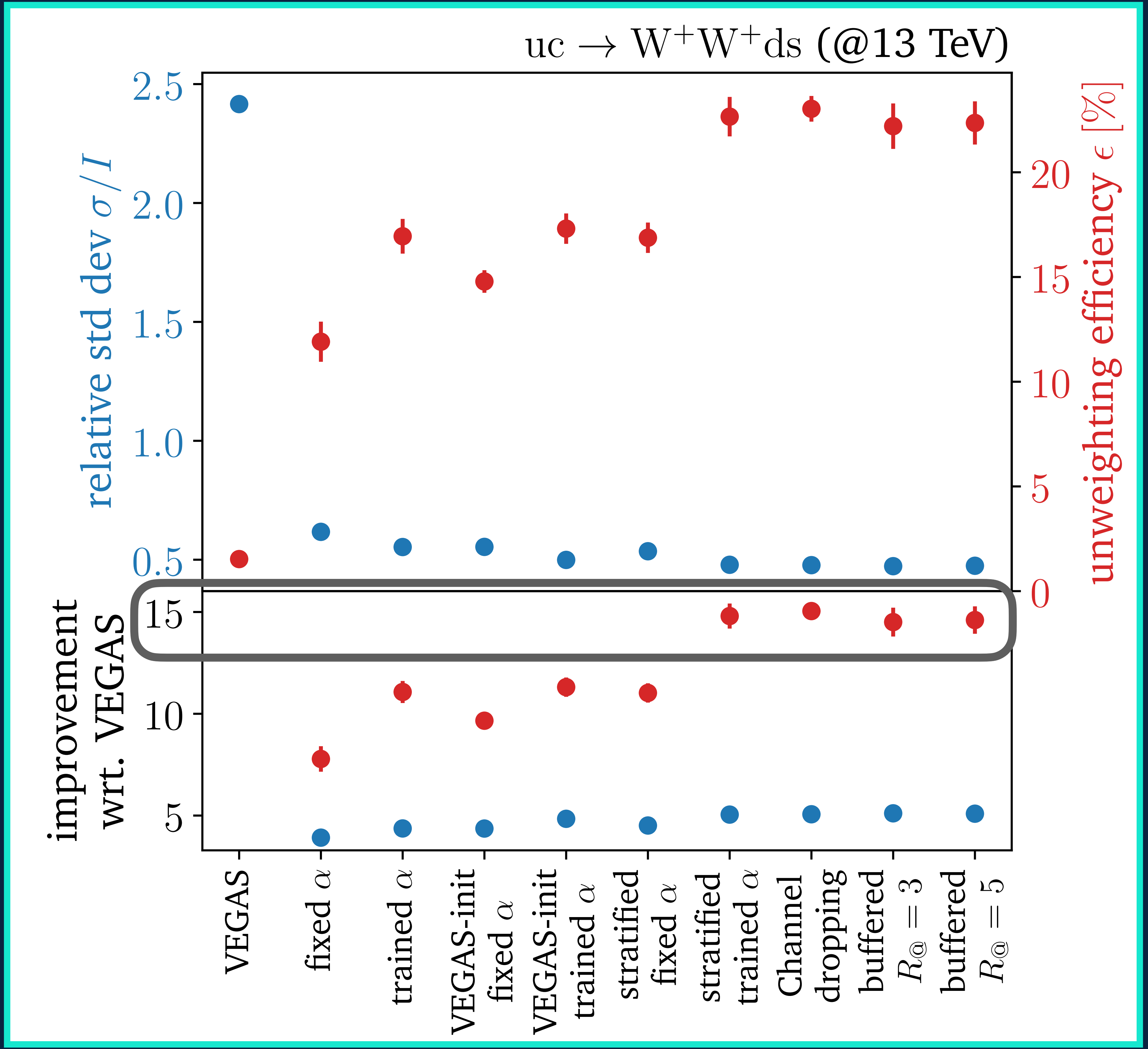
$$I = \sum_i \left\langle \alpha_i(x) \frac{f(x)}{g_i(x)} \right\rangle_{x \sim g_i(x)}$$

