WP B1 – phase space sampling

brief recap & status update

Steffen Schumann & Timo Janßen (a), Cornelius Grunwald (b)

(a) Institut für Theoretische Physik
Georg-August-Universität Göttingen

&

(b) Fakultät für Physik
Technische Universität Dortmund

KISS Annual Meeting 2024

KISS

WP B1 – TUDO/UHD/UGOE

Phase Sampling for HEP Monte Carlos

- stochastic exploration of high-dimensional spaces
  $\hookrightarrow$ phase space of collider scattering events
  $\hookrightarrow$ Baysian Inference techniques
- ML augmented samplers, novel MCMC-type algorithms
- incorporate into SHERPA (MADGRAPH) generator

M1 interface SHERPA & BAT.jl ✓

M2 performance studies ... on the way

M3 novel MCMC-type algorithms within BAT.jl ... still to come

M4 improved multi-channel samplers ... on the way

The HEP trinity

Theory

relativistic QFT

quantum $\rightsquigarrow$ non-deterministic

reference Standard Model $\mathcal{L}_{\mathrm{SM}}$

hypothetical New Physics $\mathcal{L}_{\mathrm{BSM}}$

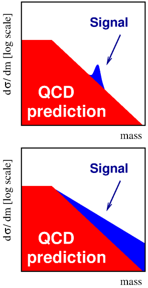
Experiment

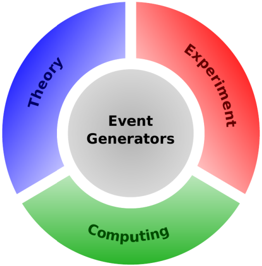
multi-component detector

ATLAS, CMS, LHCb, ALICE, ...

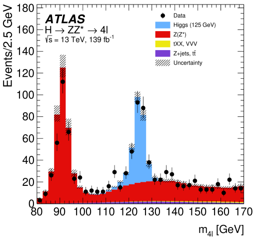
reconstruction of events

operation, degrading, upgrades





[ATLAS EPJC 80 (2020) 942]



Simulation – emulating nature

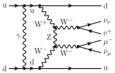
particle-level event generation (theory)

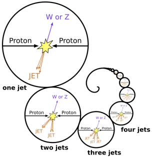
detector-response simulation (experiment)

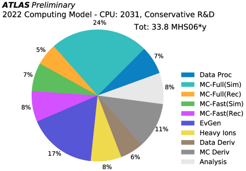
Novel algorithms for event generation and beyond

Computational bottleneck: the hard event component

$$\boxed{\sigma_{pp\to X_n} = \sum_{ab}\int \mathrm{d}x_a\mathrm{d}x_b\,\mathrm{d}\Phi_n\; f_a(x_a,\mu_F^2)f_b(x_b,\mu_F^2)\,|\mathcal{M}_{ab\to X_n}|^2\;\Theta_n(p_1,\ldots,p_n)}$$







[CERN-LHCC-2022-005]

$\hookrightarrow$ $|\mathcal{M}|^2$ multi-modal, wildly fluctuating, expensive

$\hookrightarrow$ real- & virtual quantum corrections, IR subtractions

$\hookrightarrow$ Monte-Carlo phase space sampling [$\mathrm{dim}[\Phi_n] = 3n-4$]

main research thrusts (towards HL-LHC)

$\hookrightarrow$ sustainable simulations on modern hardware (GPU)

[Bothmann *et al.*] [Carrazza *et al.*] [Mattelaer *et al.*]

$\hookrightarrow$ application of machine learning (ML), e.g.

NN surrogate unweighting, ML sampling algorithms

A1 **B1**

KISS

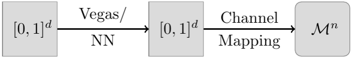
Novel algorithms: Neural Importance Sampling

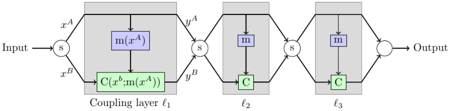
ML-assisted phase space sampling

- MCEG use physics informed importance sampling
  - $\rightsquigarrow$ aim to reduce event weight variations (automation)
  - $\rightsquigarrow$ adaptive multi-channel sampler: SHERPA, MADGRAPH
- **improve sampling efficiency through Normalizing Flows**
  - $\rightsquigarrow$ bijective remapping of random numbers for channel maps

