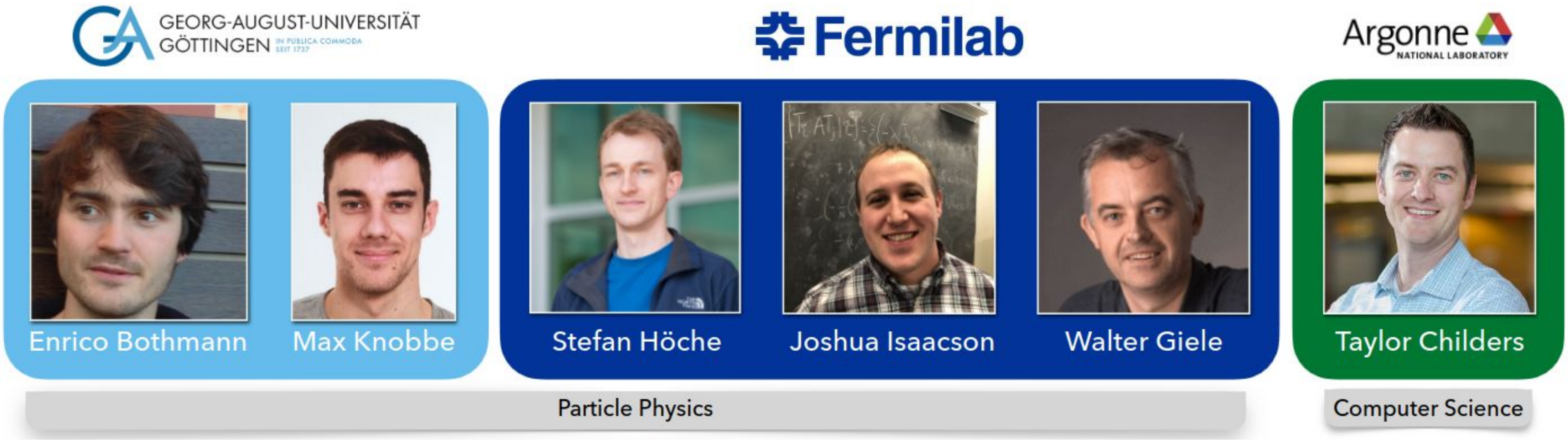
Fermilab U.S. DEPARTMENT OF ENERGY Office of Science

Towards Precision Calculations on Modern Computers

Joshua Isaacson
In Collaboration With: Enrico Bothmann, Taylor Childers, Walter Giele, Stefan Höche, and Max Knobbe
Arxiv: 2106.06507, 2302.10449, 2309.13154, 2311.06198
LoopFest 2024
22 May 2024

The Team

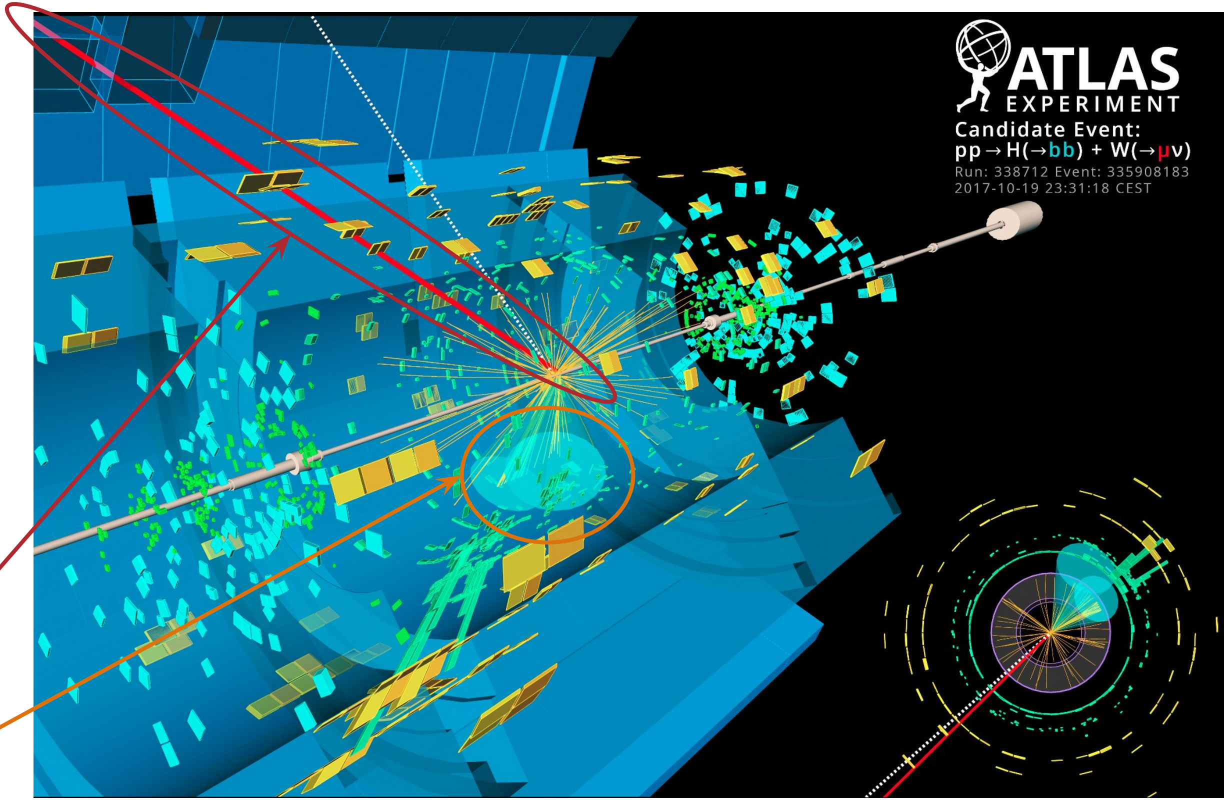

GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN
Fermilab
Argonne
NATIONAL LABORATORY
Enrico Bothmann
Max Knobbe
Stefan Höche
Joshua Isaacson
Walter Giele
Taylor Childers
Particle Physics
Computer Science

Very High-Level Experimental View of an Event

The detector measures:

- Charged particle tracks
- Energy deposits in different parts of the detector

Events saved if deemed interesting through the use of a trigger

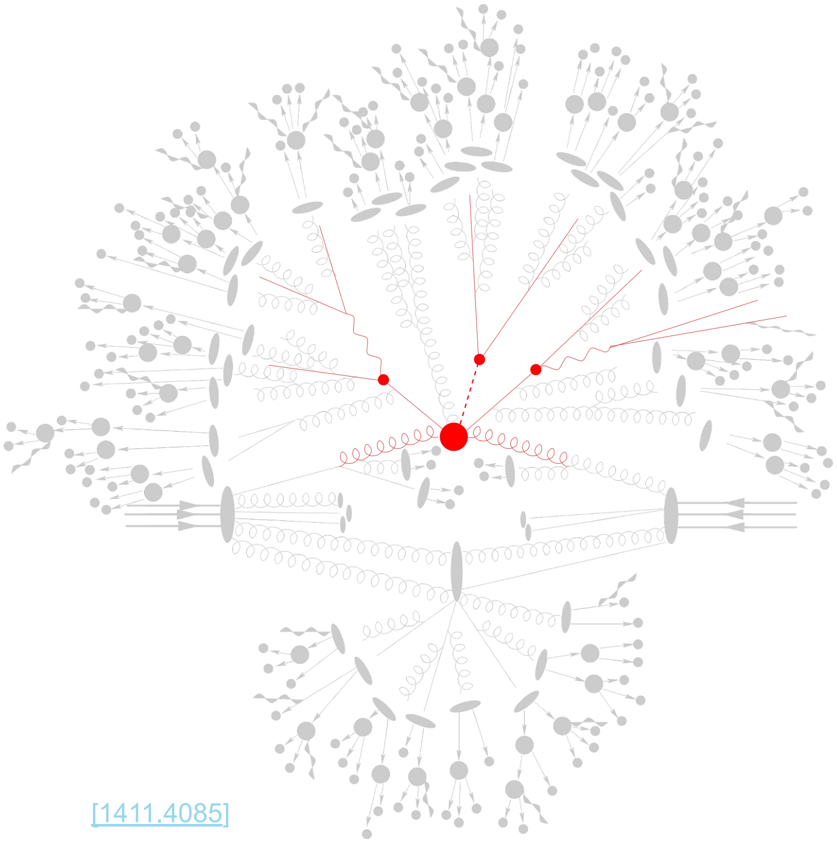


muon

b-jets

Credit: ATLAS

Theorist View of an Event

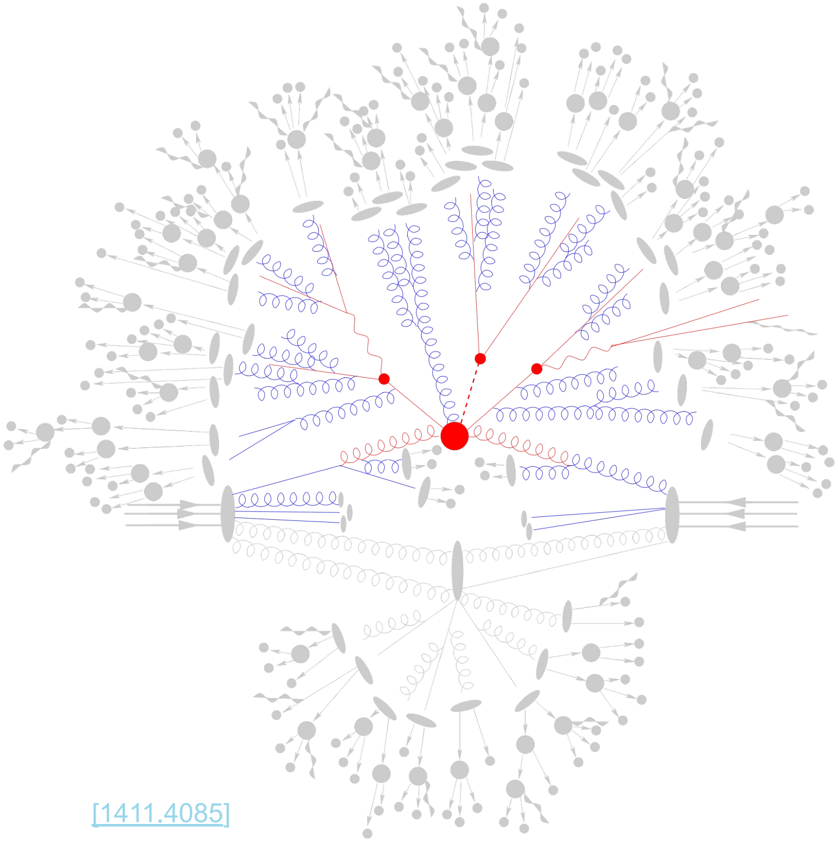


[1411.4085]

➔ **Initial proton beams:**
- Incoming partons determined from Parton Distribution Functions (PDFs)

➔ **Hard Scattering**
- Fixed order perturbative calculation: Leading order (LO), Next-to-leading order (NLO), etc.