Parametrize with **NN**

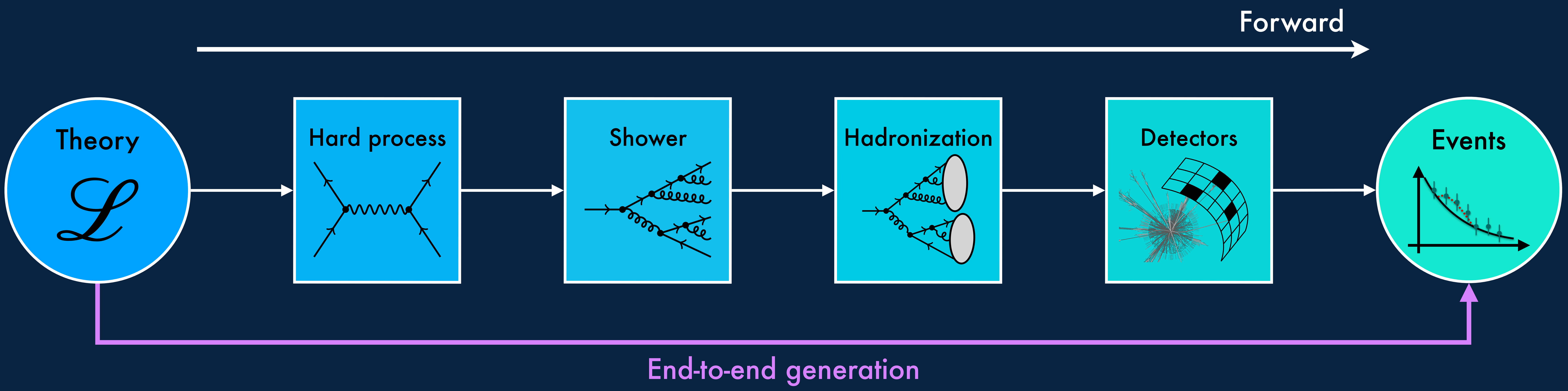
Parametrize with **NF**

→ Details in talk by **Theo Heimel**

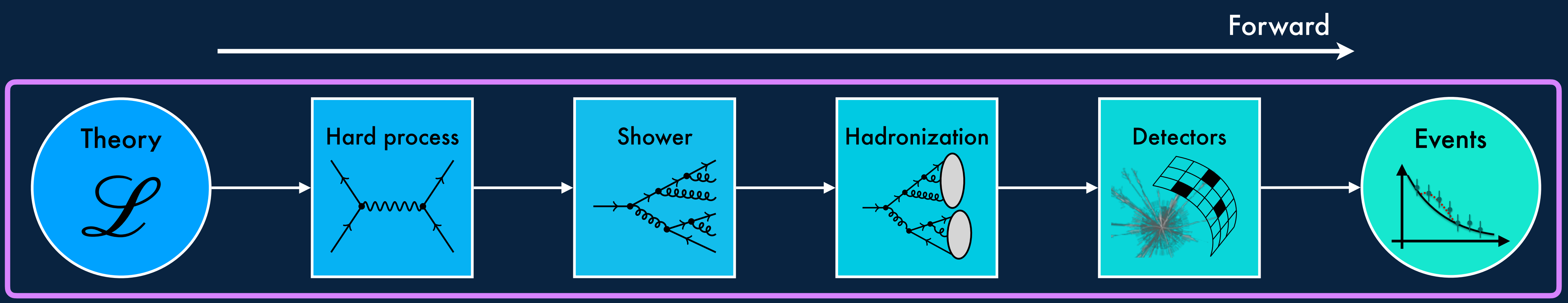
Heimel, Huetsch, Maltoni, Mattelear, Plehn, RW [2311.01548]