    [Müller *et al.*, arXiv:1808.03856] [Bothmann *et al.*, SciPost Phys. 8 (2020) no.4, 069]

    [Gao *et al.*, PRD 101 (2020) no.7, 076002] [Heimel *et al.* SciPost Phys. 15 (2023) 141]





  - $\rightsquigarrow$ invertible *coupling layers* with tractable Jacobian
  - $\rightsquigarrow$ more expressive than standard VEGAS remapping

Novel algorithms: Neural Importance Sampling

ML-assisted phase space sampling – closing in on production

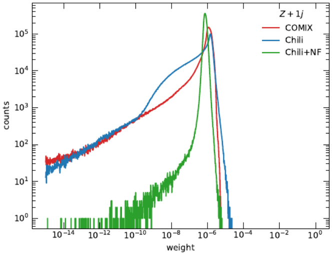
- implementation in SHERPA framework

  [Gao *et al.*, PRD 101 (2020) no.7, 076002] [Bothmann *et al.*, SciPost Phys. 15 (2023) 4]

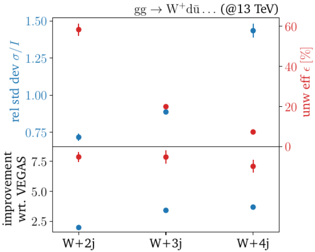
- MADNIS multi-channel sampler for MADGRAPH

  [Heimel *et al.*, SciPost Phys. 15 (2023) 141 & 2311.01548]

SHERPA $Z + 1j$



MADNIS



⇝ powerful integration/sampling method

⇝ enormous potential for other applications, e.g. loop calcs

[Winterhalder *et al.*, SciPost Phys. 12 (2022) no.4, 129] [Jinno *et al.*, JHEP 7 (2023) 181]

Bayesian Inference – in a nutshell

- consider model M with parameters Θ_M, some data D
- estimate *conditional* probability $P(\Theta_M|D,M)$ (**posterior**)

$\rightsquigarrow$ **Bayes' Theorem** yields

$$P(\Theta_M|D,M) = \frac{P(D|\Theta_M,M)P(\Theta_M|M)}{P(D|M)} = \frac{\mathcal{L}(\Theta)\pi(\Theta)}{\mathcal{Z}}$$

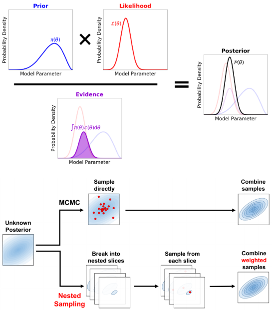
$\hookrightarrow$ with **likelihood** $\mathcal{L}(\Theta)$, **prior** $\pi(\Theta)$, **evidence** $\mathcal{Z}$

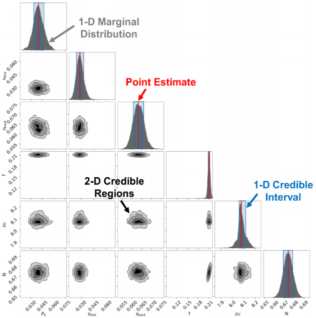
$$P(D|M) = \int_{\Omega_\Theta} P(D|\Theta_M,M)P(\Theta_M|M)d\Theta_M$$

challenge – efficiently sample posterior distribution

- typically high dimensional, possibly multi-modal
- likelihood function $\mathcal{L}(\Theta)$ expensive to evaluate

Novel algorithms: MCMC-type samplers





- Bayesian analysis standard tool in HEP, cosmo, astro, ...
- realm of MCMC algorithms to sample posterior
- evidence integral for example via Nested Sampling [Skilling '04 & '06]

  $\hookrightarrow$ dedicated tools: Gambit, CosmoMC, PolyChord, BAT.jl
- explore application to HEP phase-space sampling/integration

  [Kröninger et al. '14] [Yallup et al. '22]

Generating phase space for particle collisions

We need to generate four-momenta of final-state particles in the Lorentz-invariant phase space

$$\mathrm{d}\Phi_n(P, p_1, \dots, p_n) = \prod_{i=1}^{n} \frac{\mathrm{d}^3 p_i}{(2\pi)^3 2E_i} (2\pi)^4 \delta^4\Big(P - \sum_{i=1}^{n} p_i\Big).$$