Theorist View of an Event



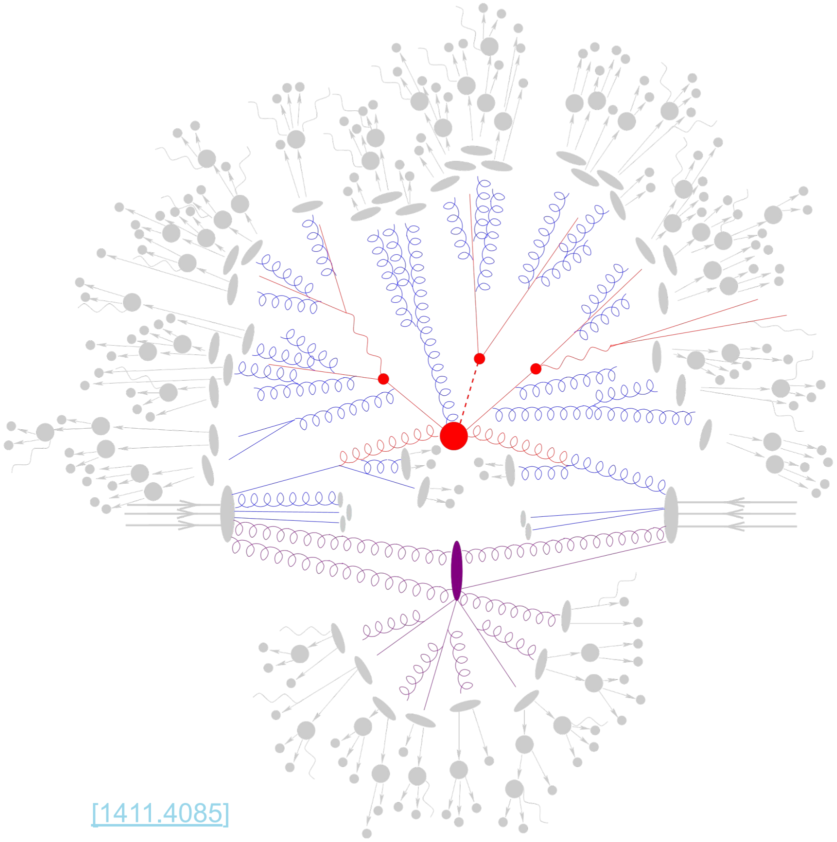
[1411.4085]

→ **Initial proton beams:**
- Incoming partons determined from Parton Distribution Functions (PDFs)

→ **Hard Scattering**
- Fixed order perturbative calculation: Leading order (LO), Next-to-leading order (NLO), etc.

→ **Soft gluon and parton radiation:**
- Analytic Resummation
- Numerical Resummation (i.e. Parton Showers)

Theorist View of an Event



[1411.4085]

→ **Initial proton beams:**
- Incoming partons determined from Parton Distribution Functions (PDFs)

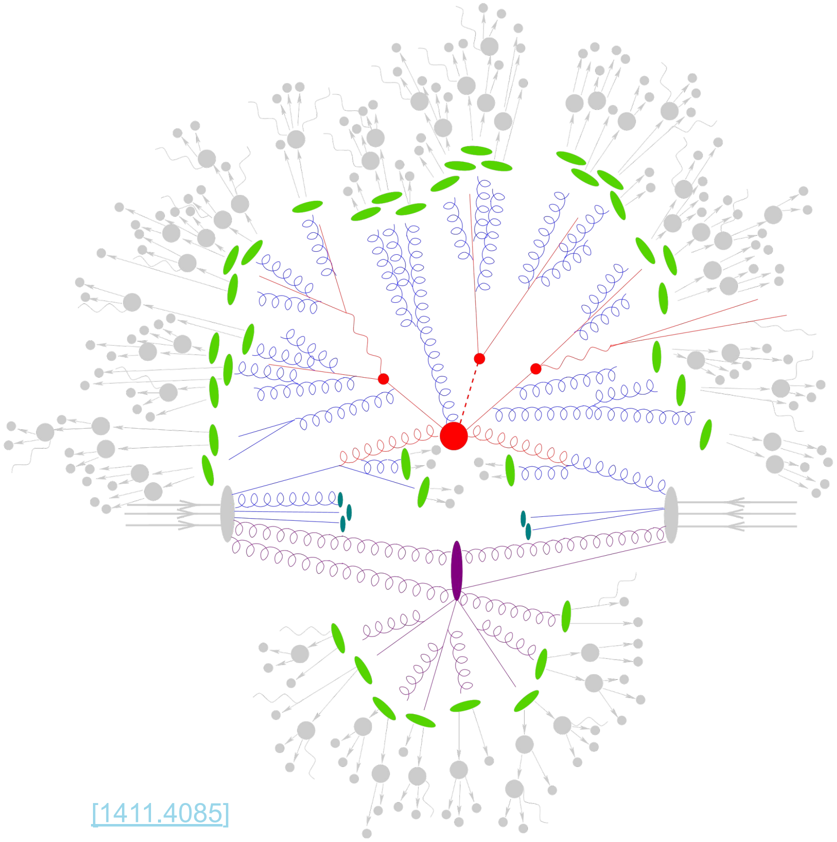
→ **Hard Scattering**
- Fixed order perturbative calculation: Leading order (LO), Next-to-leading order (NLO), etc.

→ **Soft gluon and parton radiation:**
- Analytic Resummation
- Numerical Resummation (i.e. Parton Showers)

→ **Multiple Parton Interactions:**
- Interactions between beam remnants

Fermilab

Theorist View of an Event



[1411.4085]

→ **Initial proton beams:**
- Incoming partons determined from Parton Distribution Functions (PDFs)

→ **Hard Scattering**
- Fixed order perturbative calculation: Leading order (LO), Next-to-leading order (NLO), etc.

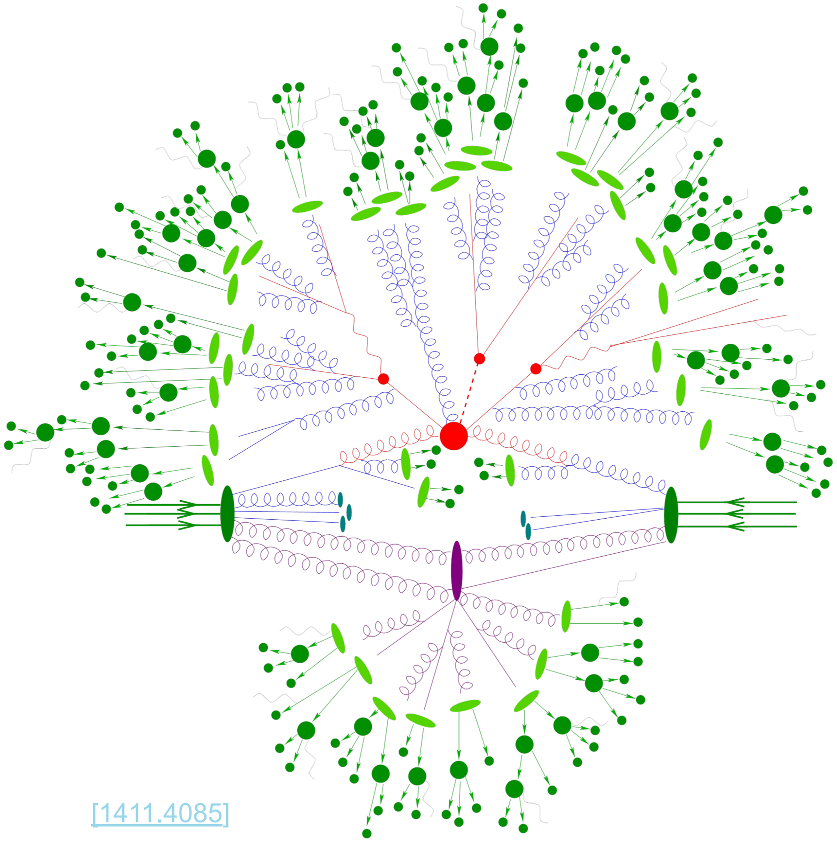
→ **Soft gluon and parton radiation:**
- Analytic Resummation
- Numerical Resummation (i.e. Parton Showers)

→ **Multiple Parton Interactions:**
- Interactions between beam remnants

→ **Hadronization:**
- Conversion of quarks and gluons into hadrons

Fermilab

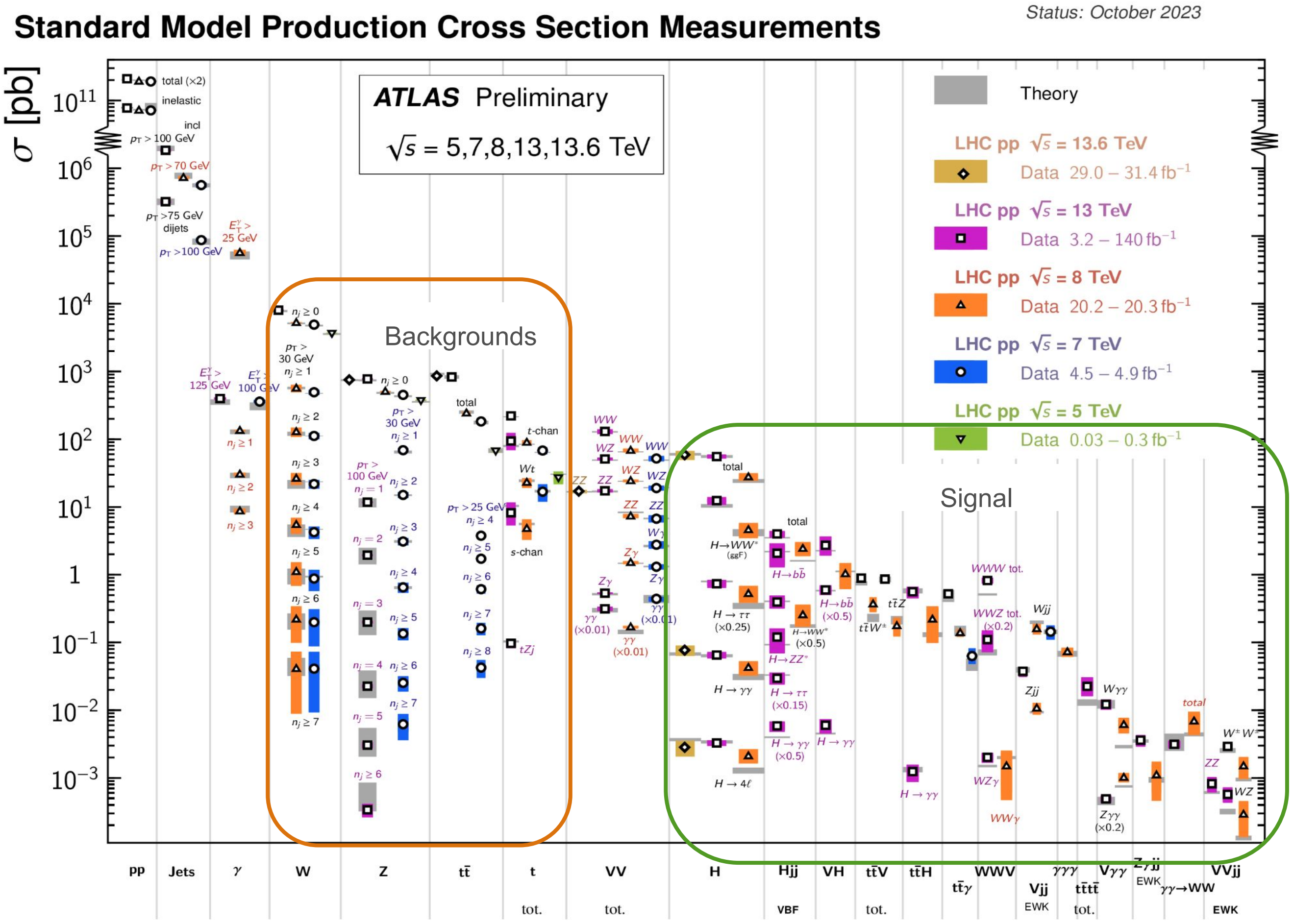
Theorist View of an Event



[1411.4085]