ML for forward simulations



ML for forward simulations

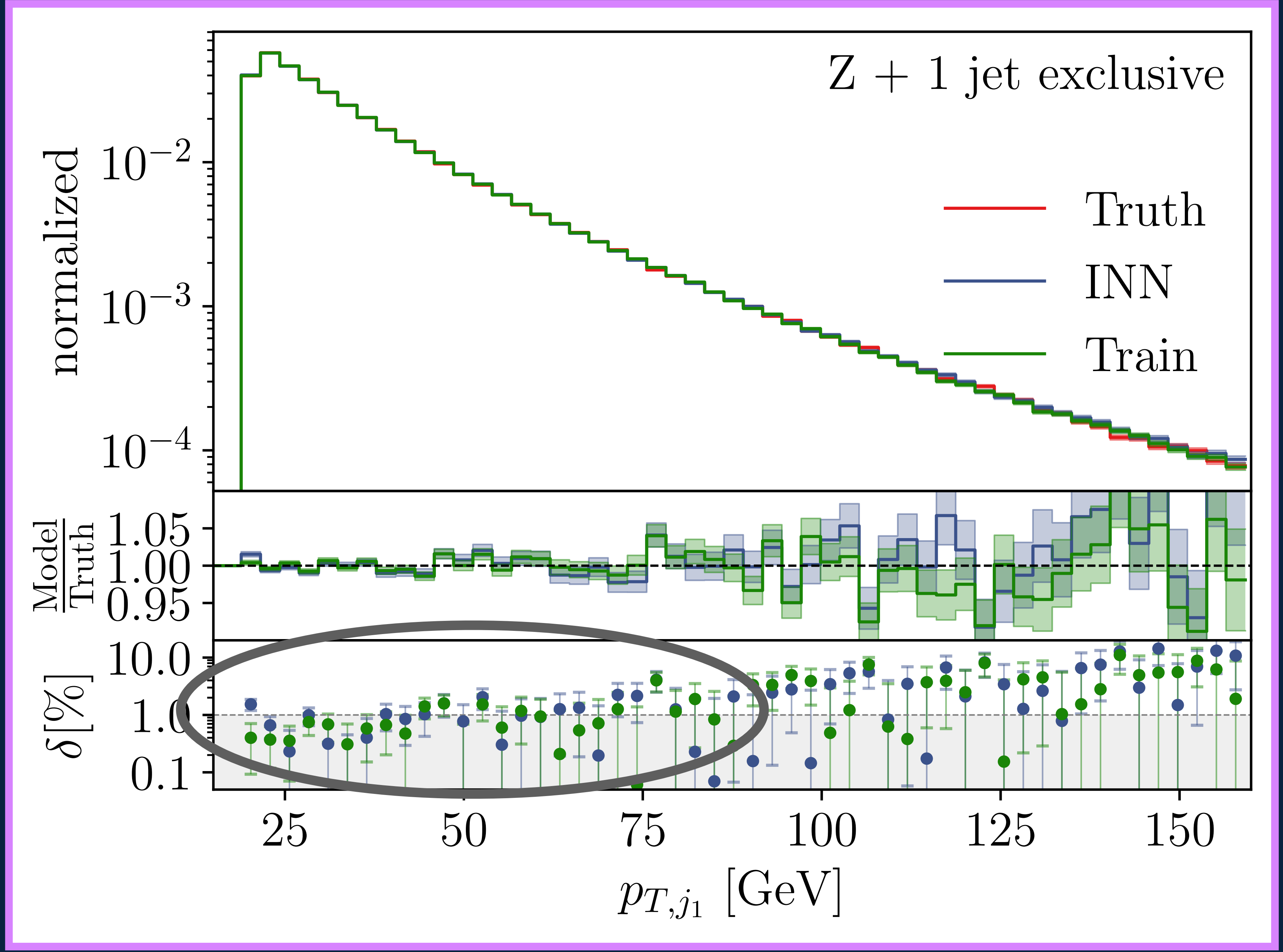


End-to-end generators learn multiple steps at once

Precision generation

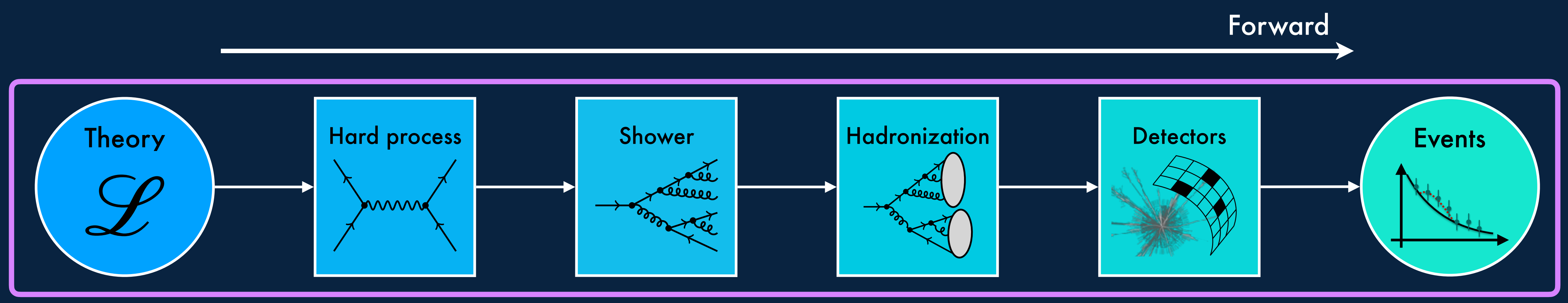
- First attempts based on **GANs and VAEs**
- Improved speed and efficiency with **Flows**

Butter, Heimel, Hummerich, Krebs, Plehn, Rousselot, Vent [2110.13632]



[1901.00875, 1901.05282, 1903.02433, 1907.03764, 1912.02748, 2001.11103, 2011.13445, 2101.08944, 2110.13632, 2211.13630, 2303.05376, 2305.07696, 2305.10475, 2305.16774, 2307.06836 .....]