Four-momentum conservation and on-shell conditions reduce the dimensionality to

$$d = 3n - 4\,.$$

Generating phase space for particle collisions

We need to generate four-momenta of final-state particles in the Lorentz-invariant phase space

$$\mathrm{d}\Phi_n(P, p_1, \dots, p_n) = \prod_{i=1}^{n} \frac{\mathrm{d}^3 p_i}{(2\pi)^3 2E_i} (2\pi)^4 \delta^4\Big(P - \sum_{i=1}^{n} p_i\Big).$$

Four-momentum conservation and on-shell conditions reduce the dimensionality to

$$d = 3n - 4\,.$$

For massless particles, we can populate the phase space *uniformly* using the **Rambo algorithm** [Kleiss *et al.*. Comp.Phys.Comm. 40, (1986), 359]:

1. generate n four-momenta with isotropic angles and energies distributed like xe^{-x}
2. use Lorentz boost and scaling transformation to transform to the physical momenta

$\rightarrow$ maps the hypercube $[0,1]^{4n}$ to n four-momenta with $E_{\mathrm{CMS}} = \sqrt{P^2}$

Generating phase space for particle collisions

- **RAMBO on diet** [Plätzer 1308.2922] reduces the number of random variables to $3n - 4$, the minimum number of d.o.f.
- requires to find root of polynomial
- *unique, invertible mapping* between point in hypercube and physical phase space point

Allows for the following approach:

1. use a generic sampler to generate points in the hypercube
2. map points to physical momenta
3. evaluate differential cross-section on those momenta

Interface to SHERPA

- define all parameters (process definition, beam energy, kinematic cuts, PDFs, ...) in the SHERPA run card
- hadronic collisions require to sample the momentum fractions of the initial-state partons
  $\rightarrow$ increase phase space dimensionality to $d_{\text{hadronic}} = 3n - 4 + \mathbf{2}$
- for now only partonic subprocesses but full jet processes could be automated as well

Program flow:

- take point in unit hypercube
- perform RAMBO mapping internally
- return differential cross-section as weight
- if outside of cuts, return zero
- events can be written to HEPMC format for further processing

B1.M1: Developing an interface between Sherpa & BAT.jl for phase space sampling

The Bayesian Analysis Toolkit - BAT.jl

- collection of state-of-the art algorithms for Bayesian data analysis
- widely extended re-write of C++ tool in Julia language
- provides modern sampling approaches & new algorithms

BAT.jl
Bayesian Analysis Toolkit
https://github.com/bat/BAT.jl

user-specified:

- likelihood & data
- parameters & prior

provided by BAT.jl:

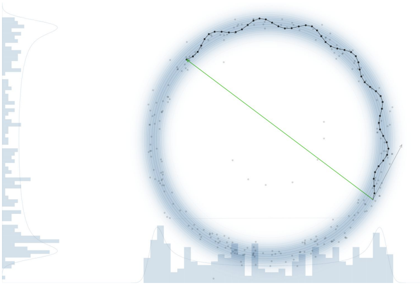
- sampling algorithms
  - MCMC sampling
  - Nested Sampling
- integration algorithms
- optimization algorithms

automated posterior exploration

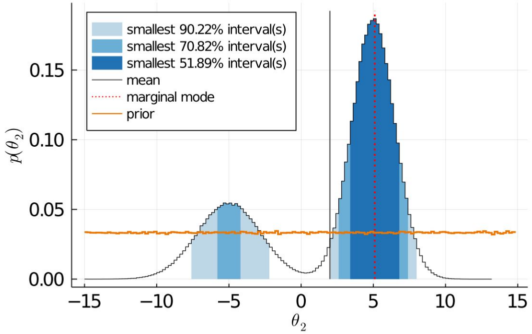
(tuning, parameter space transformations, parallelization, ...)

outputs

- samples
- plots
- modes, mean values, intervals



https://github.com/chi-feng/mcmc-demo



Features of BAT.jl

- use custom target distributions
- collection of sampling algorithms:
  - MCMC: Metropolis-Hastings, Hamiltonian-MC (e.g. NUTS)
  - Importance Samplers
  - Nested Sampling
  - in development: Normalizing flow enhanced MCMC sampling
- automated initialization, tuning & convergence tests
- parameter space transformations
- integration algorithms: Nested Sampling, Adaptive Harmonic Mean Integration (AHMI), CUBA
- design idea: offer reasonable default settings for ease of use, but allow fine-grained control for experienced users