- Initial proton beams:
  - Incoming partons determined from Parton Distribution Functions (PDFs)
- Hard Scattering
  - Fixed order perturbative calculation: Leading order (LO), Next-to-leading order (NLO), etc.
- Soft gluon and parton radiation:
  - Analytic Resummation
  - Numerical Resummation (i.e. Parton Showers)
- Multiple Parton Interactions:
  - Interactions between beam remnants
- Hadronization:
  - Conversion of quarks and gluons into hadrons
- Excited Hadron Decays

Computational Challenges



[ATL-PHYS-PUB-2023-039]

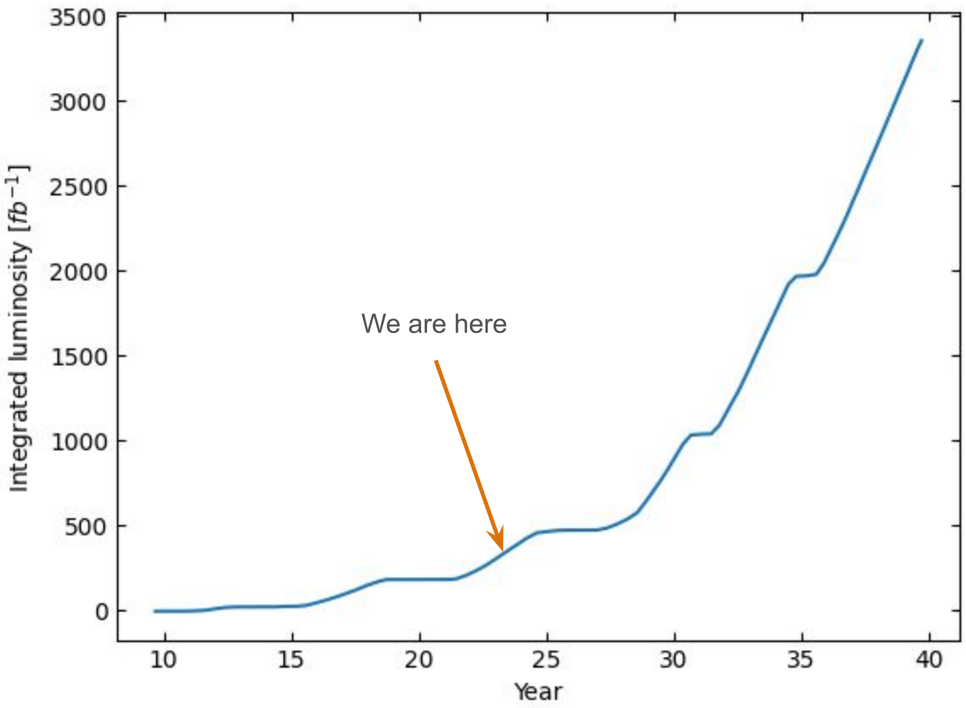
Fermilab

Computational Challenges

- Number of events given by:

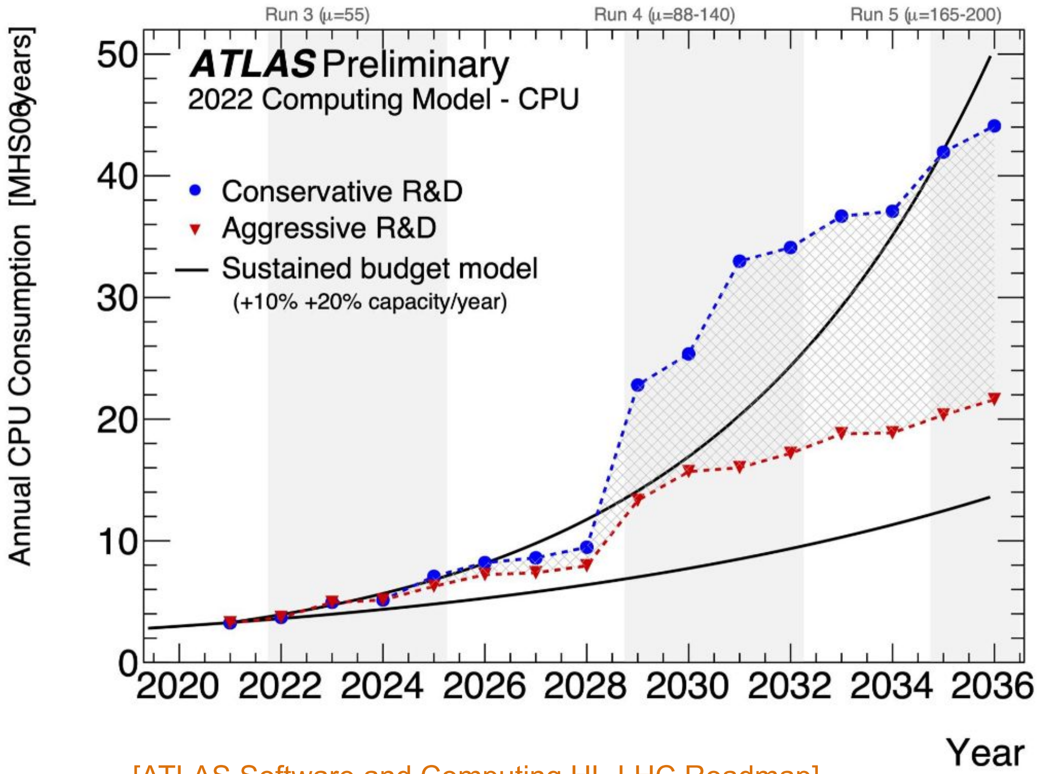
  (#Events) = (Luminosity) x (Cross Section)

- ATLAS will require about 330 billion V+jet events for the HL-LHC per event generator [2112.09588]
- Estimated to cost roughly **200,000 CPU-years** to generate with current tools (4 months on Frontier)
- This is simply unaffordable



Reproduced from:Rep. Prog. Phys. 85 046201

Computational Challenges



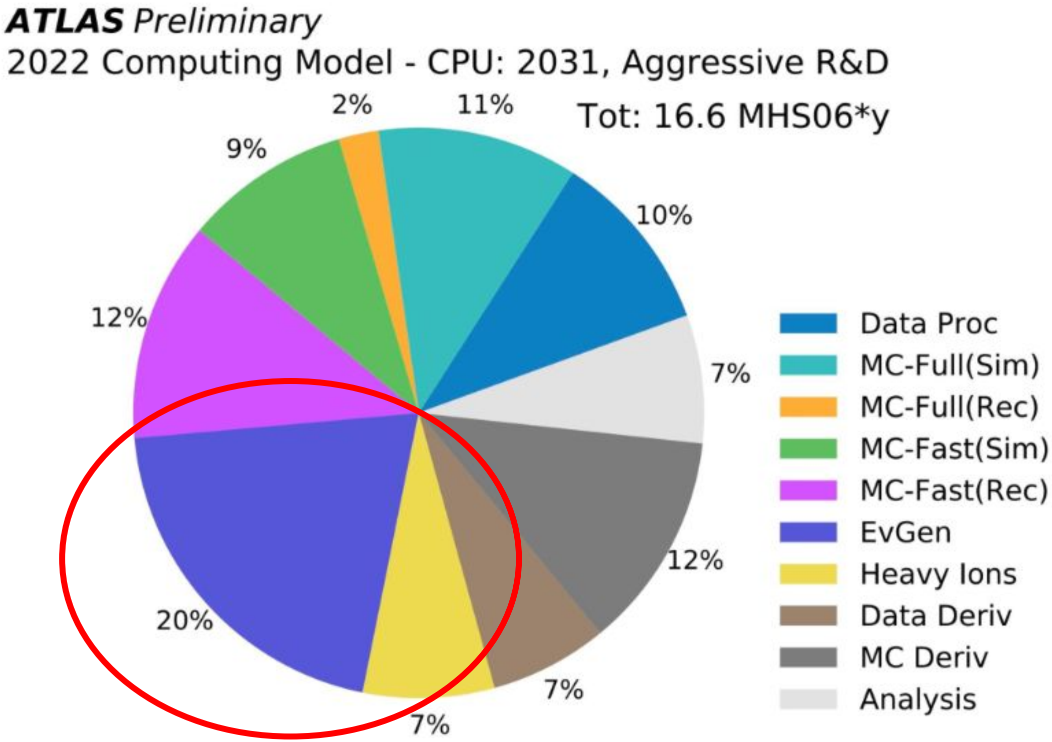
[ATLAS Software and Computing HL-LHC Roadmap]

- Current estimates require significant R&D on event generation

Computational Challenges

- Event generation significant component
- 70% of time per event spent on tree-level matrix element and momentum generation

[Bothmann, et. al.: 2209.00843]

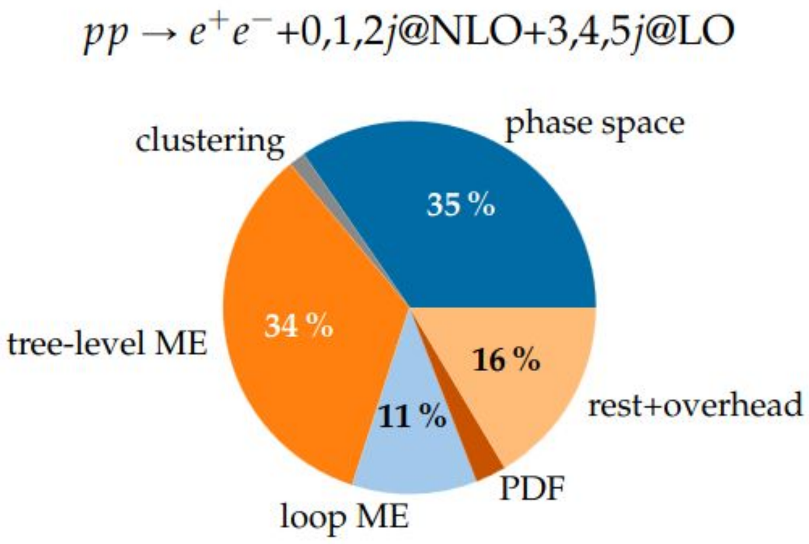


[ATLAS Software and Computing HL-LHC Roadmap]