ML for forward simulations

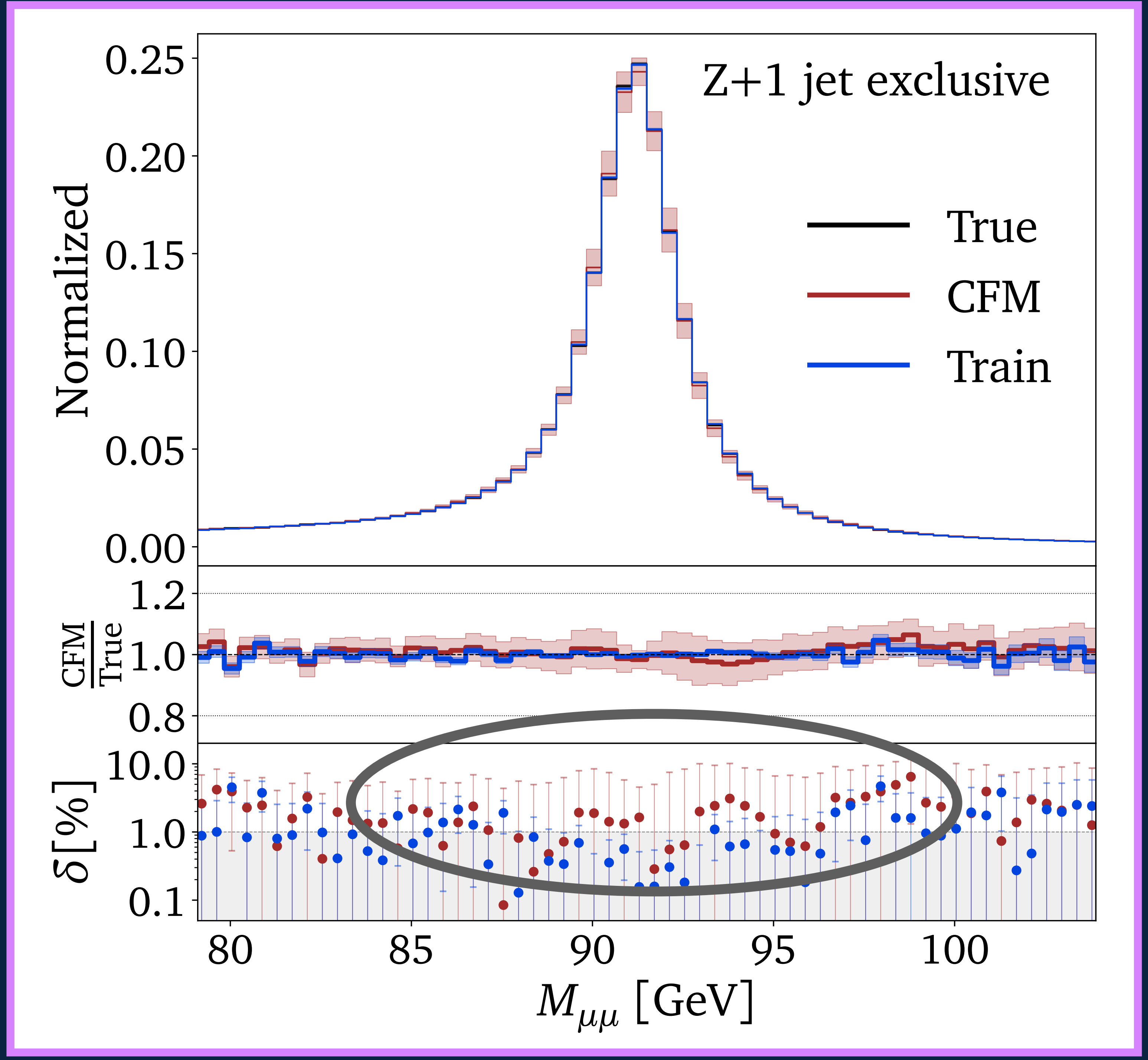


End-to-end generators learn multiple steps at once

Precision generation

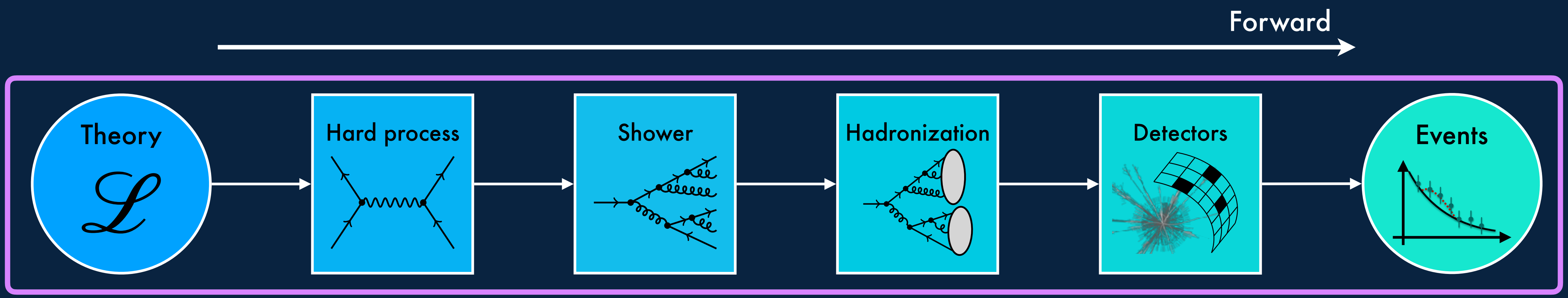
- First attempts based on **GANs and VAEs**
- Improved speed and efficiency with **Flows**
- High precision with **Diffusion** and **Transformer** models

Butter, Huetsch, Schweitzer, Plehn, Sorrenson, Spinner [2110.11377]



[1901.00875, 1901.05282, 1903.02433, 1907.03764, 1912.02748, 2001.11103, 2011.13445, 2101.08944, 2110.13632, 2211.13630, 2303.05376, 2305.07696, 2305.10475, 2305.16774, 2307.06836 .....]

ML for forward simulations



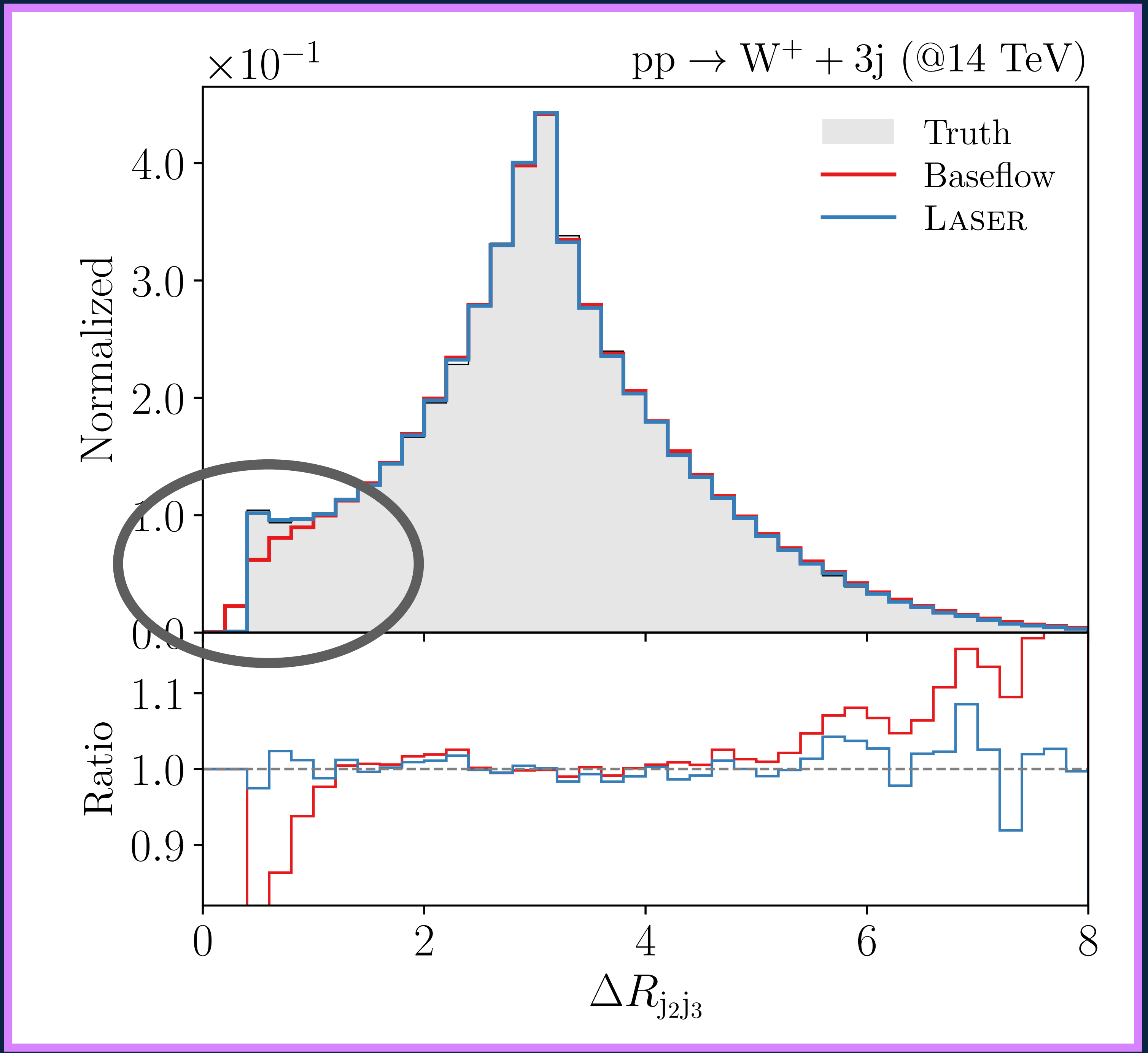
End-to-end generators learn multiple steps at once

Precision generation

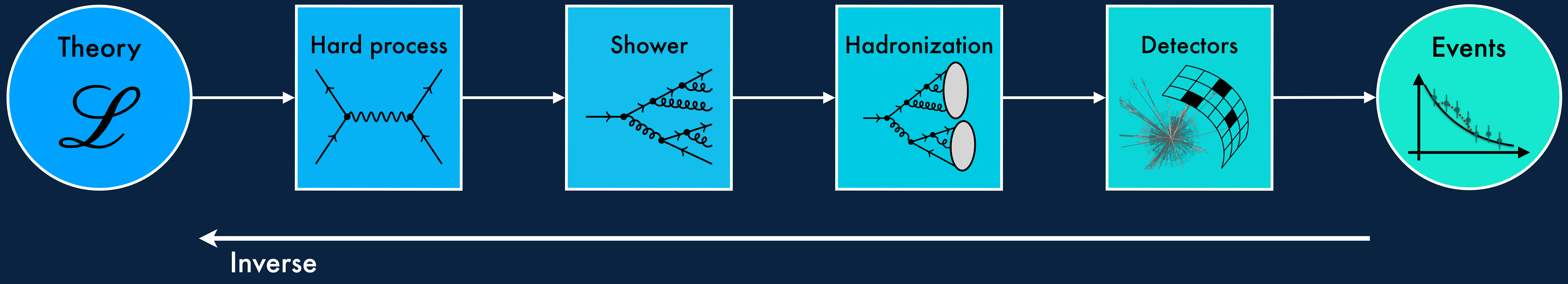
- First attempts based on **GANs and VAEs**
- Improved speed and efficiency with **Flows**
- High precision with **Diffusion** and **Transformer** models
- **Bayesian NN** and **classifiers** for full control

[1901.00875, 1901.05282, 1903.02433, 1907.03764, 1912.02748, 2001.11103, 2011.13445, 2101.08944, 2110.13632, 2211.13630, 2303.05376, 2305.07696, 2305.10475, 2305.16774, 2307.06836 .....]

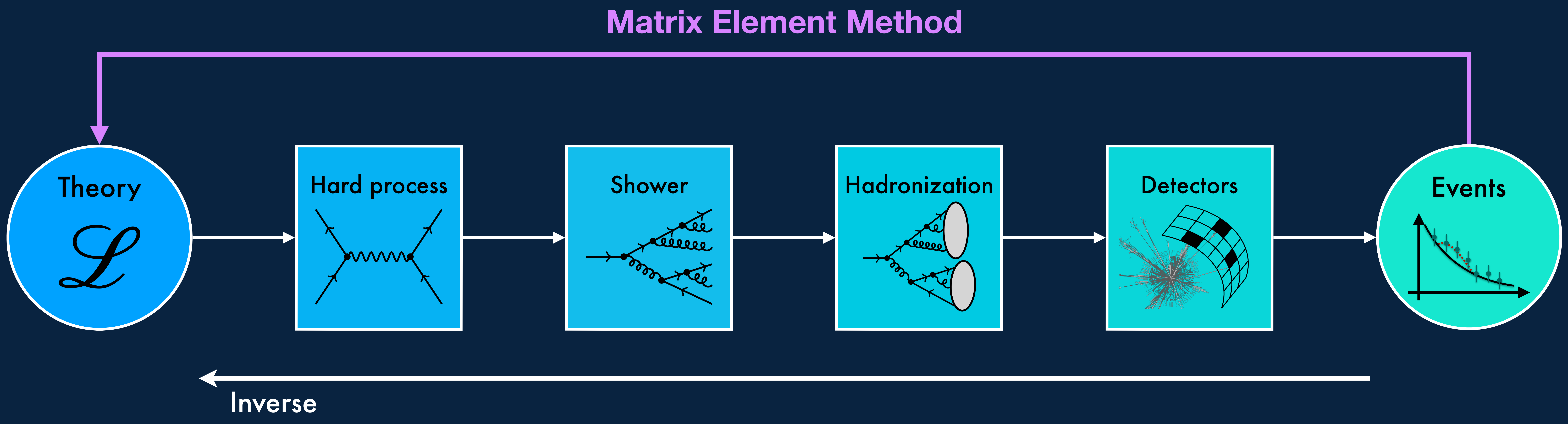
Nachman, RW [2305.07696]



ML for inverse simulations



ML for inverse simulations



Historically → Tevatron
Top mass: D0 (98’, 04’), CDF 06’, Fiedler et al. [1003.1316]
Single-top: Review [1710.10699]

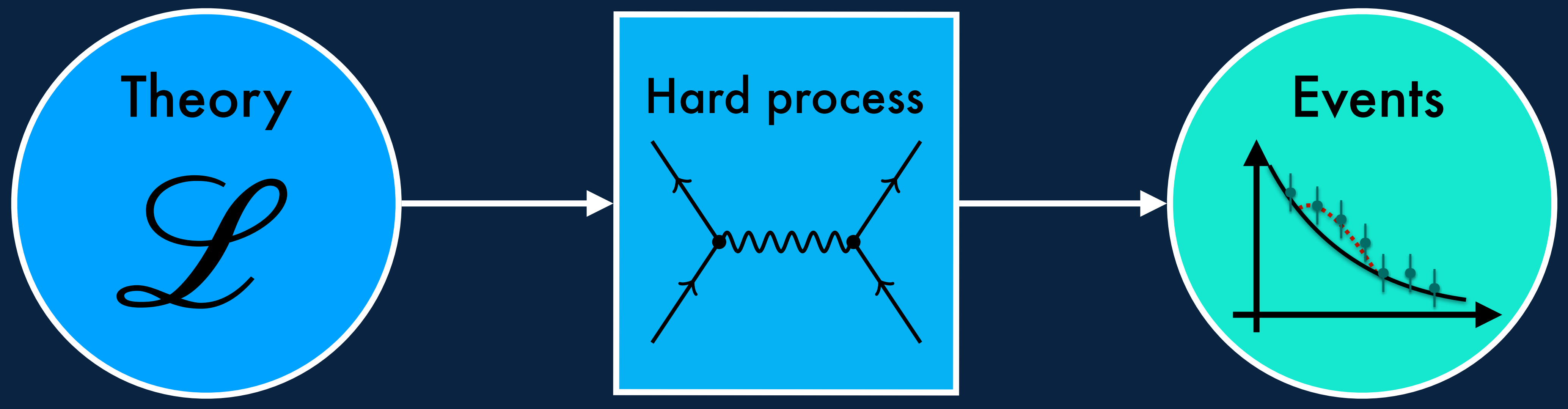
MEM-ML

Matrix Element Method

Heimel, Huetsch, RW, Plehn, Butter [2310.07752]

Butter, Heimel, Martini, Peitzsch, Plehn [2210.00019]

Matrix Element Method — Theory



Theory parameter α

Known from theory

Hard-scattering momenta x_{hard}

Likelihood from differential cross section

$$p(x_{\text{hard}}\,|\,\alpha) = \frac{1}{\sigma(\alpha)} \frac{\mathrm{d}\sigma(\alpha)}{\mathrm{d}x_{\text{hard}}}$$

Classical analysis

⊖ hand-crafted observables

⊖ binned data

→ not all information is used ☹

Matrix Element Method (MEM)

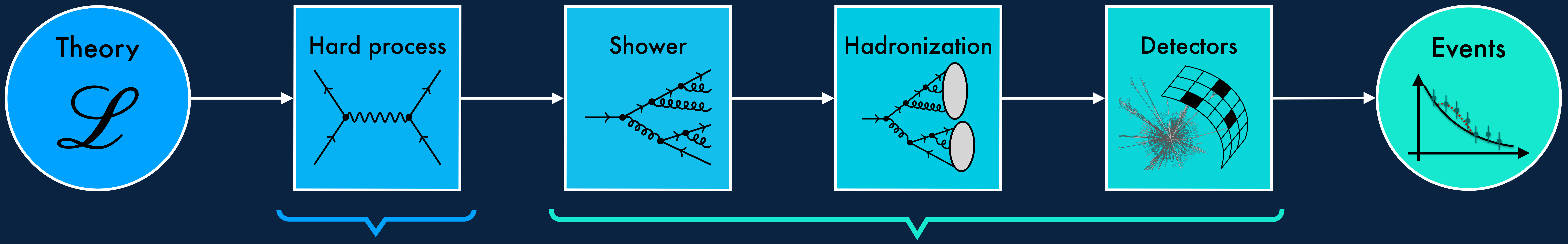
⊕ based on first principles

⊕ estimates uncertainties reliably

⊕ optimal use of information

→ perfect for processes with few events ☺

Matrix Element Method — Reality



Theory parameter
α

Known from theory

Likelihood intractable
→ parametrize with **NF**

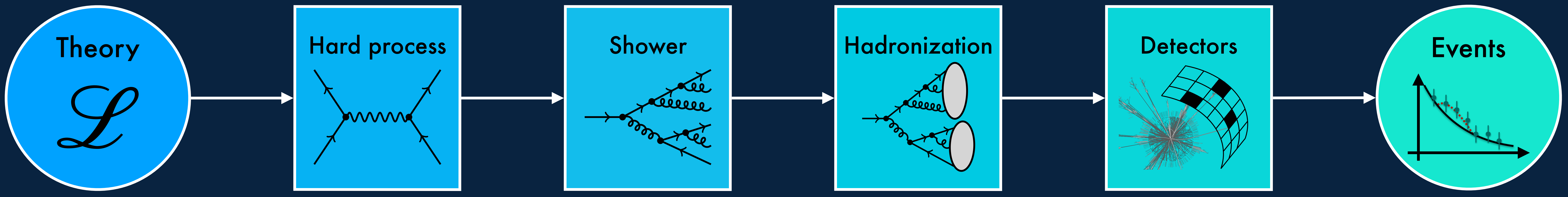
Reconstructed momenta
x_{reco}

parametrize with **NN**

MEM master formula:

$$p(x_{\text{reco}}\,|\,\alpha) = \int \mathrm{d}x_{\text{hard}}\; p(x_{\text{hard}}\,|\,\alpha)\, p(x_{\text{reco}}\,|\,x_{\text{hard}})\; \epsilon(x_{\text{hard}})$$

Matrix Element Method — Reality



Theory parameter
α

Known from theory

Likelihood intractable
→ parametrize with **NF**

Reconstructed momenta
x_{reco}

parametrize with **NN**

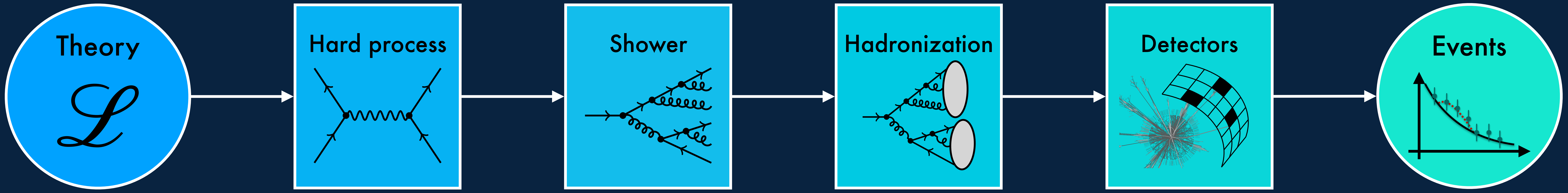
MEM master formula: $$p(x_{\text{reco}}\,|\,\alpha) = \int \mathrm{d}x_{\text{hard}}\; p(x_{\text{hard}}\,|\,\alpha)\, p(x_{\text{reco}}\,|\,x_{\text{hard}})\, \epsilon(x_{\text{hard}})$$

⟶ Details in talk by **Nathan Huetsch**

Summary and Outlook

Take-home message

- ML beneficial in **every step** in the simulation chain
- We find both **proof-of-concepts** as well as established use cases **(→ MadNIS)**
- Interesting **interplay** between **HEP** and **ML**

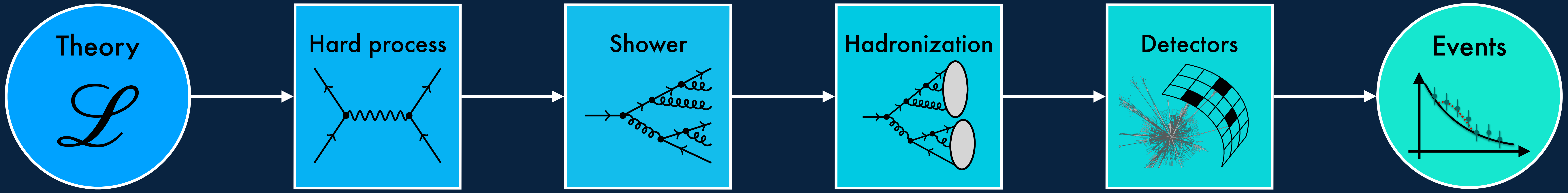


Summary and Outlook

Take-home message

- ML beneficial in **every step** in the simulation chain
- We find both **proof-of-concepts** as well as established use cases **(→ MadNIS)**
- Interesting **interplay** between **HEP** and **ML**

→ HEP simulations provide **~infinite data** for ML



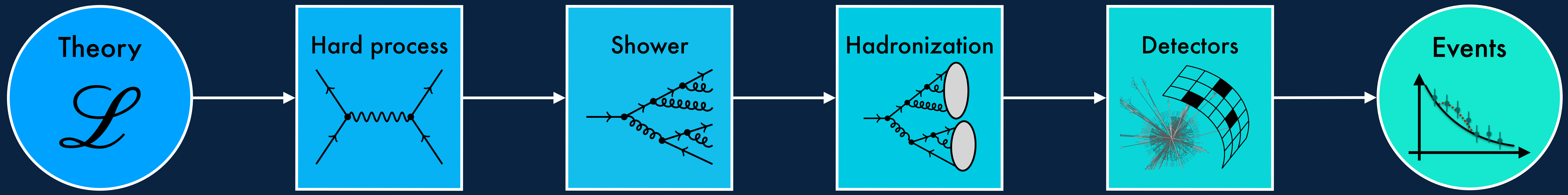
Summary and Outlook

Take-home message

- ML beneficial in **every step** in the simulation chain
- We find both **proof-of-concepts** as well as established use cases **(→ MadNIS)**
- Interesting **interplay** between **HEP** and **ML**

→ HEP simulations provide **~infinite data** for ML

→ HEP requirements (**precision, symmertries,…**) **different** than industry applications



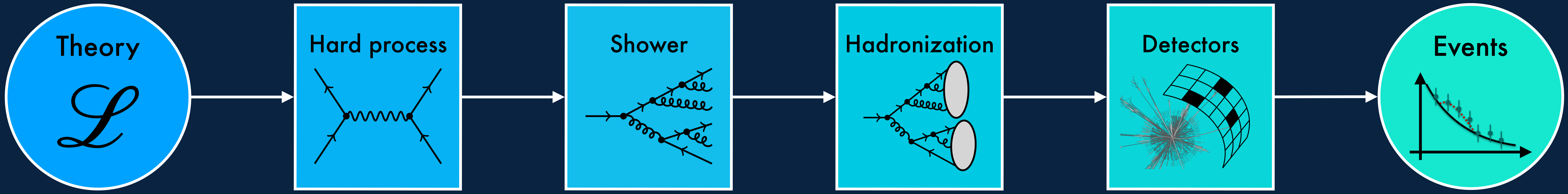
Summary and Outlook

Take-home message

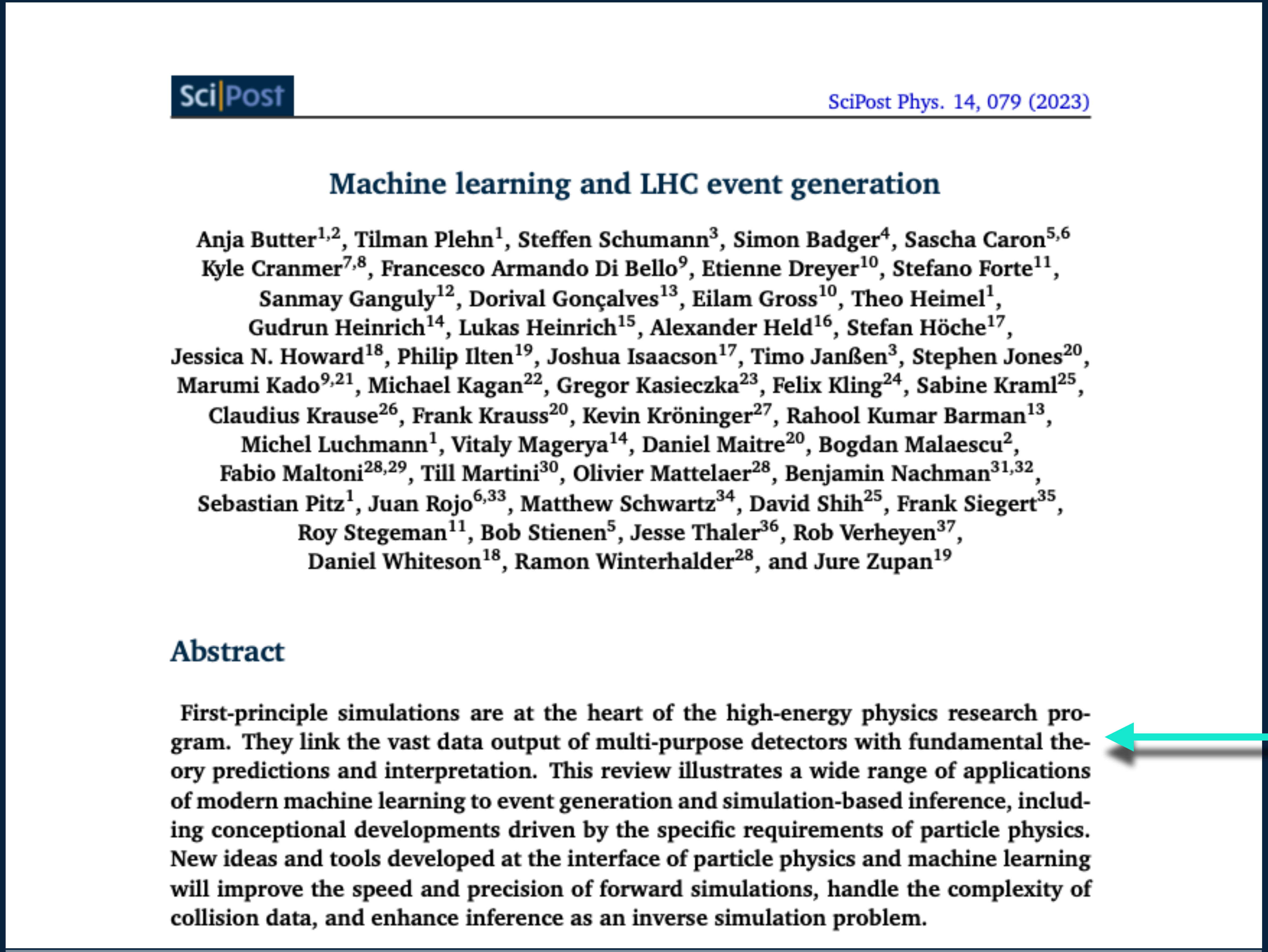
- ML beneficial in **every step** in the simulation chain
- We find both **proof-of-concepts** as well as established use cases **(→ MadNIS)**
- Interesting **interplay** between **HEP** and **ML**
  - → HEP simulations provide **~infinite data** for ML
  - → HEP requirements (**precision, symmertries,…**) **different** than industry applications

Future tasks

- **Full integration** of ML-based simulations into standard tools → **MadGraph,….**
- Make everything run on the **GPU and differentiable** (MadJax - Heinrich et al. [2203.00057])
- Further foster collaboration between **theory, experiment,** and **ML** community



Summary and Outlook



SciPost — SciPost Phys. 14, 079 (2023)

Machine learning and LHC event generation

Anja Butter[1,2], Tilman Plehn[1], Steffen Schumann[3], Simon Badger[4], Sascha Caron[5,6] Kyle Cranmer[7,8], Francesco Armando Di Bello[9], Etienne Dreyer[10], Stefano Forte[11], Sanmay Ganguly[12], Dorival Gonçalves[13], Eilam Gross[10], Theo Heimel[1], Gudrun Heinrich[14], Lukas Heinrich[15], Alexander Held[16], Stefan Höche[17], Jessica N. Howard[18], Philip Ilten[19], Joshua Isaacson[17], Timo Janßen[3], Stephen Jones[20], Marumi Kado[9,21], Michael Kagan[22], Gregor Kasieczka[23], Felix Kling[24], Sabine Kraml[25], Claudius Krause[26], Frank Krauss[20], Kevin Kröninger[27], Rahool Kumar Barman[13], Michel Luchmann[1], Vitaly Magerya[14], Daniel Maitre[20], Bogdan Malaescu[2], Fabio Maltoni[28,29], Till Martini[30], Olivier Mattelaer[28], Benjamin Nachman[31,32], Sebastian Pitz[1], Juan Rojo[6,33], Matthew Schwartz[34], David Shih[25], Frank Siegert[35], Roy Stegeman[11], Bob Stienen[5], Jesse Thaler[36], Rob Verheyen[37], Daniel Whiteson[18], Ramon Winterhalder[28], and Jure Zupan[19]

Abstract

First-principle simulations are at the heart of the high-energy physics research program. They link the vast data output of multi-purpose detectors with fundamental theory predictions and interpretation. This review illustrates a wide range of applications of modern machine learning to event generation and simulation-based inference, including conceptional developments driven by the specific requirements of particle physics. New ideas and tools developed at the interface of particle physics and machine learning will improve the speed and precision of forward simulations, handle the complexity of collision data, and enhance inference as an inverse simulation problem.

Future tasks

- **Full integration** of ML-based simulations into standard tools → **MadGraph,….**
- Make everything run on the **GPU and differentiable** (MadJax - Heinrich et al. [2203.00057])
- Further foster collaboration between **theory, experiment,** and **ML** community
- More details in our **Snowmass report**