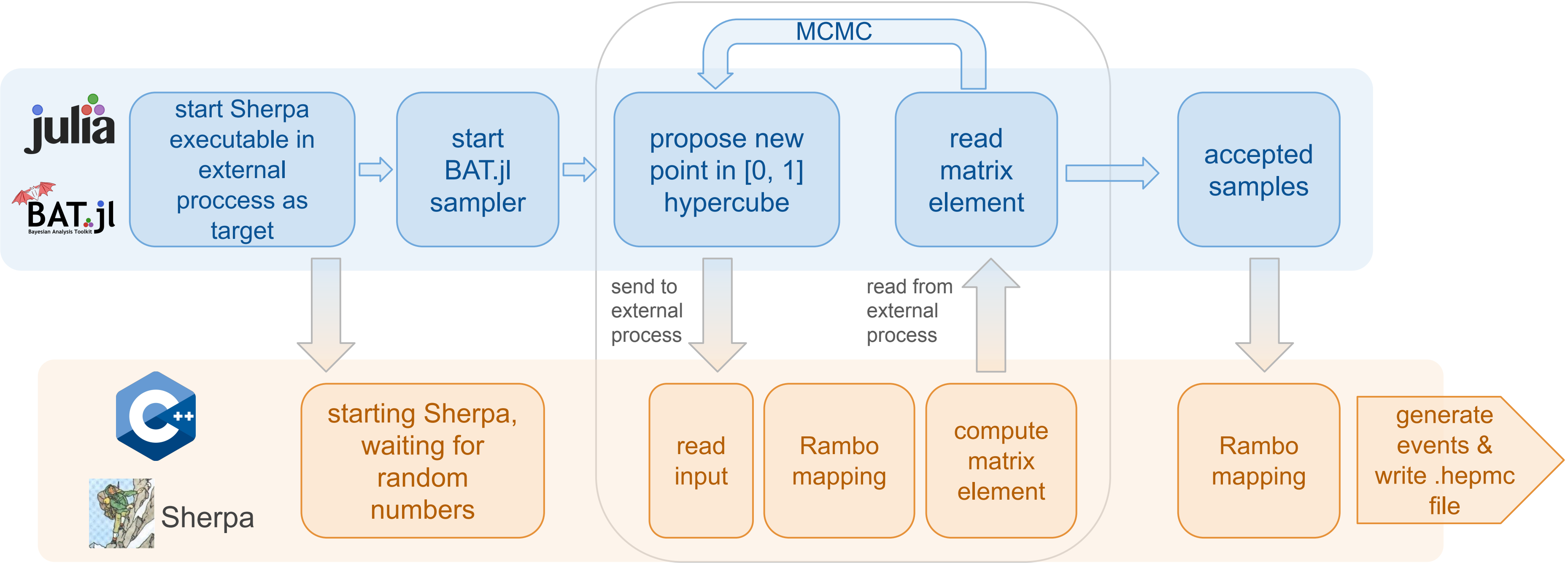
BAT.jl
Bayesian Analysis Toolkit

Installing Julia & BAT.jl:

```
# download & unzip Julia
$ wget https://julialang-s3.julialang.org/bin/linux/x64/1.5/julia-
1.5.3-linux-x86_64.tar.gz
$ tar -xvzf julia-1.5.3-linux-x86_64.tar.gz
$ cd julia-1.5.3/bin

# run Julia
$ julia
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.3.1 (2019-12-30)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

# install BAT.jl (only on first use)
julia> using Pkg
julia> Pkg.add("BAT")

# use BAT.jl
julia> using BAT
```