Computational Challenges

- Event generation significant component
- 70% of time per event spent on tree-level matrix element and momentum generation

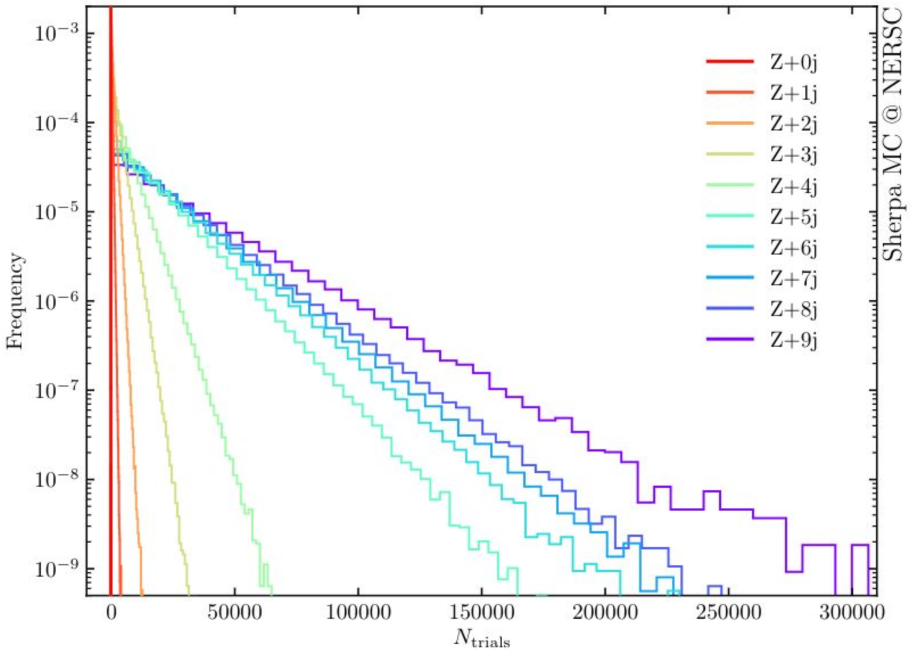
[Bothmann, et. al.: 2209.00843]



[Bothmann, et. al.: 2209.00843]

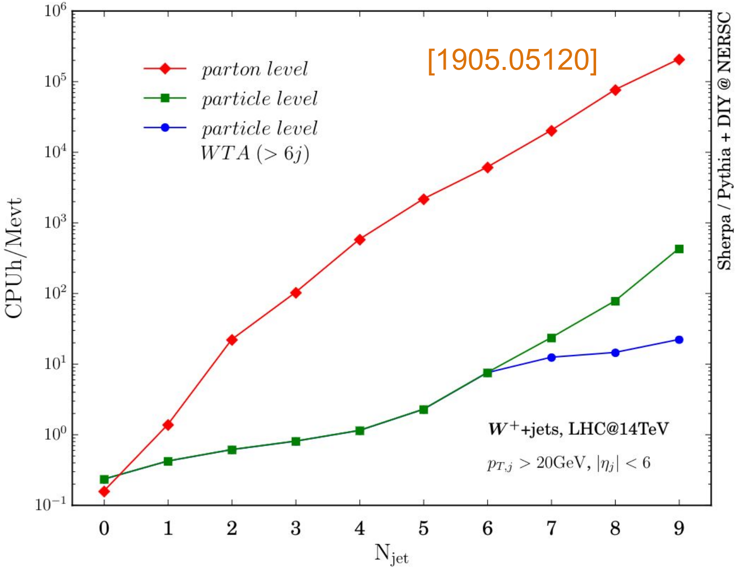
Computational Challenges

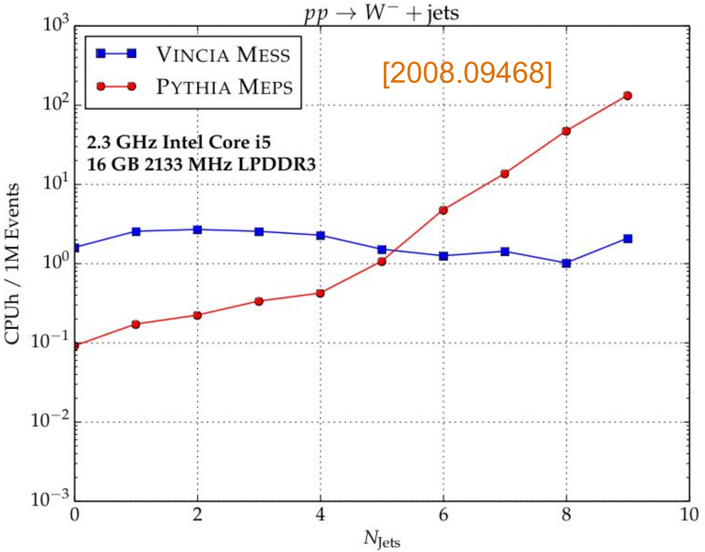
- Main Issues:
  - Low unweighting efficiencies
  - Expensive ME for high multiplicities
  - Inefficient phase space



[Höche, Prestel, Schulz: 1905.05120]

Computational Challenges





- Hard scattering more expensive than parton shower [Höche, Prestel, Schulz: 1905.05120]
- Computational complexity of merging ME & PS can be made linear with sector showers [Brooks, Preuss: 2008.09468]

Can we speed up event generators?

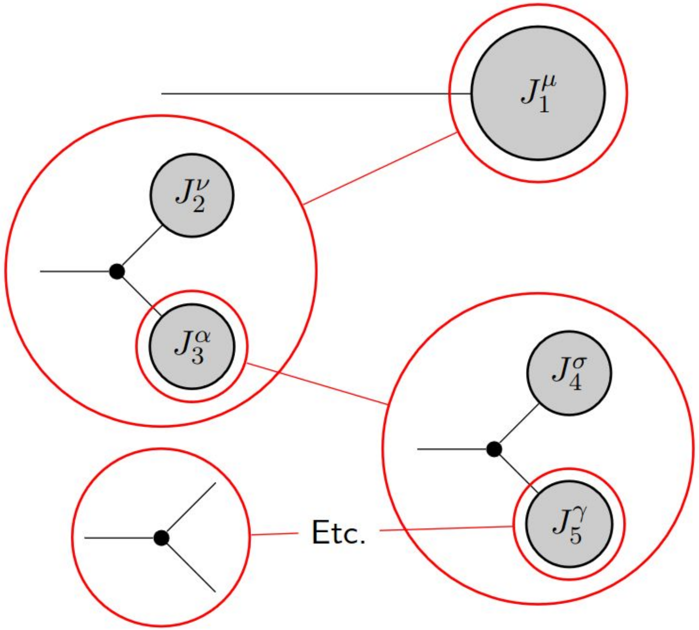
Recursive Matrix Element Evaluation

[Berends, Giele: Nucl. Phys. B306 (1988), 759]

$$\mathcal{J}_\alpha(\pi) = P_\alpha(\pi) \sum_{\mathcal{V}_\alpha^{\alpha_1,\alpha_2}} \sum_{\mathcal{P}_2(\pi)} \mathcal{S}(\pi_1,\pi_2) V_\alpha^{\alpha_1,\alpha_2}(\pi_1,\pi_2) \mathcal{J}_{\alpha_1}(\pi_1) \mathcal{J}_{\alpha_2}(\pi_2)$$

Berends-Giele Recursion

- Reuse parts of calculation
- Most efficient for high multiplicity
- Reduces computation from from O(n!) to O(3^n) for color-dressed or O(n^3) for color-ordered
- **Note:** For color-ordered, the color factor goes like O((n-1)!^{2})

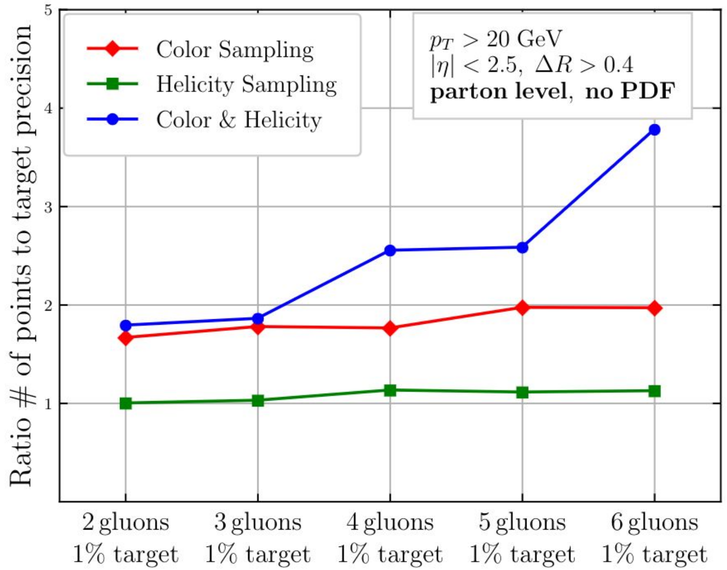


Fermilab

Investigating Matrix Element Algorithms

Bothmann, Giele, Höche, **Isaacson**, Knobbe [2106.06507]

- How to handle sum over unobserved quantum numbers?

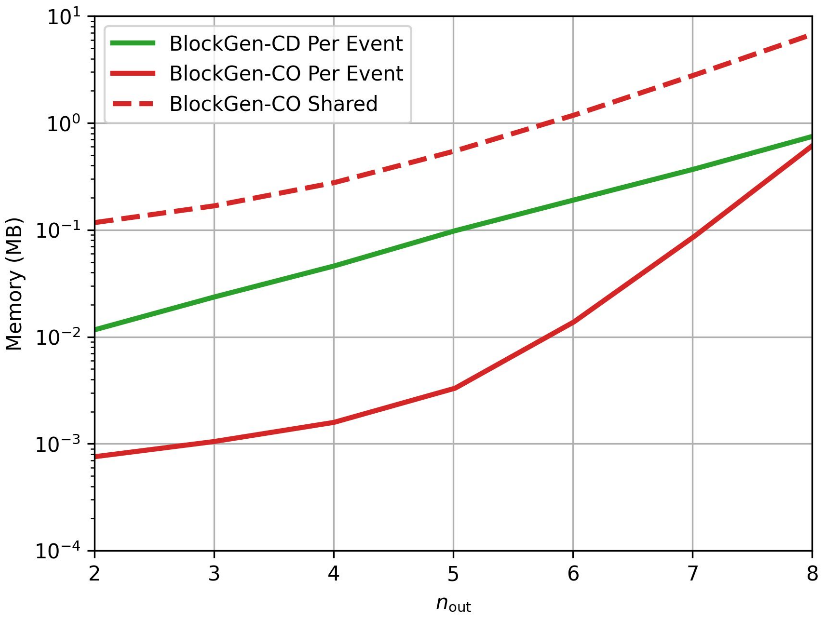


Fermilab

Investigating Matrix Element Algorithms

Bothmann, Giele, Höche, **Isaacson**, Knobbe [2106.06507]

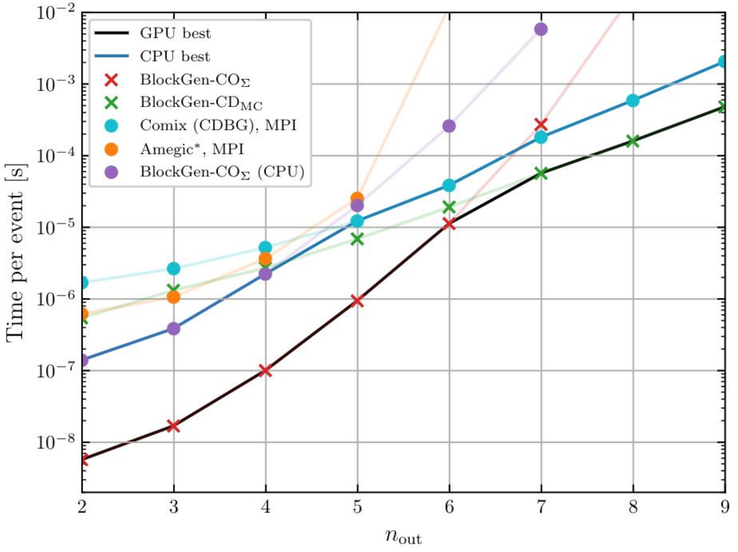
- How to handle sum over unobserved quantum numbers?
- GPUs have taken over scientific computing
- GPUs best suited for this problem
- Memory is challenging for GPUs. What is the best approach?



Investigating Matrix Element Algorithms

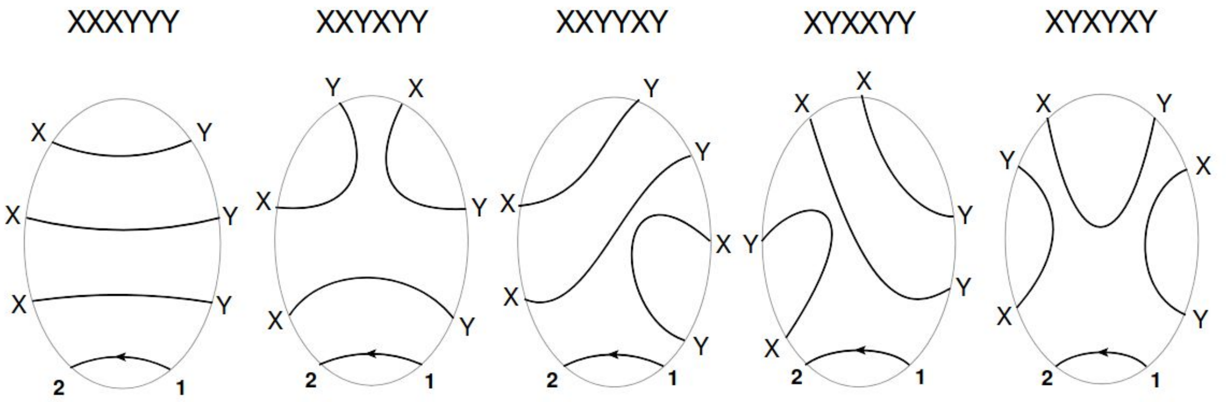
Bothmann, Giele, Höche, **Isaacson**, Knobbe [2106.06507]

- How to handle sum over unobserved quantum numbers?
- GPUs have taken over scientific computing
- GPUs best suited for this problem
- Memory is challenging for GPUs. What is the best approach?
- Chip-to-chip comparison of various algorithms for gluon only processes
- Determine optimal algorithm for a full-fledged generator is color-ordered up to 6 jets



Handling Color

- Use minimal QCD color-basis {A(1,2,σ), σ in Dyck Words}
  [T. Melia: 1304.7809, 1312.0599, 1509.03297] [H. Johansson, A. Ochirov: 1507.00332]
- Allows to fix on fermion line, remaining permutations are given by Dyck Words
- Four particle Dyck Words: ()(), (())
- Significantly fewer amplitudes to compute



[1304.7809]

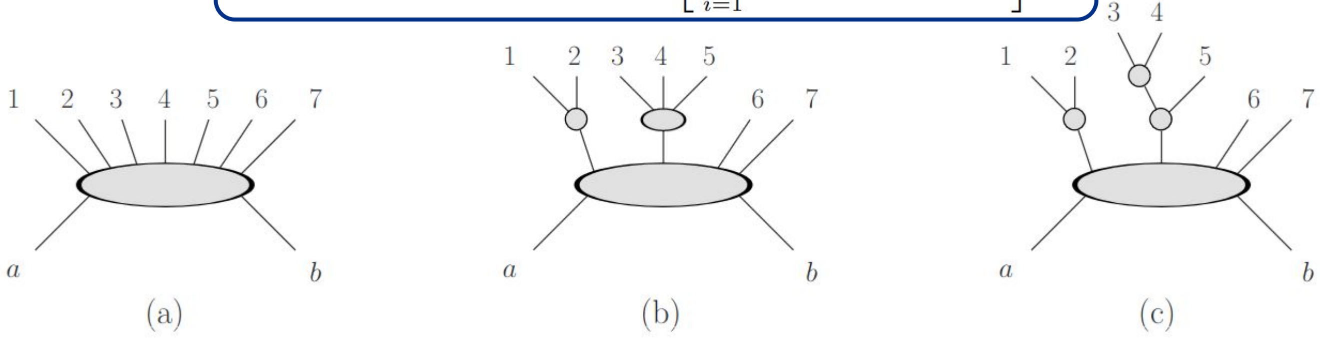
Simplifying Momentum Generation

Bothmann, Childers, Giele, Herren, Höche, **Isaacson**, Knobbe, Wang [2302.10449]

- Uses basic t-channel with single resonance from MCFM as basic building block
- Expand upon MCFM algorithm by adding a controllable number of resonances
- Limiting resonances turns channel scaling from factorial to polynomial

$$\boxed{\mathrm{d}\Phi_n(a,b;1,\ldots,n) = \mathrm{d}\Phi_{n-m+1}(a,b;\pi,m+1,\ldots,n)\frac{\mathrm{d}s_\pi}{2\pi}\mathrm{d}\Phi_m(\pi;1,\ldots,m)}$$

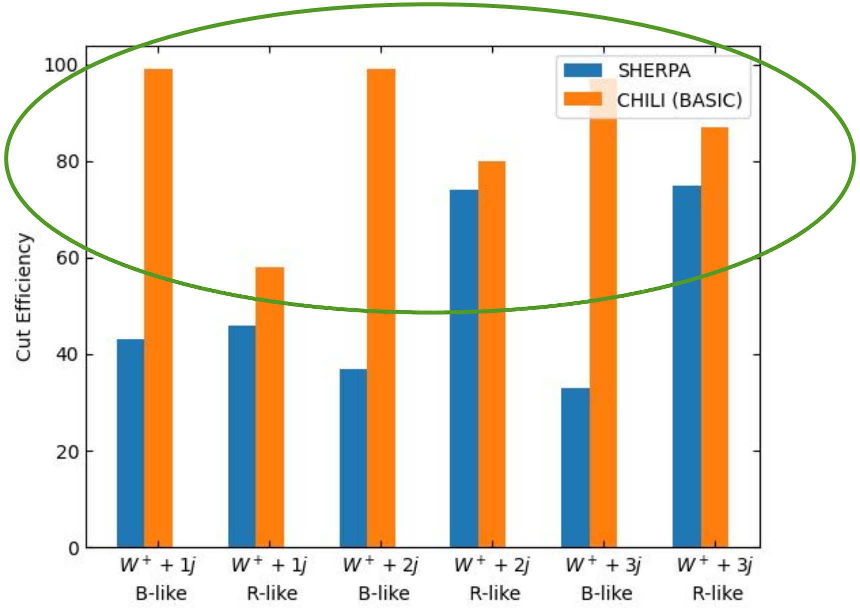
$$\boxed{\mathrm{d}x_a\mathrm{d}x_b\,\mathrm{d}\Phi_n(a,b;1,\ldots,n) = \frac{2\pi}{s}\left[\prod_{i=1}^{n-1}\frac{1}{16\pi^2}\,\mathrm{d}p_{i,\perp}^2\,\mathrm{d}y_i\,\frac{\mathrm{d}\phi_i}{2\pi}\right]\mathrm{d}y_n}$$

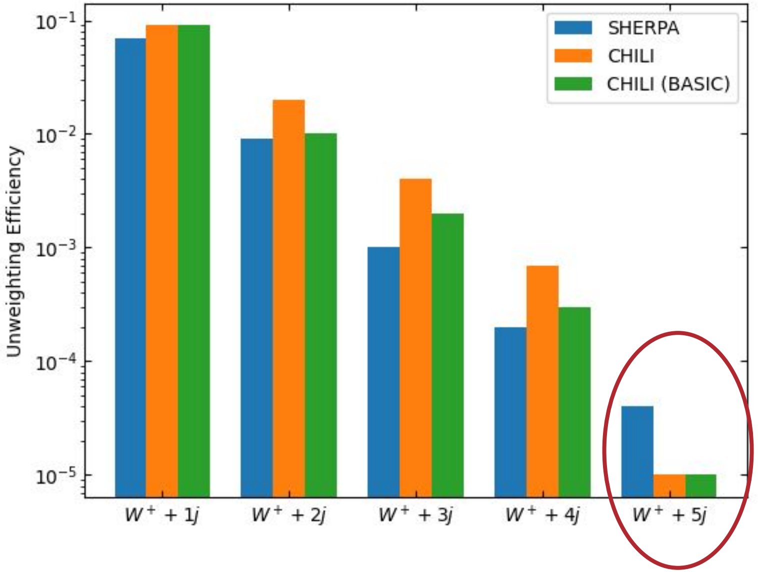


Common High-energy Integration LIbrary (Chili)

Bothmann, Childers, Giele, Herren, Höche, **Isaacson**, Knobbe, Wang [2302.10449]

- Significant cut efficiency improvement
- Simplified phase space evaluation
- Comparable performance up to 5 additional jets

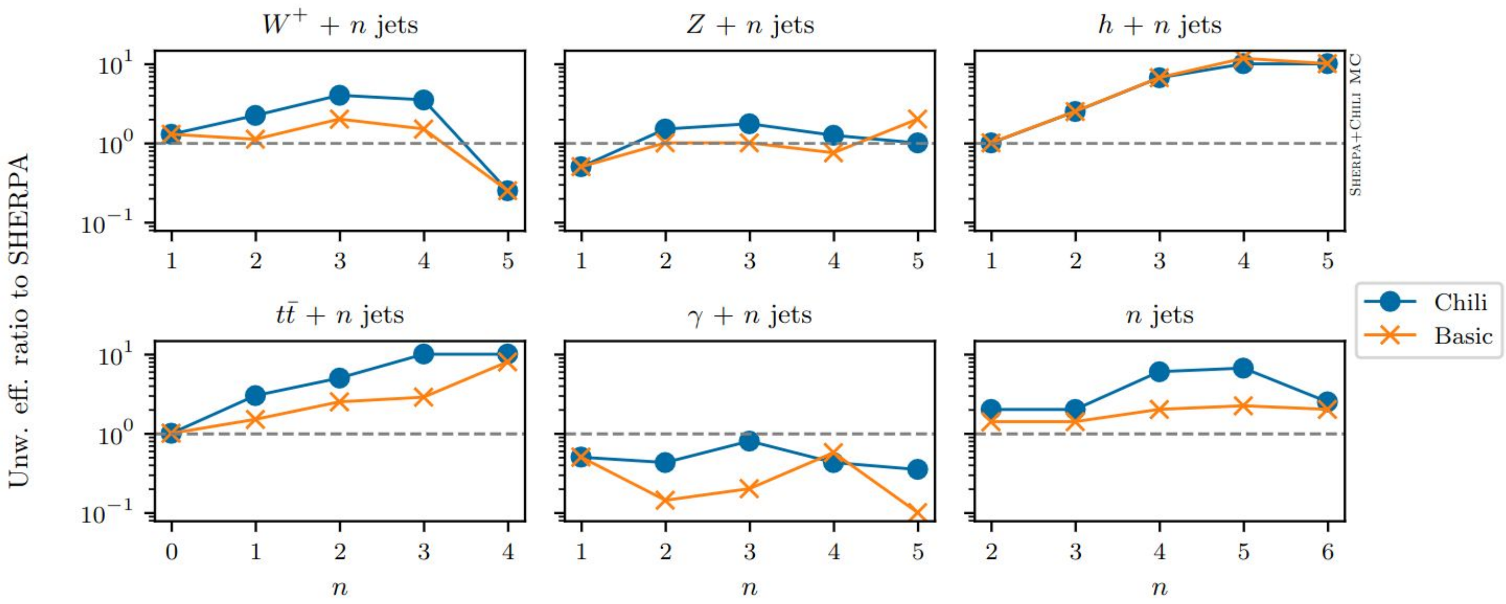




Common High-energy Integration LIbrary (Chili)

Bothmann, Childers, Giele, Herren, Höche, **Isaacson**, Knobbe, Wang [2302.10449]

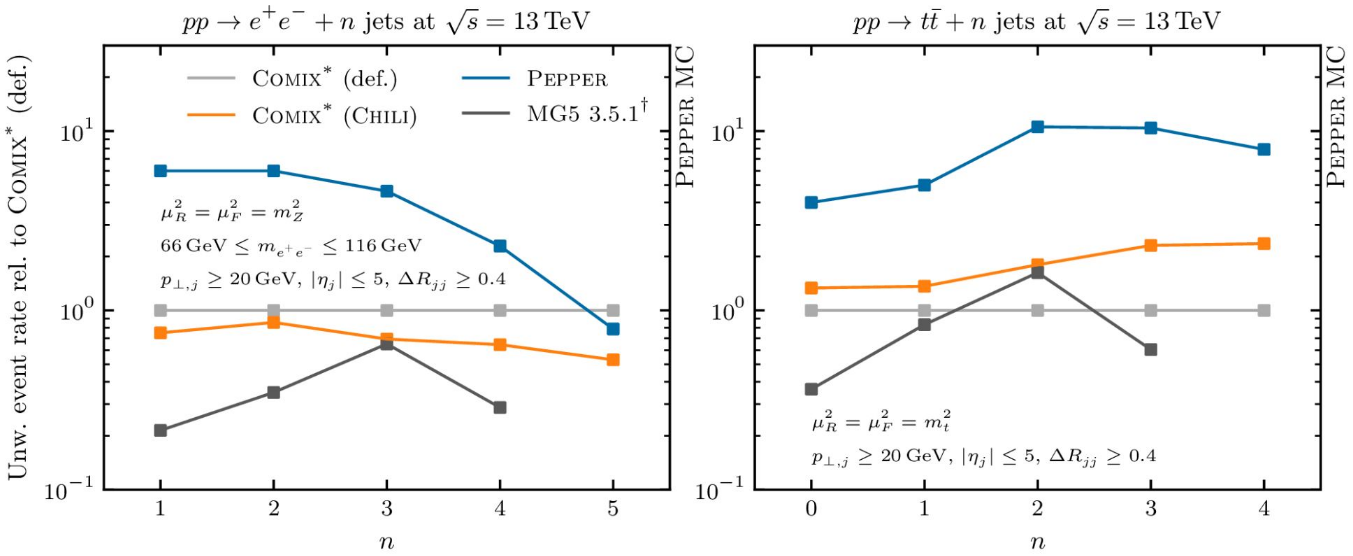
- Significant cut efficiency improvement
- Simplified phase space evaluation
- Comparable performance up to 5 additional jets



Introducing Pepper

Bothmann, Childers, Giele, Höche, **Isaacson**, Knobbe [2311.06198]

- Portable Engine for the Production of Parton-level Event Records
- Interfaces with Sherpa using HDF5 [Bothmann, Childers, Guetschow, Höche, Hovland, **Isaacson**, Knobbe, Latham: 2309.13154]
- Can run on all existing CPU and GPU architectures



PEPPER
PORTABLE PARTON-LEVEL EVENT GENERATOR

Fermilab

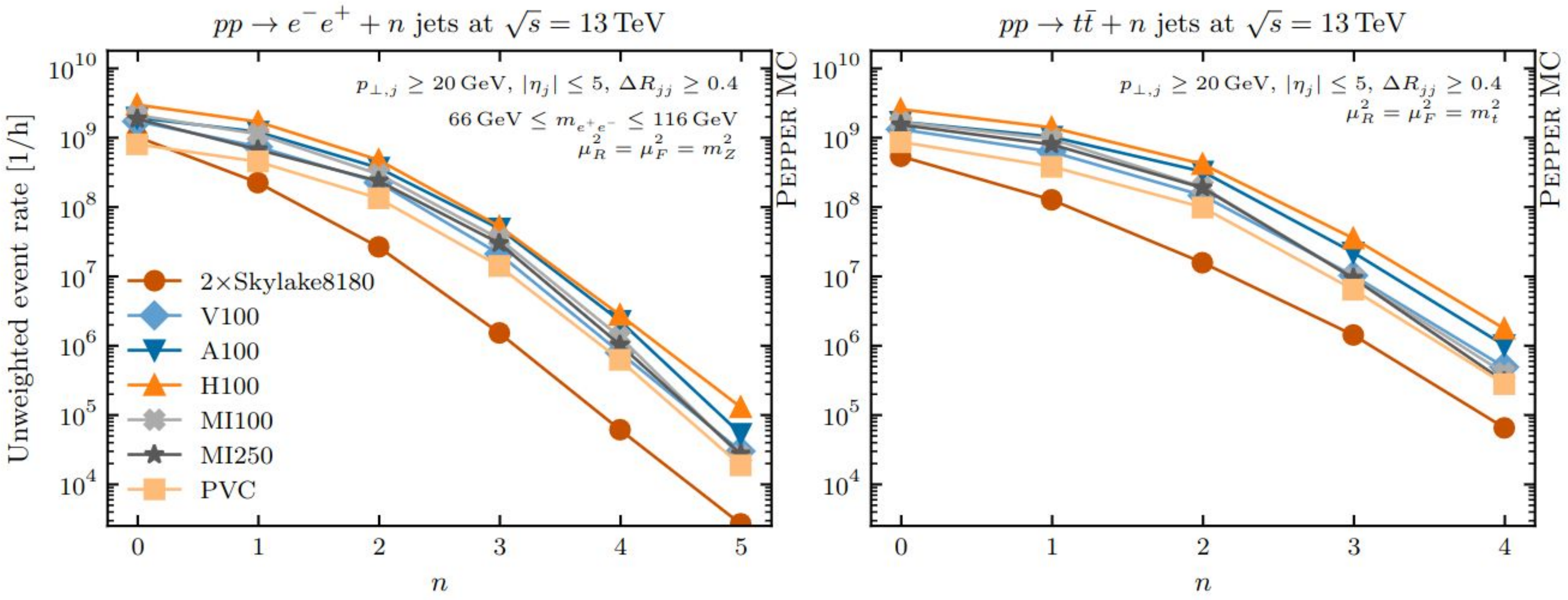
(Pre-)Exascale Computers Pepper can fully use

Bothmann, Childers, Giele, Höche, **Isaacson**, Knobbe [2311.06198]

PEPPER
PORTABLE PARTON-LEVEL EVENT GENERATOR

Fermilab

Comparing Runtimes

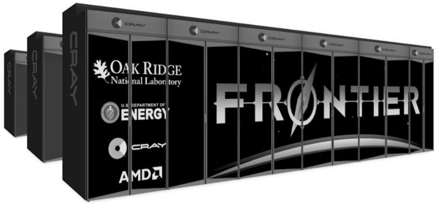


Pepper Current Status

Bothmann, Childers, Giele, Höche, **Isaacson**, Knobbe [2311.06198]

- ATLAS estimated a requirement of **200,000 CPU-years** for V+jet sample in 2021
- **Note:** Pepper results are GPU only!

Sherpa: 4 months



Pepper: 4 hours

Sherpa: 2 months

Pepper: 6 hours

Sherpa: 14 months

Pepper: 8 hours

Sherpa: 6 months

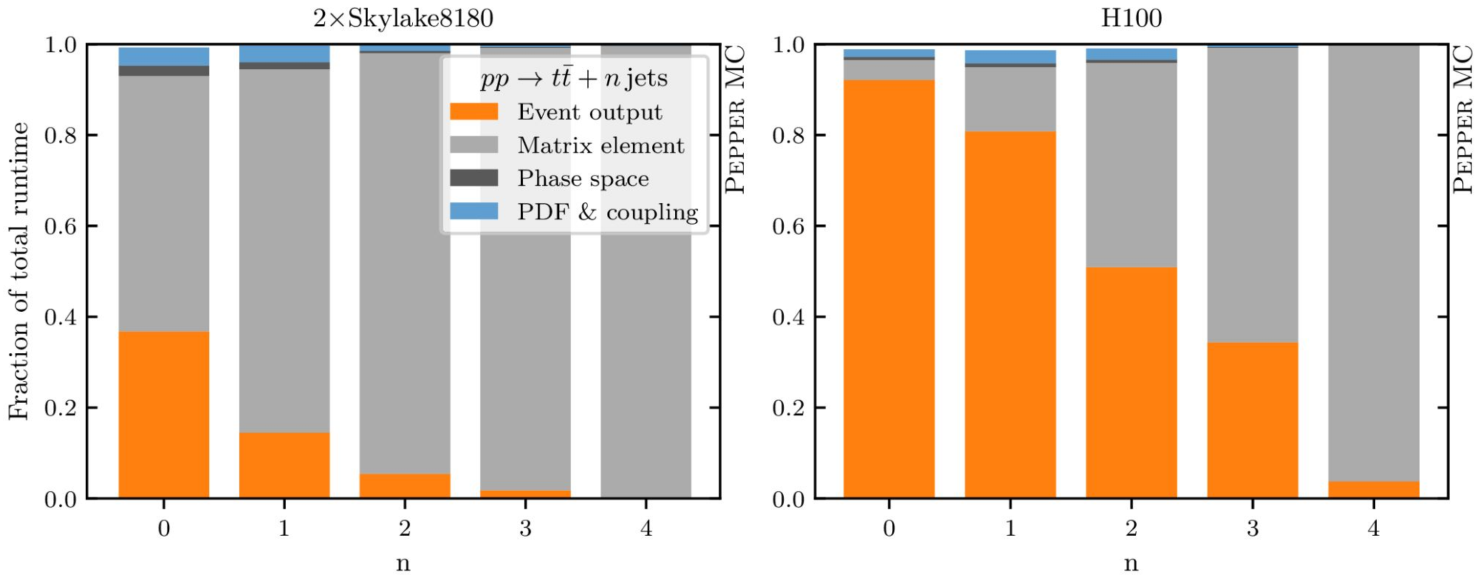
Pepper: 15 hours

	Intel CPU	NVIDIA GPU			AMD GPU		Intel GPU
MEvents / hour	2×Skylake8180	V100	A100	H100	MI100	MI250	PVC
$pp \to t\bar{t} + 4j$	0.06	0.5	1.0	1.7	0.4	0.3	0.3
$pp \to e^- e^+ + 5j$	0.003	0.03	0.05	0.1	0.03	0.03	0.02

PEPPER
PORTABLE PARTON-LEVEL EVENT GENERATOR

Fermilab

Timing Details



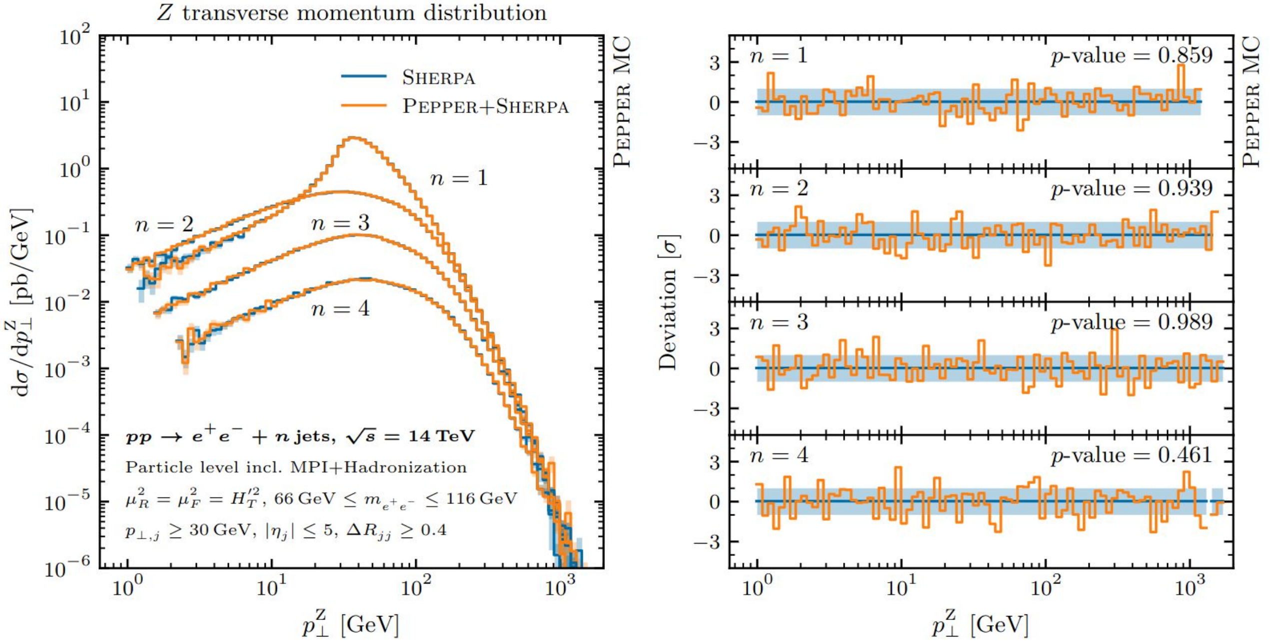
➔ Lower multiplicity are I/O limited

➔ Computing relevant component only at large multiplicities

Physics Validation

Bothmann, Childers, Giele, Höche, **Isaacson**, Knobbe [2311.06198],
Bothmann, Childers, Gütschow, Höche, Hovland, **Isaacson**, Knobbe, Latham [2309.13154]

- ➔ Pepper can interfaced with Sherpa through LHEH5 format



PEPPER
PORTABLE PARTON-LEVEL EVENT GENERATOR

Fermilab

Pepper Future Outlook

Going beyond leading order:

- ➔ Development of novel GPU real subtraction term almost complete [Höche, Knobbe]
- ➔ Converting MCFM virtual corrections to run on GPUs (Preliminary numbers with laptop GPU)
- ➔ Developing library for special functions on GPUs:
  - ◆ Currently have all Polylogarithms (needs optimization)
  - ◆ Working on GPLs

	MCFM (CPU)	MCFM (GPU)
W+1jet	~4000 ns / event	~300 ns / event
Z+1jet	~8000 ns / event	~600 ns / event

Pepper Future Outlook

Going beyond leading order:

- ➔ Developing library for special functions on GPUs:
  - ◆ Currently have all Polylogarithms (needs optimization)
  - ◆ Working on GPLs
- ➔ Li_2 and Li_3 implementations based on [Voigt: 2201.01678, 2308.11619]

$$\mathrm{li}_3(x) = x\sum_{k=0}^{\infty}\frac{x^k}{(k+1)^3} \approx x\frac{\sum_{k=0}^{5} p_k x^k}{\sum_{k=0}^{6} q_k x^k}$$

	FastGPL	Pepper	
		CPU	GPU
Li_2	240 ns / point	27 ns / point	6.5 ns / point
Li_3	220 ns / point	17 ns / point	6.5 ns / point

Fermilab

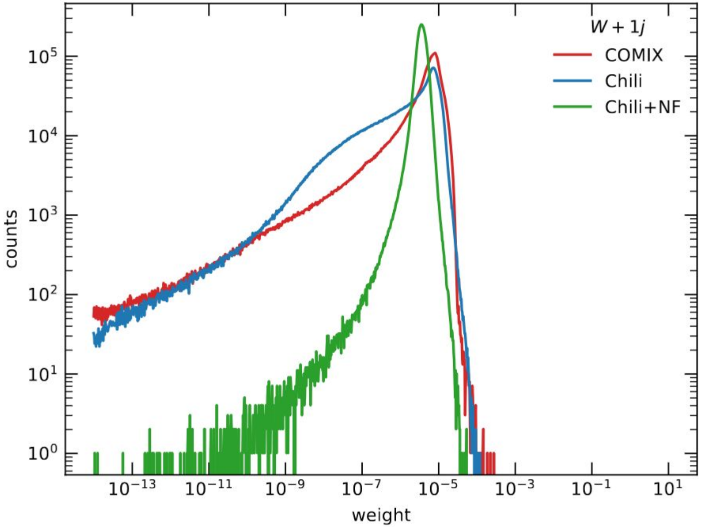
Pepper Future Outlook

Provide a python interface to Pepper for machine learning applications

➔ Normalizing Flows for Phase Space:

[Bothmann et al: 2001.05478, **JI** et al. 2001.10028, **JI** et al. 2001.05486, Heimel, **JI**, et al. 2212.06172, Heimel et al. 2311.01548]

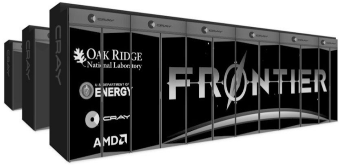
- Pepper matrix elements are sufficiently fast on GPUs
- Previously limited to CPU training of networks, now all can be run on GPUs
- Will enable investigation of higher multiplicity processes than previously possible



Process (color sum)	COMIX		CHILI		CHILI +NF	
	Opt 0.8M pts	Gen 6M pts	Opt 0.8M pts	Gen 6M pts	Opt 1.2M pts	Gen 6M pts
W^{+}+1j	2m	10m	1m	8m	5m	8m
W^{+}+2j	14m	1.9h	13m	1.7h	29m	1.3h

Conclusions

Sherpa: 4 months



Pepper: 4 hours

Sherpa: 2 months

Pepper: 6 hours

Sherpa: 14 months

Pepper: 8 hours

Sherpa: 6 months

Pepper: 15 hours

- ➔ Pepper resolves tree-level computational issue for experiments
  - ◆ Achieves scalability laptop from a Leadership Computing Facility
  - ◆ Interface to Sherpa for direct usage by experiments already
- ➔ Ongoing work to extend the framework beyond leading order
- ➔ Adding additional processes beyond V+jet and ttbar+jet
- ➔ Developing as a framework to be used for fixed order calculations by providing a library interface (under way)
- ➔ Use synergies between Pepper / Chili, on-device training, and ML

PEPPER
PORTABLE PARTON-LEVEL EVENT GENERATOR

Backup

Normalizing Flows

Gao, **Isaacson**, Krause [2001.05486]

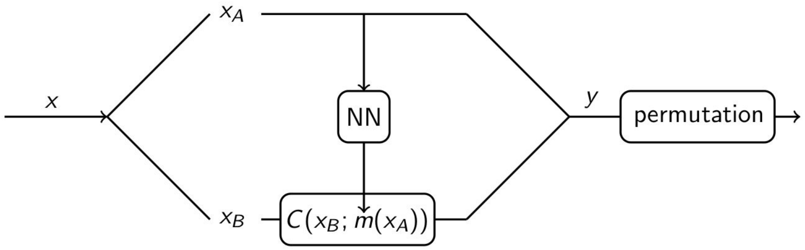
GOAL: Create a network that can learn a change of variables, that transforms an easy to sample base distribution into the desired target distribution

$$z \sim \pi(z), x = f(z), z = f^{-1}(x)$$

$$p(x) = \pi(z)|\det \frac{\mathrm{d}z}{\mathrm{d}x}|$$

[1410.08516] [1808.03856] [1906.04032]

Basic building block



Fermilab

Normalizing Flows

Gao, **Isaacson**, Krause [2001.05486]

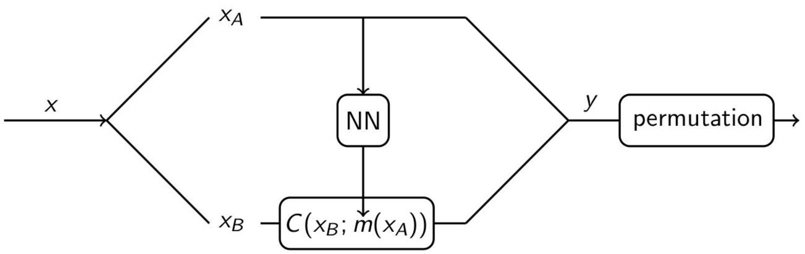
GOAL: Create a network that can learn a change of variables, that transforms an easy to sample base distribution into the desired target distribution

$$z \sim \pi(z), x = f(z), z = f^{-1}(x)$$

$$p(x) = \pi(z) |\det \frac{\mathrm{d}z}{\mathrm{d}x}|$$

[1410.08516] [1808.03856] [1906.04032]

Basic building block



Forward Transform:

$$y_A = x_A$$

$$y_{B,i} = C(x_{B,i}; m(x_A))$$

Inverse Transform:

$$x_A = y_A$$

$$x_{B,i} = C^{-1}(y_{B,i}; m(y_A))$$

Jacobian

$$J = \begin{vmatrix} 1 & \frac{\partial C(x)}{\partial x_A} \\ 0 & \frac{\partial C(x)}{\partial x_B} \end{vmatrix} = \frac{\partial C(x_B; m(x_A))}{\partial x_B}$$

i-flow for LHC Event Generation

Gao, Höche, **Isaacson**, Krause, Schulz [2001.10028]

Unweighting Efficiency:

1. Draw nN_{opt} events, where N_{opt} events used to optimize integrator
2. Select m replicas of N_{opt} events and compute maximum weight
3. Define w_{max} as median of individual maxima

Efficiency is then given as:

$$\eta_{\rm bs} = \frac{\langle f/g \rangle_G}{w_{\rm max}}$$

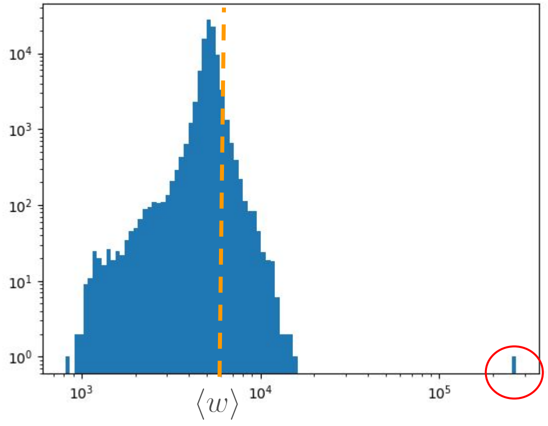
Fermilab

i-flow for LHC Event Generation

Gao, Höche, **Isaacson**, Krause, Schulz [2001.10028]

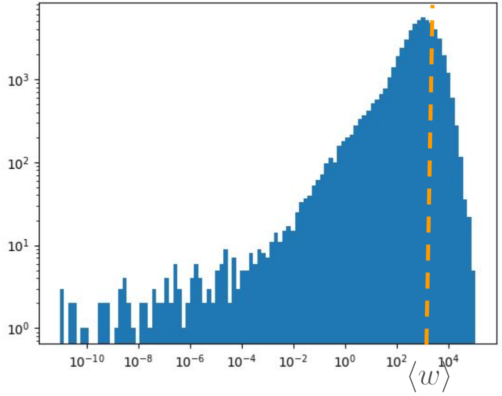
Unweighting Efficiency:

1. Draw nN_{opt} events, where N_{opt} events used to optimize integrator
2. Select m replicas of N_{opt} events and compute maximum weight
3. Define w_{max} as median of individual maxima



Efficiency is then given as:

$$\eta_{\mathrm{bs}} = \frac{\langle f/g \rangle_G}{w_{\mathrm{max}}}$$



Multichanneling (Mixture Distributions)

- Introduce several variable transformations (channels):

$$g_i(x) = \left|\frac{\partial G_i(x)}{\partial x}\right| \qquad \text{with} \qquad \int \mathrm{d}x\, g_i(x) = 1 \quad \text{for} \quad i = 1, \ldots, m\,,$$

- Introduce parameter to combine channels into single distribution

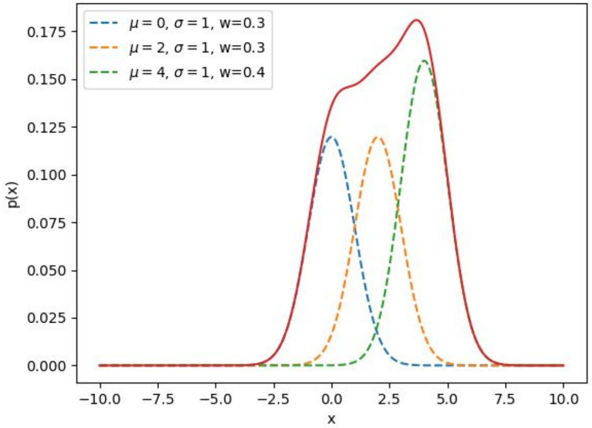
$$g(x) = \sum_i^m \alpha_i g_i(x) \qquad \text{with} \qquad \sum_i^m \alpha_i = 1 \quad \text{and} \quad \alpha_i \geq 0\,,$$

- The integral can now be calculated as:

$$I[f] = \sum_i^m \int_\Phi \mathrm{d}^d x\, \alpha_i\, g_i(x) \frac{f(x)}{g(x)} = \sum_i^m \int_{U_i} \mathrm{d}^d y\, \alpha_i\, \frac{f(x)}{g(x)}\bigg|_{x=\overline{G}_i(y)}\,,$$

- Can promote parameter to be local:

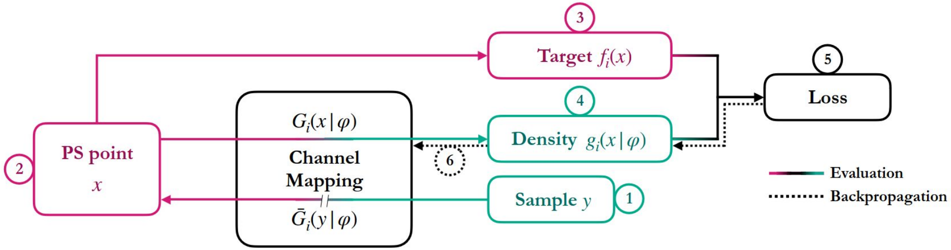
$$\alpha_i(x) = \alpha_i \frac{g_i(x)}{g(x)}\,,$$

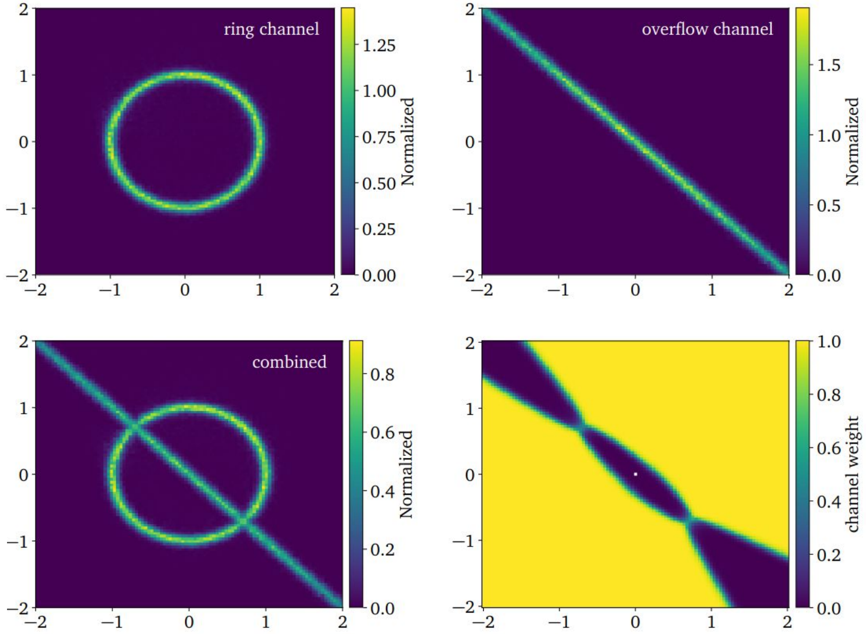


This is MadNIS!

Heimel, Winterhalder, Butter, **Isaacson,** Krause, Maltoni, Mattelaer, Plehn [2212.06172]

- Leverage ideas from i-flow
- Promote channel weights to a learnable distribution given by a NN
- Implement buffered training to improve on computational time
- Would benefit from differential matrix element

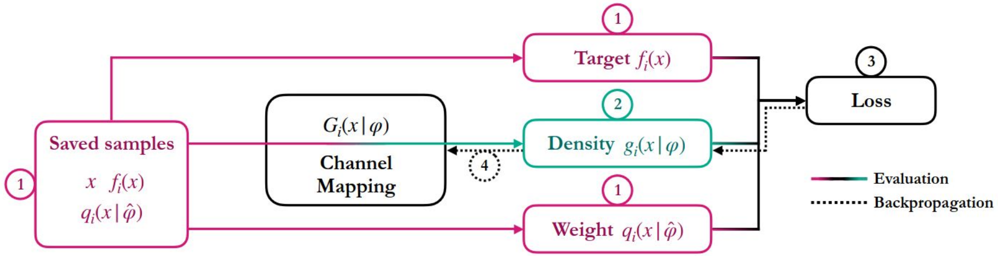


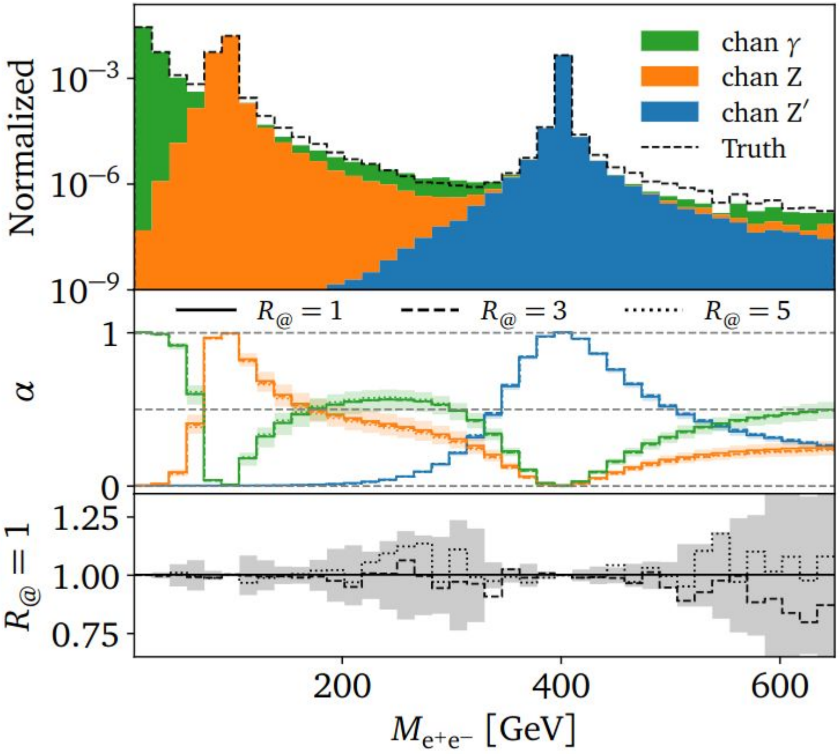


This is MadNIS!

[Heimel, Winterhalder, Butter, **Isaacson**, Krause, Maltoni, Mattelaer, Plehn [2212.06172]](https://arxiv.org/abs/2212.06172)

- Leverage ideas from i-flow
- Promote channel weights to a learnable distribution given by a NN
- Implement buffered training to improve on computational time
- Would benefit from differential matrix element





Fermilab