Interfacing BAT.jl & Sherpa

Current interface: Run BAT.jl and call Sherpa as the target distribution



Later ("in production"): call BAT.jl samplers from within Sherpa

First simple example

Process: $g\, g \longrightarrow g\, g\, g$

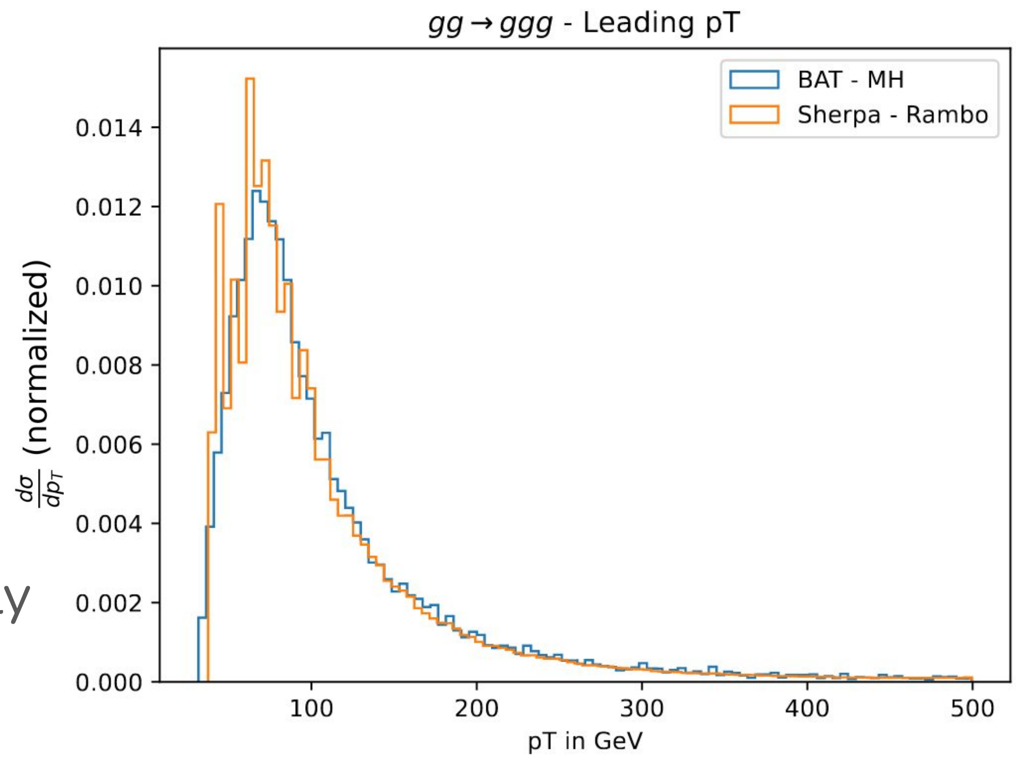
Phase space: 5 dimensions

Sherpa (Rambo)

with BAT.jl (MH) interfaced

➡ interface is working properly

p_T distribution of the leading gluon:

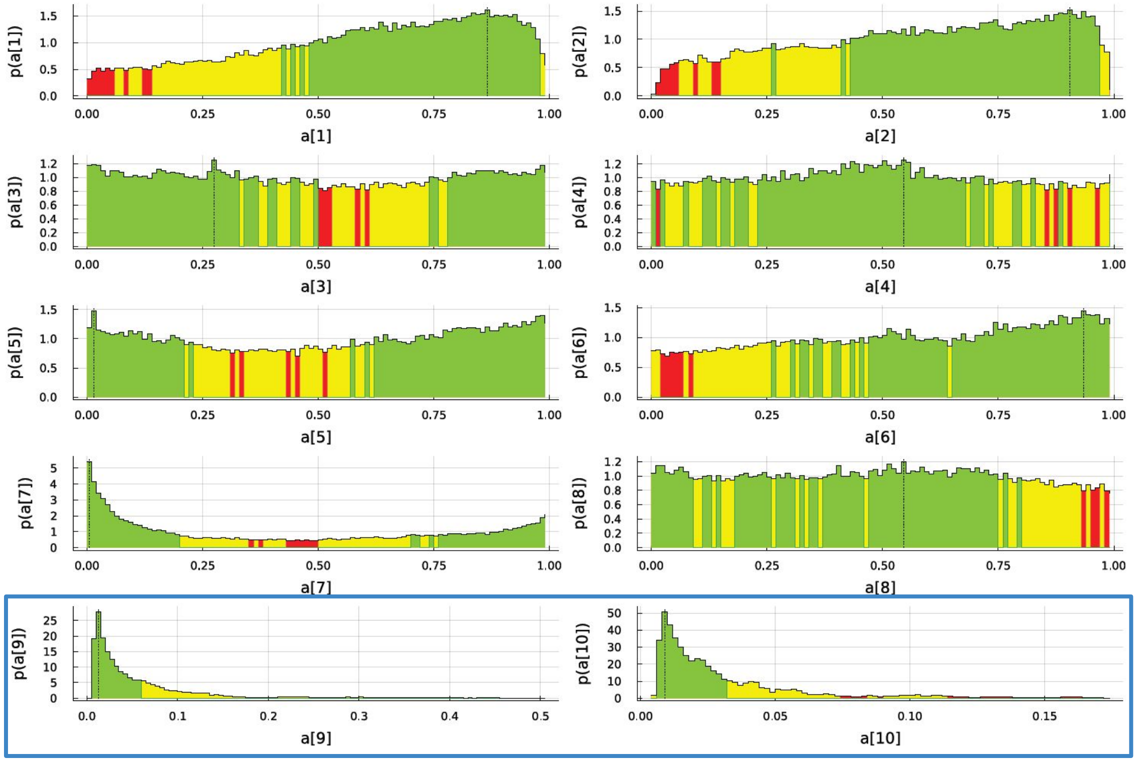


More complex example

Process: $g\, g \longrightarrow d\, \bar{d}\, e^{+}\, e^{-}$ @13GeV

Phase space: 10 dimensions

MH samples from BAT.jl:



More complex example

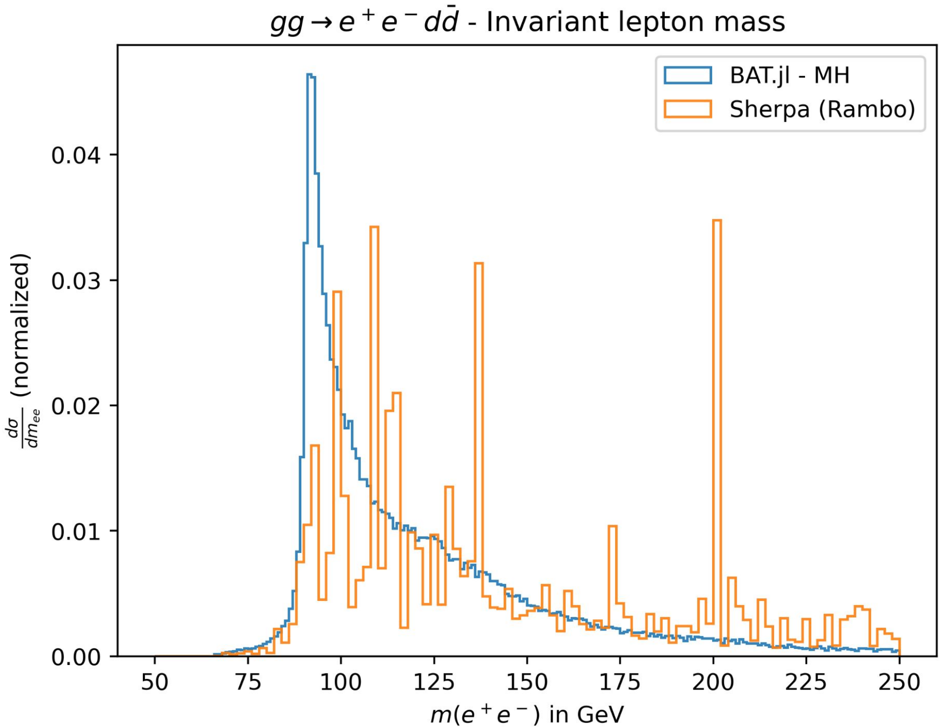
Process: $g\, g \longrightarrow d\, \bar{d}\, e^{+}\, e^{-}$ @13GeV

Phase space: 10 dimensions

Sherpa (Rambo)

with BAT.jl (MH) interfaced

Invariant lepton mass:



More complex example

Process: $g\, g \longrightarrow d\, \bar{d}\, e^{+}\, e^{-}$ @13GeV

Phase space: 10 dimensions

Sherpa (Rambo)

with BAT.jl (MH) interfaced

Next Steps

- use different BAT.jl samplers
- run performance tests (see C1)
- try more complex examples

Invariant lepton mass: