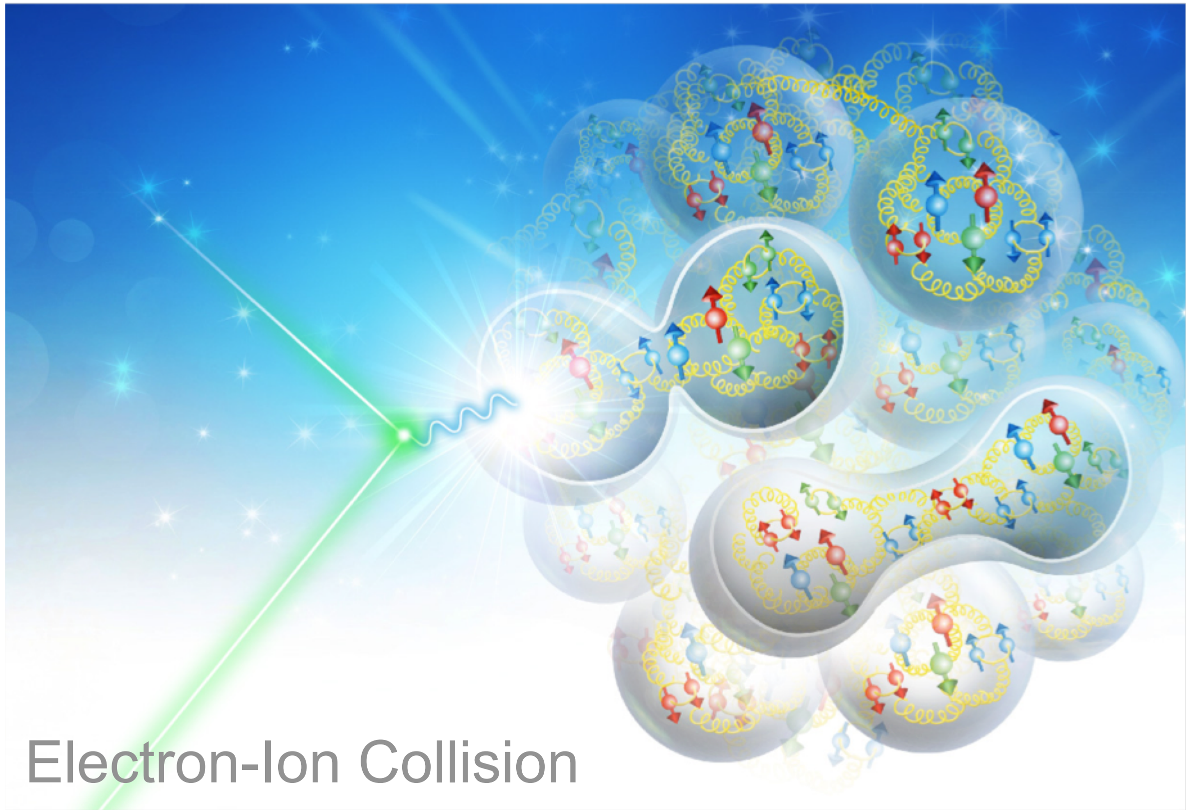
EICUG Software Working Group

Andrea Bressan (INFN, University of Trieste)
Markus Diefenthaler (EIC^2, Jefferson Lab)
Torre Wenaus (Brookhaven Lab)


Electron-Ion Collision

BROOKHAVEN NATIONAL LABORATORY

Jefferson Lab

UNIVERSITÀ DEGLI STUDI DI TRIESTE

Role of Software Working Group

Develop

Support

Workflow environment for EICUG

- **to use** (tools, documentation, support) **and**
- **to grow with user input** (direction, documentation, tools)

Involvement from EICUG

Introduction

Getting started

Point of entry

HOME | JOIN EICUG | SCIENCE | ORGANIZATION | PHONEBOOK | CALENDAR | SOFTWARE | DOCUMENTS | MEDIA | LOGIN

Home » EIC Software

EIC Software

Software Working Group

The EICUG has formed a Software Working Group that collaborates with EIC Software initiatives and other experts in NP and HEP on detector and physics simulations for the EIC. The short-term goal of the working group is to meet in FY20 the requirements for common tools and documentation in the EICUG. The current work focusses on a common Geant4 infrastructure for the EIC that allows geometry exchange between the eRHIC and JLEIC concepts.

JupyterLab

The Software Working Group has adapted JupyterLab as a collaborative workspace to further develop EIC Science, to examine detector requirements, and to work on detector designs and concepts. JupyterLab is a web-based interactive analysis environment to create and share documents that contain the analysis code, the narrative of the analysis including graphics and equations, and visualizations of the analysis results. This will allow the EICUG not only to pursue simulations in a manner that is accessible, consistent, and reproducible to the EICUG as a whole, but also to build a collection of analyses and analysis tools in the fully extensible and modular JupyterLab environment. A quick start tutorial for fast simulations is available on the website for EIC Software.

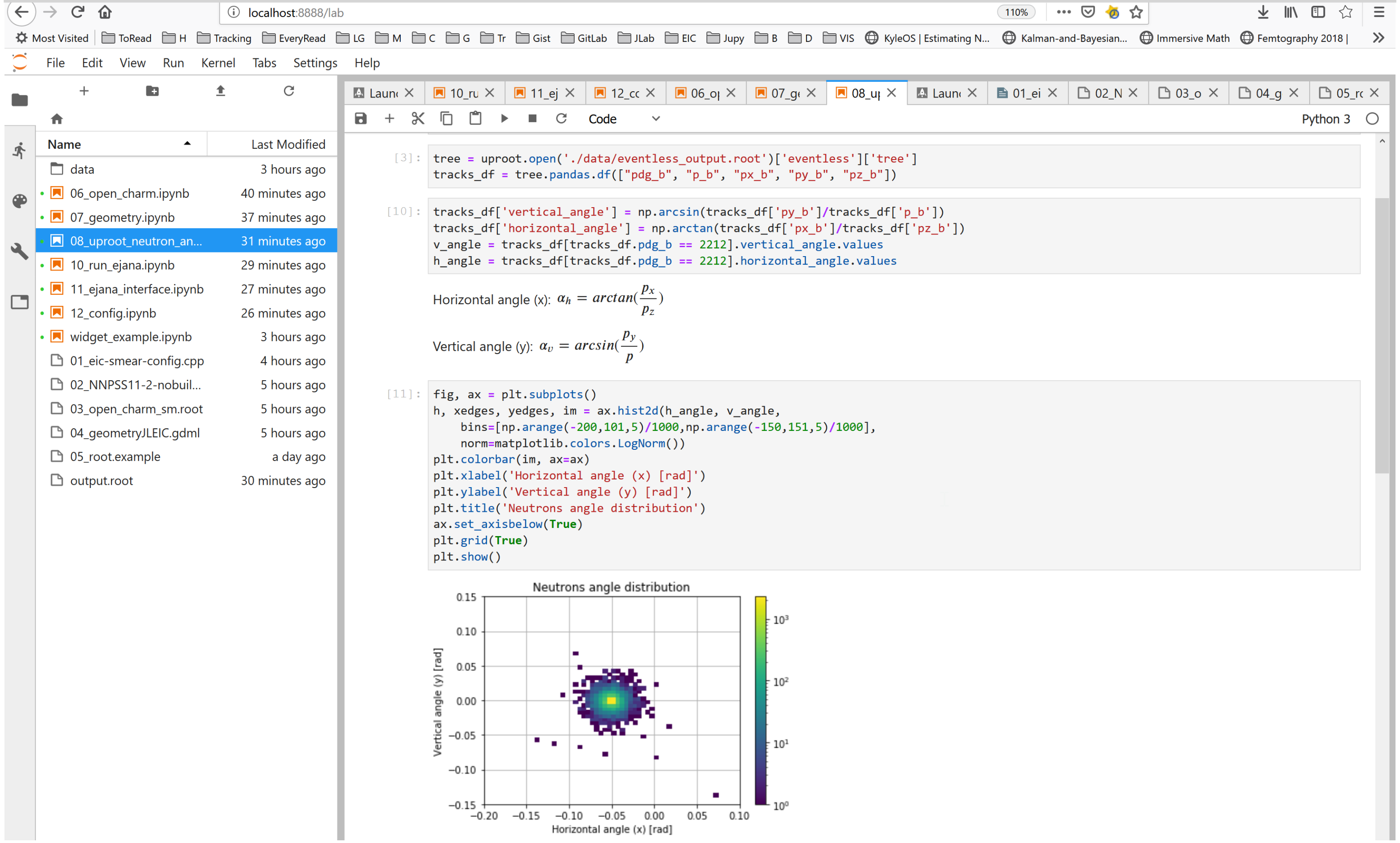
Important links

Mailing list	eicug-software@eicug.org (subscribe via Google Group)
Repository	http://gitlab.com/eic
Website	https://software.eicug.org

Collaborative workspace for EICUG

JupyterLab

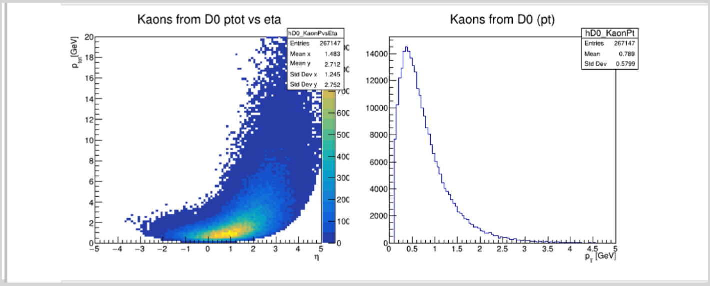
- web-based interactive analysis environment



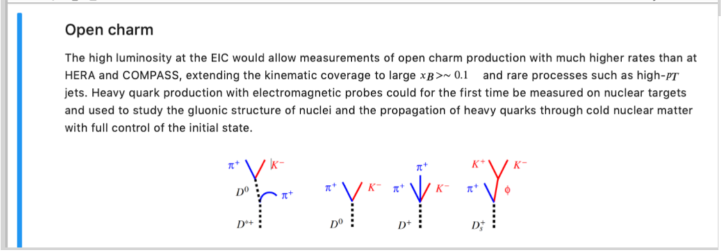
Jupyter Notebooks

- writing analysis code

- visualization of results



- narrative of the analysis



Tutorials

Planned Tutorials (in-person at BNL and/or JLAB, remote, record video)

Jan. 9 **Tutorial on fast simulations**

Jan. 29 **Tutorial on detector simulations** (implement and integrate subdetector in existing detector concepts, modify detector concept)

Continuing tutorials according to requested topics

January 2020

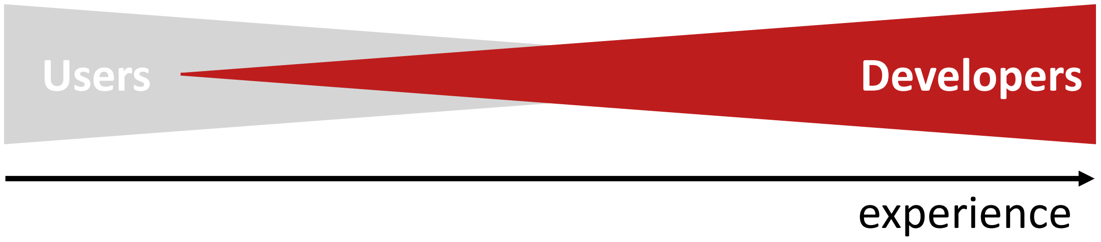
Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	**9**	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	**29**	**30***	**31***	

* **EIC Generic Detector R&D**

Support

support team

weekly shifts



software-support@eicug.org

Mailing list (anyone can contact)
Google forum (for archive of support requests and start of knowledge base)

http://eicug.slack.com/

EICUG Slack workspace with software-support channel

invitation

Section
Software design

Design goal and considerations

Design goal

Workflow environment for EICUG

- **to use** (tools, documentation, support) **and**
- **to grow with user input** (direction, documentation, tools)

Design considerations

Modular design

- **Future compatibility** modular design with structures robust against likely changes in computing environment so that changes in underlying code can be handled without an entire overhaul of the structure
- **Collaborative approach** the more modular, the easier to contribute

User-centered design

- scientists of all levels should be enabled to actively participate in EIC Physics and Detector Conceptual Development
- need for analysis toolkits using modern and advanced technologies while hiding that complexity
- resolving this tension means putting a priority on the user experience and functionality

JupyterLab environment

- **bridge to modern data science**, e.g.,
  - *Nature* **563**, 145-146 (2018): “Why Jupyter is data scientists’ computational notebook of choice”
  - more than three million Jupyter Notebooks publicly available on GitHub

- **collaborative workspace** to create and share Jupyter Notebooks

- **web-based interactive analysis environment** accessible, consistent, reproducible analyses

- **fully extensible and modular** build a collection of analyses and analysis tools

Jupyter Notebooks

- writing analysis code

```
[4]: jana.plugin('hepmc_reader') \
     .plugin('jana', nevents=10000, output='hepmc_sm.root') \
     .plugin('eic_smear', detector='jleic') \
     .plugin('open_charm')
```

[4]: **eJana** configured
plugins: hepmc_reader,eic_smear,open_charm

Python

```
[5]: jana.source('../data/herwig6_20k.hepmc')
```

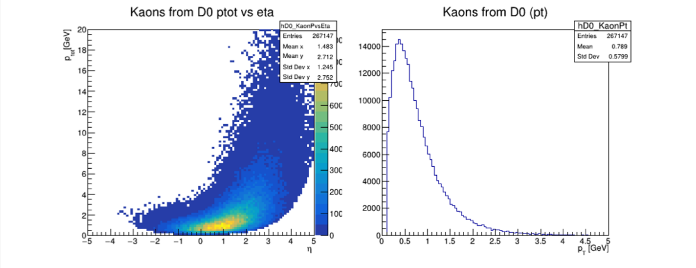
[5]: **eJana** configured
plugins: hepmc_reader,eic_smear,open_charm
sources:
../data/herwig6_20k.hepmc

Root/C++

```
[6]: jana.run()
```

Total events processed: 10001 (~ 10.0 kevt)

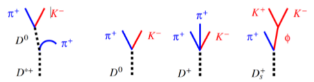
- visualization of results



- narrative of the analysis

Open charm

The high luminosity at the EIC would allow measurements of open charm production with much higher rates than at HERA and COMPASS, extending the kinematic coverage to large $x_B >\sim 0.1$ and rare processes such as high-p_T jets. Heavy quark production with electromagnetic probes could for the first time be measured on nuclear targets and used to study the gluonic structure of nuclei and the propagation of heavy quarks through cold nuclear matter with full control of the initial state.



Modular design

Escaping complexity scaling trap

- provide interfaces to internal layers
- interaction between layers must be clear

Modularity each layer must be replaceable

simple	JupyterLab web interface
moderate	analysis scripts, python
complex	eJANA, plugins, C++
expert	JANA, eic-smear, *fun4all*, ROOT, Geant4

Section

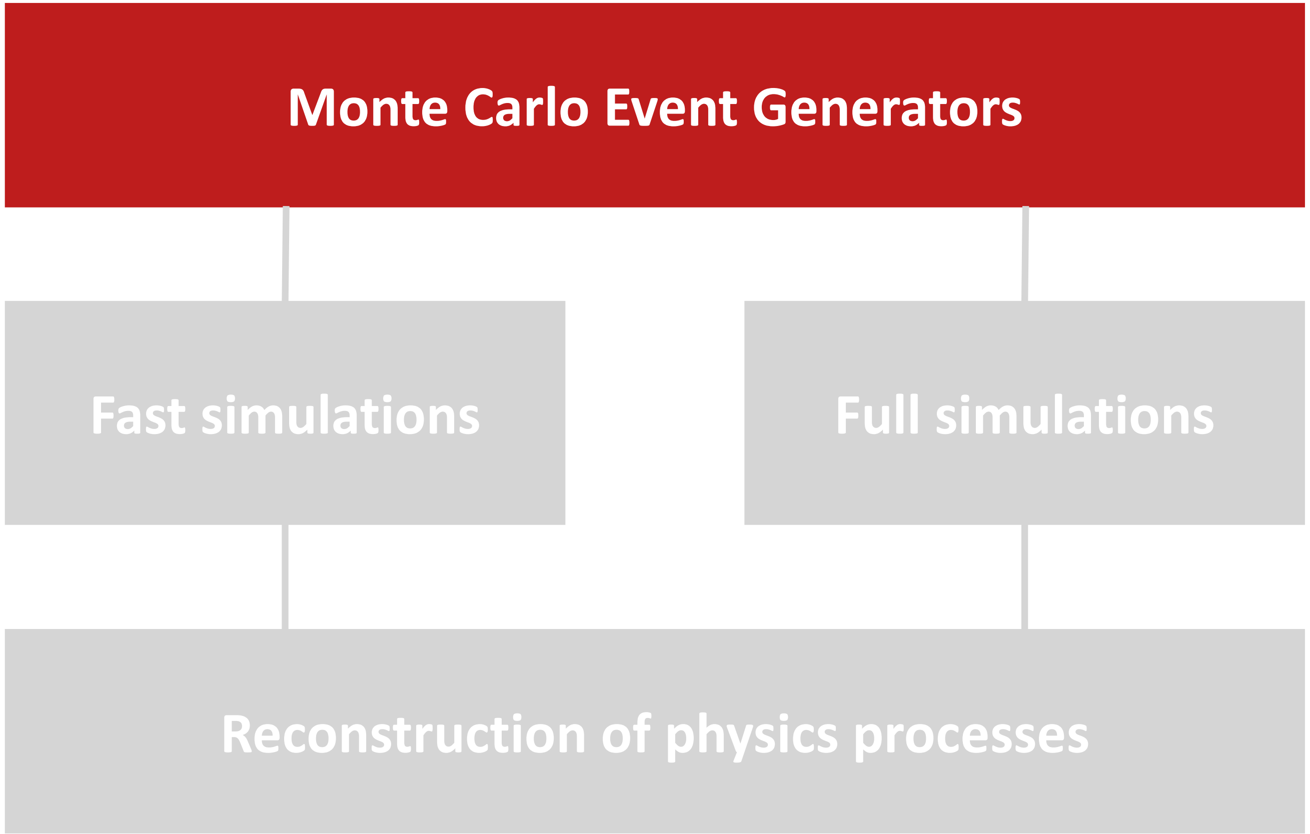
Status of EICUG simulations

Simulations of physics processes and detector responses

Simulation of physics processes

Simulation of detector responses

Physics analysis



Broad collection of event generators used for EIC

Monte Carlo Event Generators (MCEG)

The following event generators are available:

- ep
  - DJANGOH: (un)polarised DIS generator with QED and QCD radiative effects for NC and CC events.
  - gmc_trans: A generator for semi-inclusive DIS with transverse-spin- and transverse-momentum-dependent distributions.
  - LEPTO: A leptoproduction generator - used as a basis for PEPSI and DJANGOH
  - LEPTO-PHI: A version of LEPTO with "Cahn effect" (azimuthal asymmetry) implemented
  - MILOU: A generator for deeply virtual Compton scattering (DVCS), the Bethe-Heitler process and their interference.
  - PYTHIA: A general-purpose high energy physics event generator.
  - PEPSI: A generator for polarised leptoproduction.
  - RAPGAP: A generator for deeply inelastic scattering (DIS) and diffractive $e + p$ events.
- eA
  - BeAGLE: Benchmark eA Generator for LEptoproduction - UNDER CONSTRUCTION - a generator to simulate ep/eA DIS events including nuclear shadowing effects (based on DPMJetHybrid)
  - DPMJet: a generator for very low Q2/real photon physics in eA
  - DPMJetHybrid: a generator to simulate ep/eA DIS events by employing PYTHIA in DPMJet
  - Sartre is an event generator for exclusive diffractive vector meson production and DVCS in ep and eA collisions based on the dipole model.

From https://wiki.bnl.gov/eic/index.php/Simulations and available in https://gitlab.com/eic/mceg

MCEG R&D for EIC

Unique MCEG requirements for EIC Science

- MCEG for polarized ep, ed, and eHe^3
  - including novel QCD phenomena: GPDs, TMDs
- MCEG for eA

MCEG community

- focus of last two decades: **LHC**
  - **lesson learned** high-precision QCD measurements require high-precision MCEGs
  - MCEG not about tuning but about physics
- ready to work on ep/eA

MCEG R&D for EIC

General-purpose MCEGs, HERWIG, PYTHIA, and SHERPA, will be significantly improved w.r.t. MCEGs at HERA time:

- MCEG-data comparisons in Rivet will be critical to tune the MCEGs to DIS data and theory predictions.
- The existing general-purpose MCEG should soon be able to simulate NC and CC unpolarized observables also for eA. A precise treatment of the nucleus and, e.g., its breakup is needed.
- First parton showers and hadronization models for ep with spin effects, but far more work needed for polarized ep / eA simulations.
- Need to clarify the details about merging QED+QCD effects (in particular for eA).

MCEG for eA

- **pioneering projects** BeAGLE, spectator tagging in ed, Sartre
- **active development** eA adaptation of JETSCAPE, Mueller dipole formalism in Pythia8 (ala DIPSY)

TMD physics

- Vibrant community working on various computational tools for TMDs.
- CASCADE: MCEG for unpolarized TMDs (unintegrated TMDs) at high energy.
- Need more verification of MCEG models with TMD theory / phenomenology.

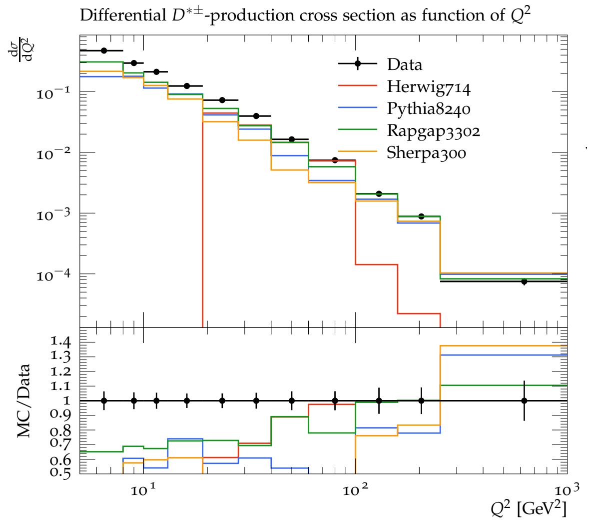
MCEG for ep We are on a very good path, but still quite some work ahead.
MCEG for eA Less clear situation about theory and MCEG.

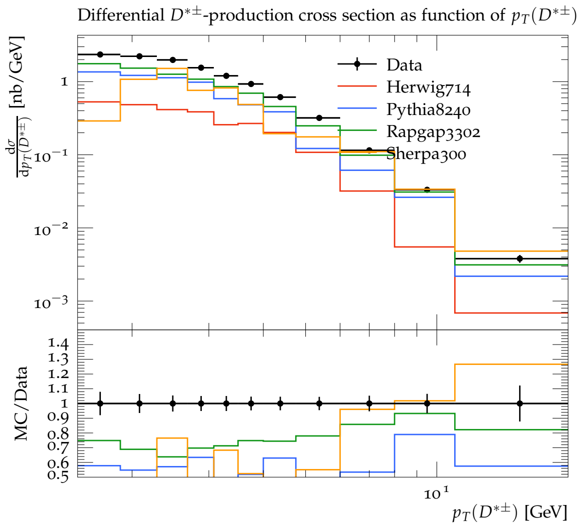
MCEG–HERA comparisons and MCEG validation for ep

MCEG R&D requires *easy* access to *data*

- data := analysis description + data points

HEP existing workflow for MCEG R&D using tools such as HZTool, Rivet and Professor





Detailed comparisons between modern MCEG and HERA data

- workshop on [Rivet for ep]() (Feb 18—20 2019)
- mailing list [rivet-ep-l@lists.bnl.gov]()
- HERA data not (yet) included in MCEG tunes

Rivet example

SIDIS analysis at HERMES

```
    // Extract the particles other than the lepton
    const FinalState& fs = apply<FinalState>(event, "FS");
    Particles particles;
    particles.reserve(fs.particles().size());
    const GenParticle* dislepGP = dl.out().genParticle();
    foreach (const Particle& p, fs.particles()) {
      const GenParticle* loopGP = p.genParticle();
      if (loopGP == dislepGP)
        continue;
      particles.push_back(p);
    }

    // Apply HERMES cuts.
    bool validx = (x > 0.023 && x < 0.6);
    if (q2 < 1. || w2 < 10. || y < 0.1 || y > 0.85 || !validx)
      vetoEvent;

    // good inclusive event, let's do bookkeeping before we look at the hadrons
    dis_tot += weight;
    dis_x->fill(x, weight);
    dis_Q2->fill(q2, weight);

    for (size_t ip1 = 0; ip1 < particles.size(); ++ip1) {
      const Particle& p = particles[ip1];

      // get the particle index, check if it is a particle of interest
      const int part_idx = get_index(p.genParticle()->pdg_id());
      if (part_idx < 0) {
        continue;
      }

      // we have a particle of interest, let's calculate the kinematics
      // z
      const double z = (p.momentum() * pProton) / (pProton * q);
      // pt
      const double pth = sqrt(p.momentum().pT2());

      // get our z index, if negative, we have a particle outside of [.2, .8]
      const int z_idx = calc_zslice(z);
      if (z_idx < 0) {
        continue;
      }

      // store the events and make cuts where necessary
      //
      // pt cut for variables not binned in pt
      if (pth > 0 && pth < 1.2) {
        mult_z[part_idx]->fill(z, weight);
        mult_zx[part_idx][z_idx]->fill(x, weight);
        mult_zQ2[part_idx][z_idx]->fill(q2, weight);
      }
      mult_zpt[part_idx][z_idx]->fill(pth, weight);
    }
```

Online catalogue for MCEGs (in preparation)

- **Categories** ep, eA, radiative effects
- Name
- Contact information
- **Brief Description** What processes are described? What is unique about the MCEG? Include version number as reference.
- **References (links)** website, repository, documentation, container, validation plots

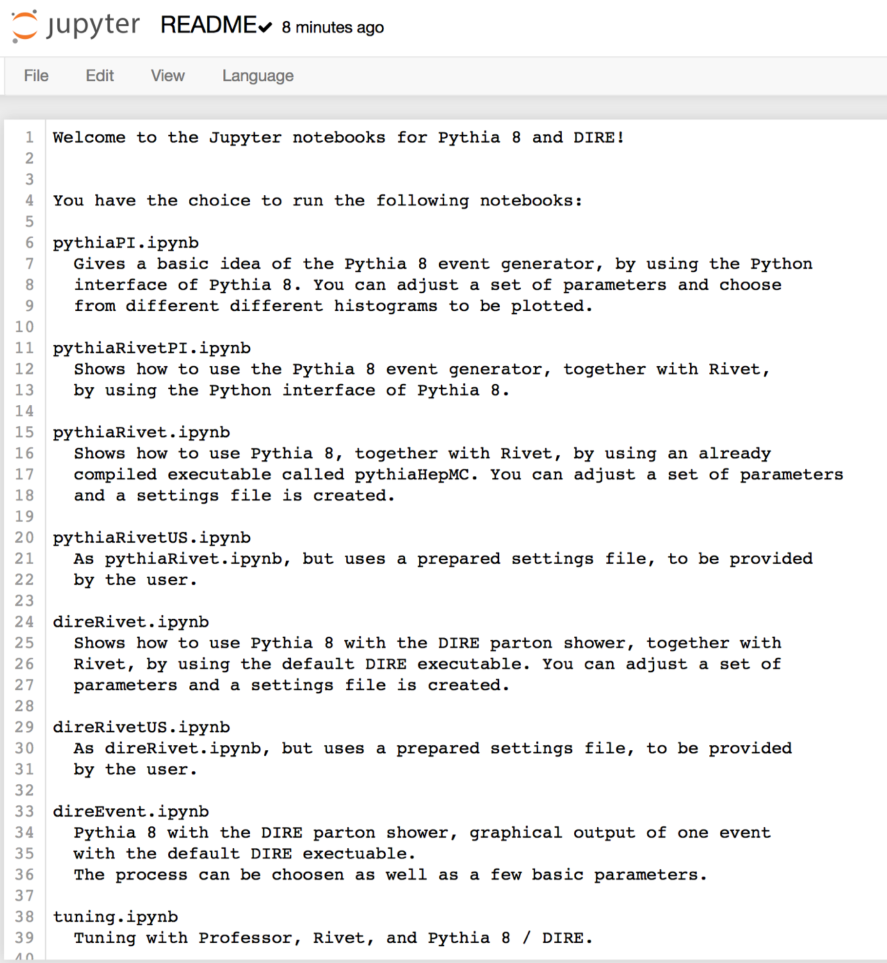
Example

- **Category** ep, eA, exclusive vector meson production, general photoproduction
- **Name** eSTARlight
- **Contact Information** Spencer Klein, srklein@lbl.gov
- **Brief description** eSTARlight simulates coherent photoproduction and electroproduction of vector mesons in ep and eA collisions. It can simulate a variety of different vector mesons, and it also includes an interface to DPMJET, which allows for general simulation of photonuclear interactions. It internally simulates most simple (2-body) vector meson decays with a correct accounting for the initial photon polarization (transverse for Q^2 ~ 0, with an increasing longitudinal component with increasing Q^2) in the angular distributions of the final state. It can also interface to PYTHIA8 to simulate more complicated decays.
- **References** The code is freely available from https://estarlight.hepforge.org/ The Readme file includes a fairly comprehensive users manual. The physics behind the code is documented in M. Lomnitz and S. Klein, Phys. Rev. C99, 015203 (2019).

Example MC simulations will be available on **HepSim** for benchmarks and validation (more and more examples added).

JupyterLab integration of MCEG (ongoing)

Example: Container for Pythia8+DIRE



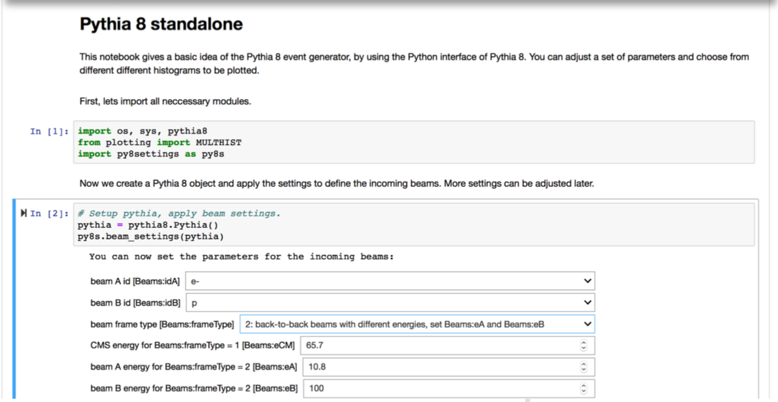
jupyter README 8 minutes ago

Logout

File Edit View Language Plain Text

```
Welcome to the Jupyter notebooks for Pythia 8 and DIRE!


You have the choice to run the following notebooks:

pythiaPI.ipynb
  Gives a basic idea of the Pythia 8 event generator, by using the Python
  interface of Pythia 8. You can adjust a set of parameters and choose
  from different different histograms to be plotted.

pythiaRivetPI.ipynb
  Shows how to use the Pythia 8 event generator, together with Rivet,
  by using the Python interface of Pythia 8.

pythiaRivet.ipynb
  Shows how to use Pythia 8, together with Rivet, by using an already
  compiled executable called pythiaHepMC. You can adjust a set of parameters
  and a settings file is created.

pythiaRivetUS.ipynb
  As pythiaRivet.ipynb, but uses a prepared settings file, to be provided
  by the user.

direRivet.ipynb
  Shows how to use Pythia 8 with the DIRE parton shower, together with
  Rivet, by using the default DIRE executable. You can adjust a set of
  parameters and a settings file is created.

direRivetUS.ipynb
  As direRivet.ipynb, but uses a prepared settings file, to be provided
  by the user.

direEvent.ipynb
  Pythia 8 with the DIRE parton shower, graphical output of one event
  with the default DIRE exectuable.
  The process can be choosen as well as a few basic parameters.

tuning.ipynb
  Tuning with Professor, Rivet, and Pythia 8 / DIRE.
```

Jupyter notebook interface



Pythia 8 standalone

This notebook gives a basic idea of the Pythia 8 event generator, by using the Python interface of Pythia 8. You can adjust a set of parameters and choose from different different histograms to be plotted.

First, lets import all neccessary modules.

In [1]:
```
import os, sys, pythia8
from plotting import MULTHIST
import py8settings as py8s
```

Now we create a Pythia 8 object and apply the settings to define the incoming beams. More settings can be adjusted later.

In [2]:
```
# Setup pythia, apply beam settings.
pythia = pythia8.Pythia()
py8s.beam_settings(pythia)
```

You can now set the parameters for the incoming beams:

beam A id [Beams:idA] e-

beam B id [Beams:idB] p

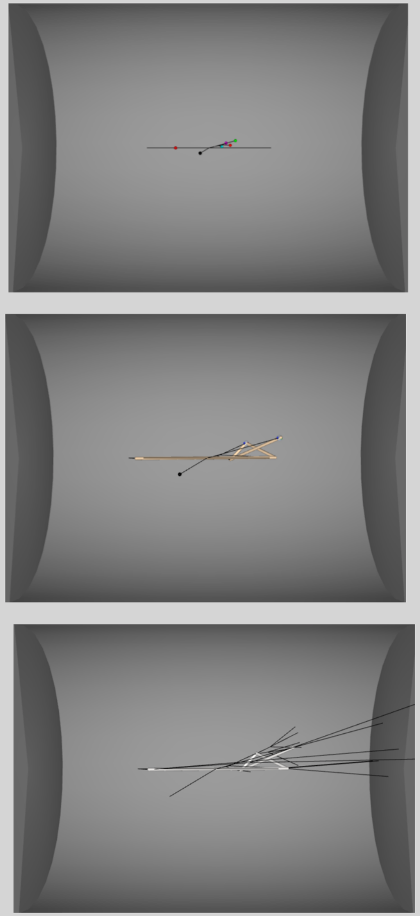
beam frame type [Beams:frameType] 2: back-to-back beams with different energies, set Beams:eA and Beams:eB

CMS energy for Beams:frameType = 1 [Beams:eCM] 65.7

beam A energy for Beams:frameType = 2 [Beams:eA] 10.8

beam B energy for Beams:frameType = 2 [Beams:eB] 100

Visualization of ep collision

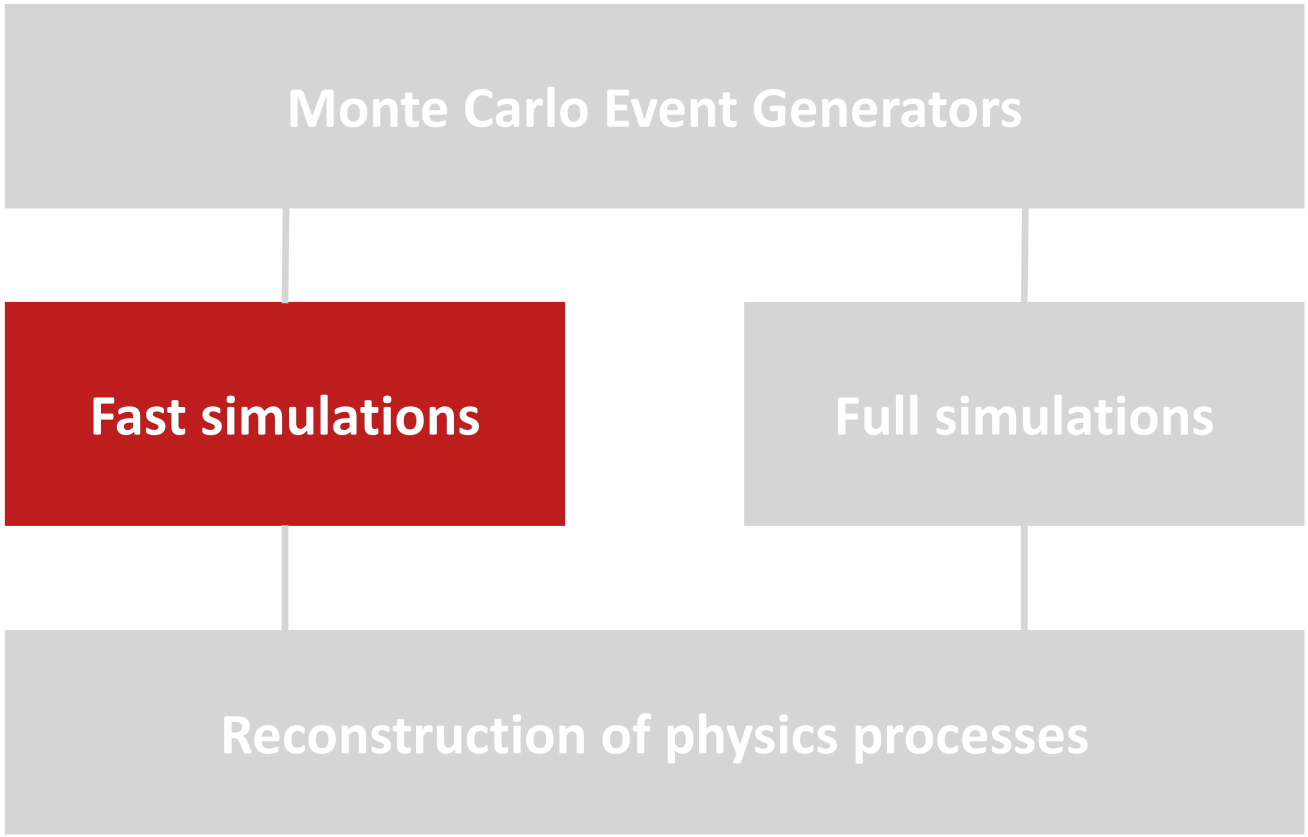


Simulations of physics processes and detector responses

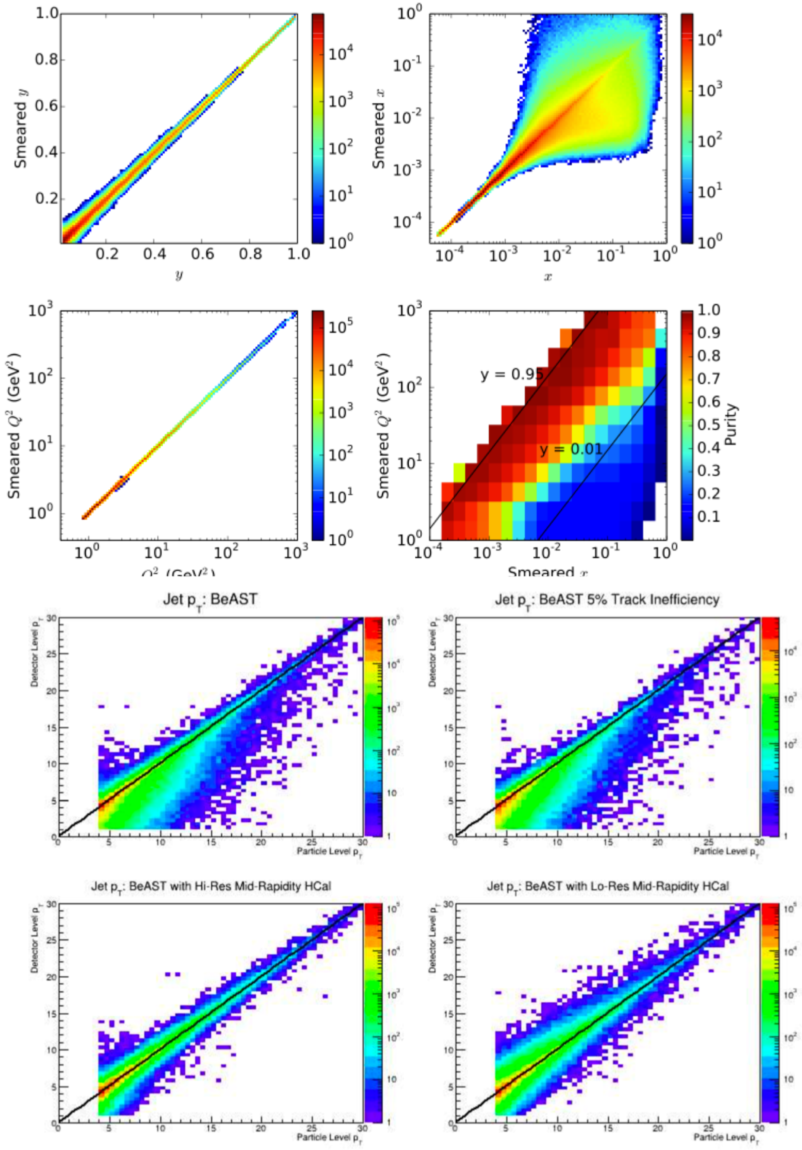
Simulation of physics processes

Simulation of detector responses

Physics analysis



eic-smear



Fast simulations using ROOT, ideal for questions like

- *"Given a (known) detector performance, how well can I measure some physics observable(s)?"*
- *"If I need to measure X to some precision, what detector performance do I need?"*
- Used extensively for **EIC White Paper**

Features

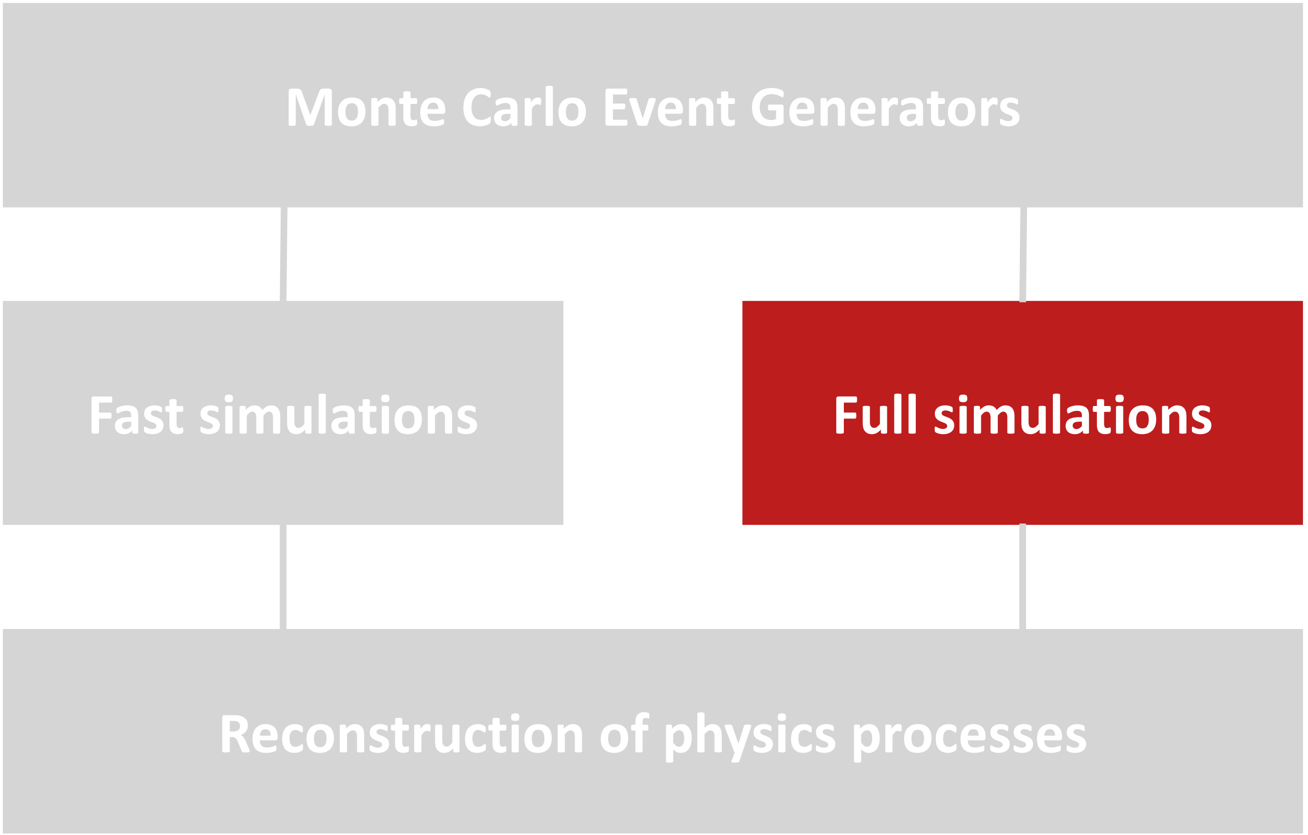
- interface to MCEGs for ep and eA
- smearing of overall detector performance:
  - can be easily modified in user code
  - includes acceptance effects
  - parametrizations for eRHIC (BeAST, ePHENIX), JLEIC and others
- ROOT trees for MC Truth and smeared information

Simulations of physics processes and detector responses

Simulation of physics processes

Simulation of detector responses

Physics analysis



Accelerator interface

Accelerator design (beam elements)

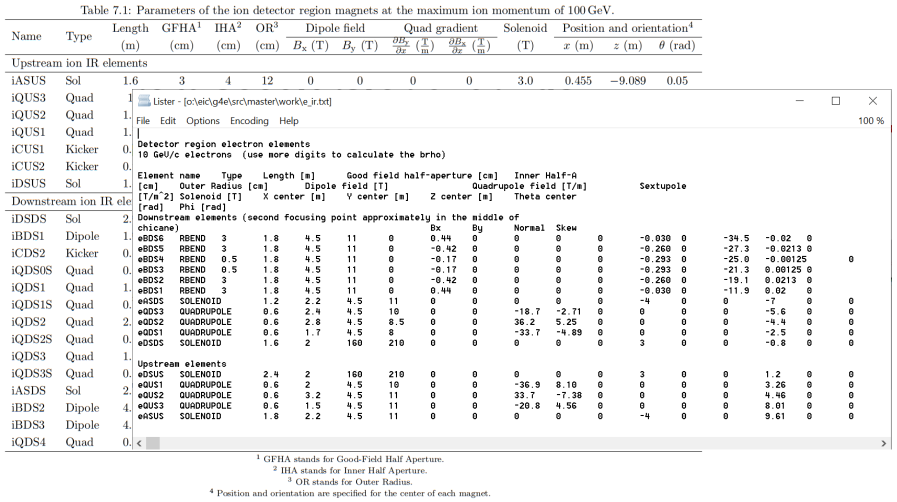


Table 7.1: Parameters of the ion detector region magnets at the maximum ion momentum of 100 GeV.

Name	Type	Length (m)	GFHA[1] (cm)	IHA[2] (cm)	OR[3] (cm)	Dipole field B_x (T)	Dipole field B_y (T)	Quad gradient $\frac{\partial B_y}{\partial x}$ $(\frac{T}{m})$	Quad gradient $\frac{\partial B_x}{\partial x}$ $(\frac{T}{m})$	Solenoid (T)	Position and orientation[4] x (m)	z (m)	θ (rad)
Upstream ion IR elements													
iASUS	Sol	1.6	3	4	12	0	0	0	0	3.0	0.455	−9.089	0.05
iQUS3	Quad	1											
iQUS2	Quad	1.											
iQUS1	Quad	1.											
iCUS1	Kicker	0.											
iCUS2	Kicker	0.											
iDSUS	Sol	1.											
Downstream ion IR ele													
iDSDS	Sol	2.											
iBDS1	Dipole	1.											
iCDS2	Kicker	0.											
iQDS0S	Quad	0.											
iQDS1	Quad	1.											
iQDS1S	Quad	0.											
iQDS2	Quad	2.											
iQDS2S	Quad	0.											
iQDS3	Quad	1.											
iQDS3S	Quad	0.											
iASDS	Sol	2.											
iBDS2	Dipole	4.											
iBDS3	Dipole	4.											
iQDS4	Quad	0.											

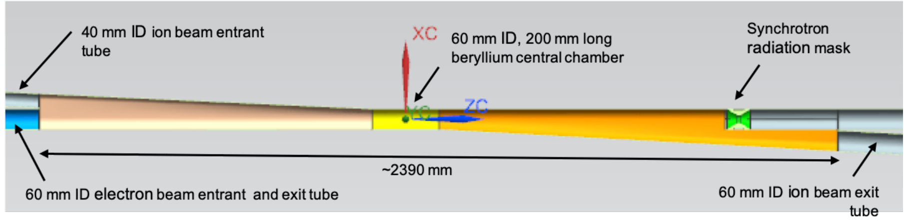
[1] GFHA stands for Good-Field Half Aperture.
[2] IHA stands for Inner Half Aperture.
[3] OR stands for Outer Radius.
[4] Position and orientation are specified for the center of each magnet.

Lister - [o:\eic\g4e\src\master\work\e_ir.txt]

File Edit Options Encoding Help 100 %

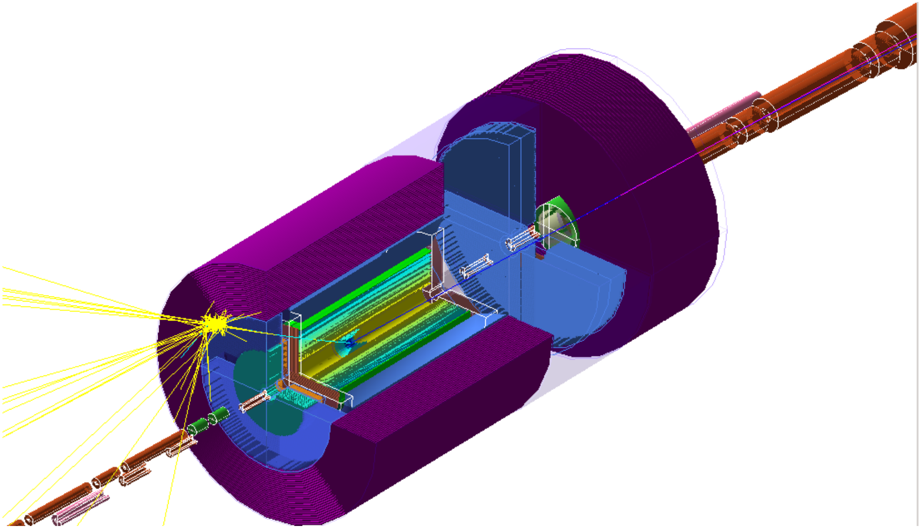
```
Detector region electron elements
10 GeV/c electrons  (use more digits to calculate the brho)

Element name    Type    Length [m]      Good field half-aperture [cm]   Inner Half-A
[cm]    Outer Radius [cm]       Dipole field [T]                Quadrupole field [T/m]          Sextupole
[T/m^2] Solenoid [T]    X center [m]    Y center [m]    Z center [m]    Theta center
[rad]   Phi [rad]
Downstream elements (second focusing point approximately in the middle of
chicane)                                        Bx      By      Normal  Skew
eBDS6   RBEND   3       1.8     4.5     11      0       0.44    0       0       0       0       -0.030  0       -34.5   -0.02   0
eBDS5   RBEND   3       1.8     4.5     11      0       -0.42   0       0       0       0       -0.260  0       -27.3   -0.0213 0
eBDS4   RBEND   0.5     1.8     4.5     11      0       -0.17   0       0       0       0       -0.293  0       -25.0   -0.00125        0
eBDS3   RBEND   0.5     1.8     4.5     11      0       -0.17   0       0       0       0       -0.293  0       -21.3   0.00125 0
eBDS2   RBEND   3       1.8     4.5     11      0       -0.42   0       0       0       0       -0.260  0       -19.1   0.0213  0
eBDS1   RBEND   3       1.8     4.5     11      0       0.44    0       0       0       0       -0.030  0       -11.9   0.02    0
eASDS   SOLENOID        1.2     2.2     4.5     11      0       0       0       0       0       -4      0       0       -7      0       0
eQDS3   QUADRUPOLE      0.6     2.4     4.5     10      0       0       -18.7   -2.71   0       0       0       0       -5.6    0       0
eQDS2   QUADRUPOLE      0.6     2.8     4.5     8.5     0       0       36.2    5.25    0       0       0       0       -4.4    0       0
eQDS1   QUADRUPOLE      0.6     1.7     4.5     8       0       0       -33.7   -4.89   0       0       0       0       -2.5    0       0
eDSDS   SOLENOID        1.6     2       160     210     0       0       0       0       0       3       0       0       -0.8    0       0

Upstream elements
eDSUS   SOLENOID        2.4     2       160     210     0       0       0       0       0       3       0       0       1.2     0       0
eQUS1   QUADRUPOLE      0.6     2       4.5     10      0       0       -36.9   8.10    0       0       0       0       3.26    0       0
eQUS2   QUADRUPOLE      0.6     3.2     4.5     11      0       0       33.7    -7.38   0       0       0       0       4.46    0       0
eQUS3   QUADRUPOLE      0.6     1.5     4.5     11      0       0       -20.8   4.56    0       0       0       0       8.01    0       0
eASUS   SOLENOID        1.8     2.2     4.5     11      0       0       0       0       0       -4      0       0       9.61    0       0
```

Engineering Design (CAD)



Detector Simulations (Geant4)

Tuning



Status

eRHIC and JLEIC information available
Common interface under active development

Detector Simulation

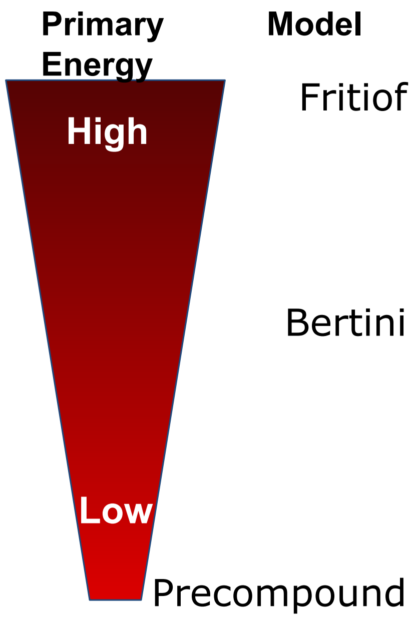
- **collaboration with Geant4 International Collaboration**
  - **liaison** Makoto Asai (SLAC)
- **Geant4 for EIC**
  - coordinate input for Geant4 validation based on EIC physics list maintained by (former) SLAC Geant4 group
  - Geant4 10.6 recommended (released Dec. 6)

09/24 Geant4 Technical Forum on EIC

- EIC detector and physics simulations rely on Geant4
- knowledge transfer (e.g., sub-event parallelism or tessellated solids)
- maintain EIC physics lists
- **request** improved photo-nuclear and electro-nuclear reactions

EIC

- energy range is different from LHC
- validation, tuning and extension including test beam studies



Geant4 infrastructure for EICUG

Requirements

- **EIC Generic Detector R&D program (T. Ullrich)** *"a simple lite setup with a well defined geometry description standard that is easy to use"*
- **EICUG** Flexible accelerator and detector interface with full support of eRHIC and JLEIC parameters and IR designs and existing detector concepts

Approach

- common repository for detector R&D for EIC
- common detector description in Geant4 (C++) and not yet DD4hep (sub-detectors developed in Geant4 (C++))
- common detector naming convention for EIC
- possible common hits output structure
- concise document and template on how to implement and integrate subdetector in EIC detector concepts

Discussion

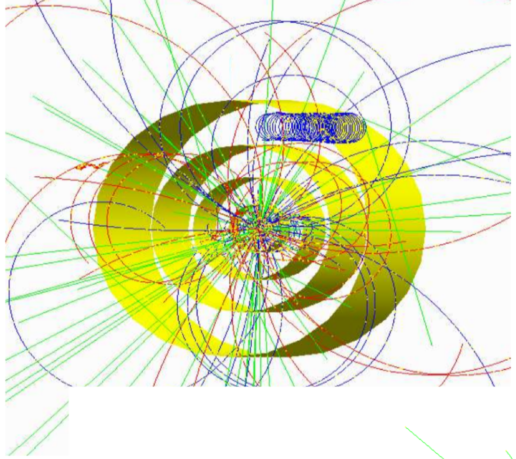
- **two in-person meetings**
  - 07/10 EIC Software Meeting at BNL (minutes)
  - 09/24 EIC Software Meeting at JLAB (minutes)
- **evaluation** 09/30, 10/21, 10/28, 11/18, 11/25

Two solutions proposed

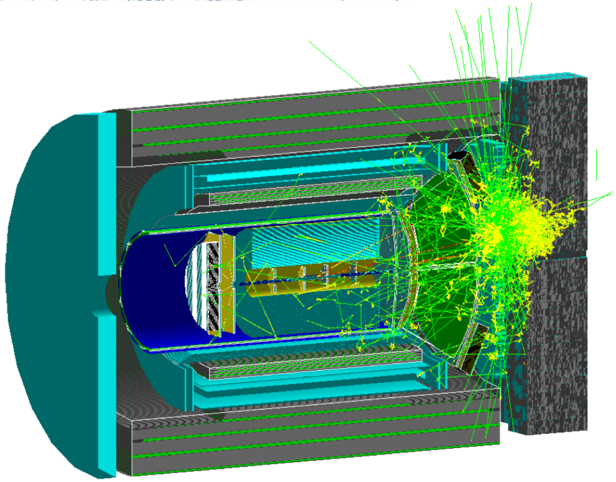
1. detector simulations in **fun4all**, major update for common EIC simulations (e.g., to Geant4 10.5)
2. Geant4 application **g4e**

Fun4All + GEANT4

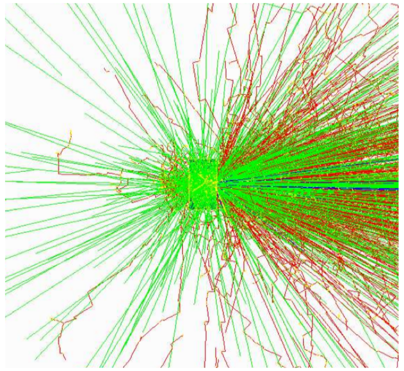
- Mature Framework based on ROOT, steering with ROOT macros
- Modular – each detector is its own entity
- No central code needs to be modified when adding new detectors
- Detectors are combined using ROOT macros
- Distribution as singularity container + libraries in cvmfs*
- Daily builds + Continuous Integration
- No geometry model enforced
- Interface to eic-smear: most EIC specific Event generators accessible
- Pre-canned configurations for EIC-sPHENIX and partial JLEIC
- Used to provide input for our EIC detector LOI**
- Generic Volumes (box, cylinder, cone) can be implemented no macro level



Pythia8 in a six layer silicon detector mockup and 2T field



Sarte as seen by an EIC detector



10 GeV Au on water phantom (NASA Space Radiation Lab)

*Installation: https://github.com/EIC-Detector/Singularity

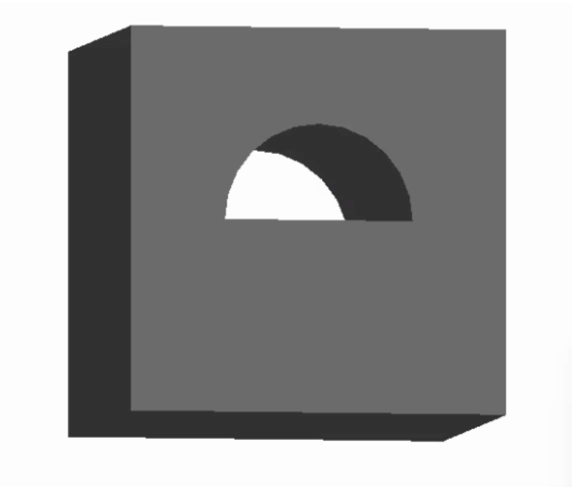
**https://arxiv.org/pdf/1402.1209.pdf
https://indico.bnl.gov/event/5283/attachments/20546/27556/eic-sphenix-dds-final-2018-10-30.pdf

For details: see selected Fun4All presentations https://www.phenix.bnl.gov/WWW/publish/pinkenbu/EIC/

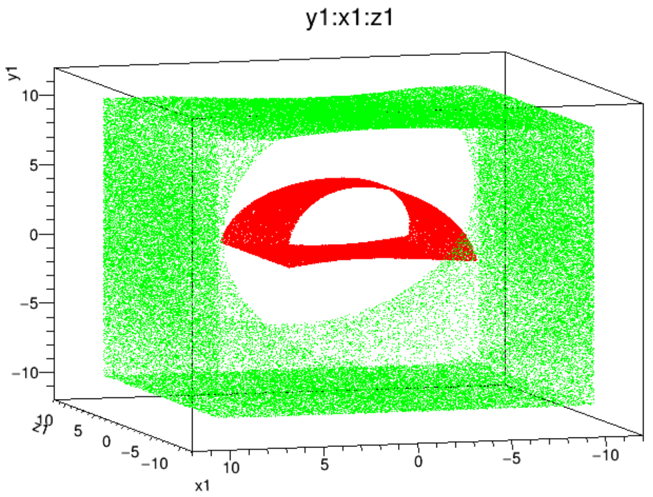
Implementing a Detector in Fun4All

Simplest Example, more sophisticated to come:
https://github.com/EIC-Detector/g4exampledetector:
simple/source: Simplest case - everything hardcoded, only active volumes
simple/macro: Fun4All_G4_Example01.C to run the show (and save Hits in ntuple)

Let's call your detector PDirc*, 3 classes need to be implemented :
G4PDircSubsystem → interface between Fun4All and Detector
G4PDircDetector → GEANT4 Construct method
G4PDircSteppingAction → select which quantities to store for each hit
*Detector names can be set on the command line but you do not want identically named sources



Example01: block with ½ cylindrical hole



Geantino Scan to verify geometry using entry/exit coordinates of geantino tracks

You will find that help will always be given at Hogwarts to those who ask for it.

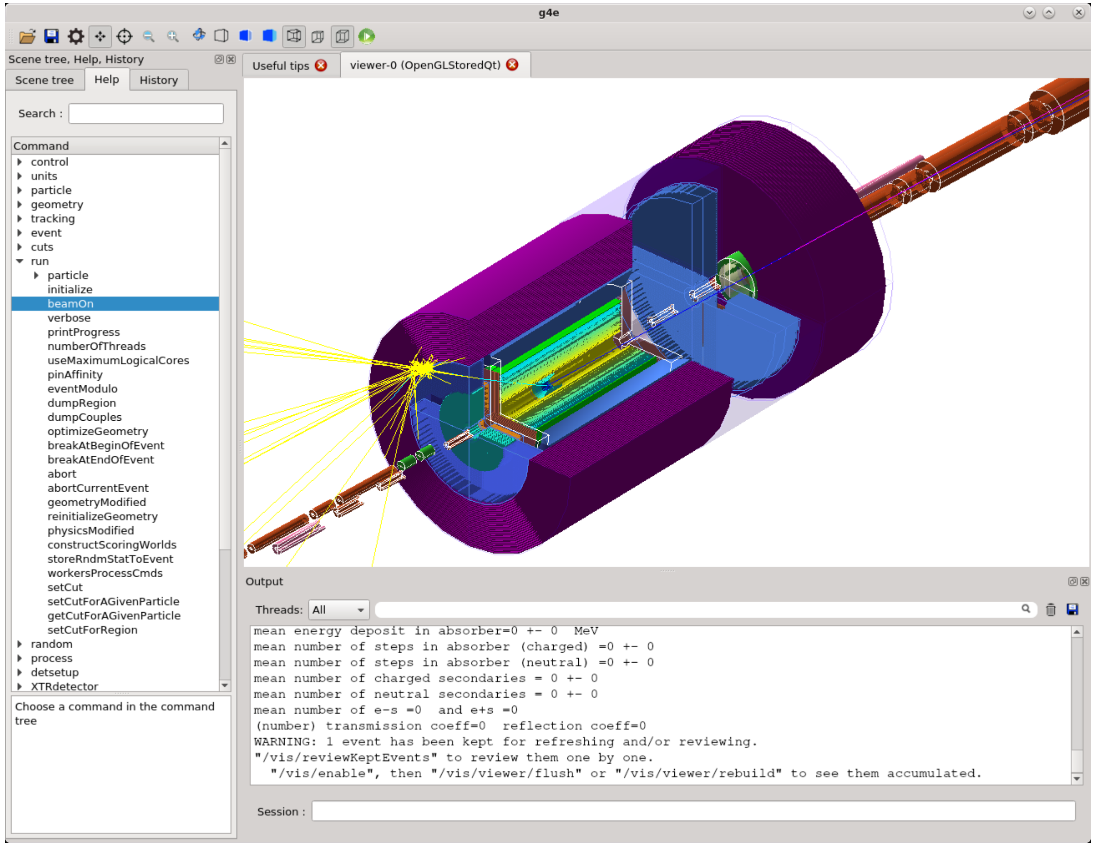
– Dumbledore

Tutorials:
https://github.com/EIC-Detector/tutorials

Join slack channel for support:
https://join.slack.com/t/eic-design-study/signup

Email:
Chris Pinkenburg pinkenburg@bnl.gov
Jin Huang jhuang@bnl.gov

G4E (Geant 4 EIC)



Standalone C++ Geant4 application

Various EIC MC file formats
Beagle, Pythia6, HEPMC - Pythia8, Herwig and others

Integration with accelerator elements

Infrastructure to import Geant4 detector geometry and simulation code

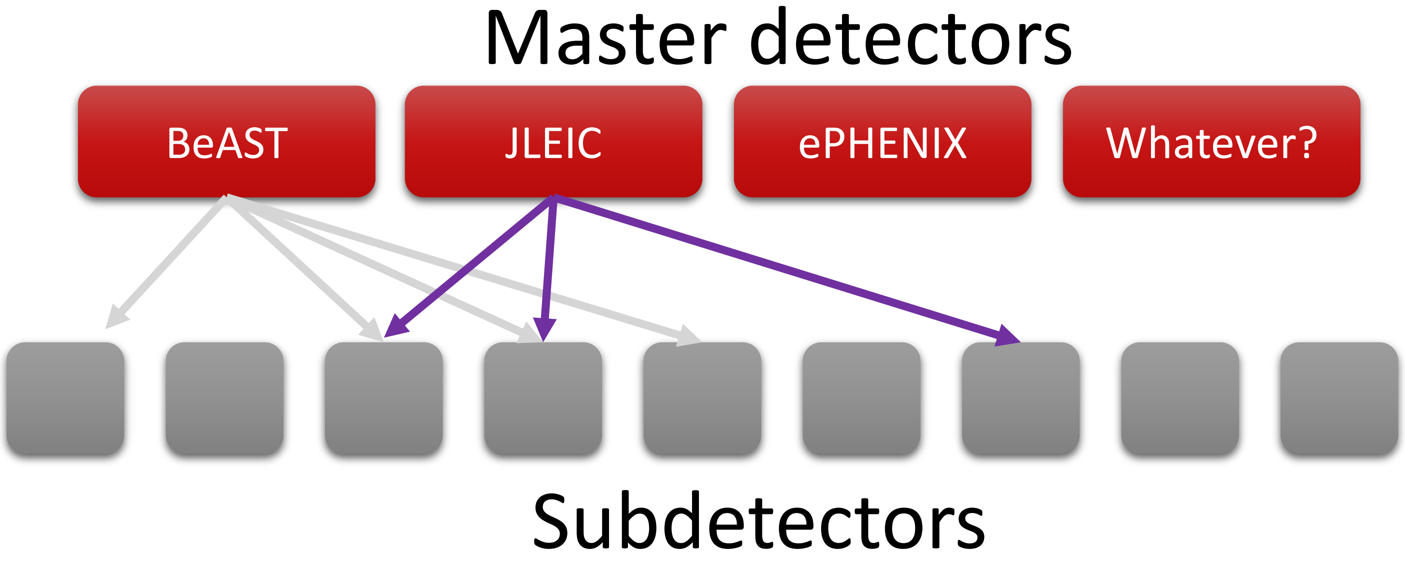
Subdetector integration in G4E

G4E goals

- Ability to have several “master” detectors (and IRs) such as BeAST, JLEIC etc.
- Ability to easily import Geant4 standalone detector implementations
- Ways to send the “right” configuration to a subdetector depending on selected “master” detector, so that one detector implementation could serve for different IRs
- **Stay as close to Geant4 as possible, to stay convenient for Geant4 experts**

To import a subdetector:

- *SubDetectorInterface* class implementation
- Subscribe to various standard Geant 4 actions *(SteppingAction, StackingAction, etc)*
- Define subdetector’s place in one or many “master” detectors

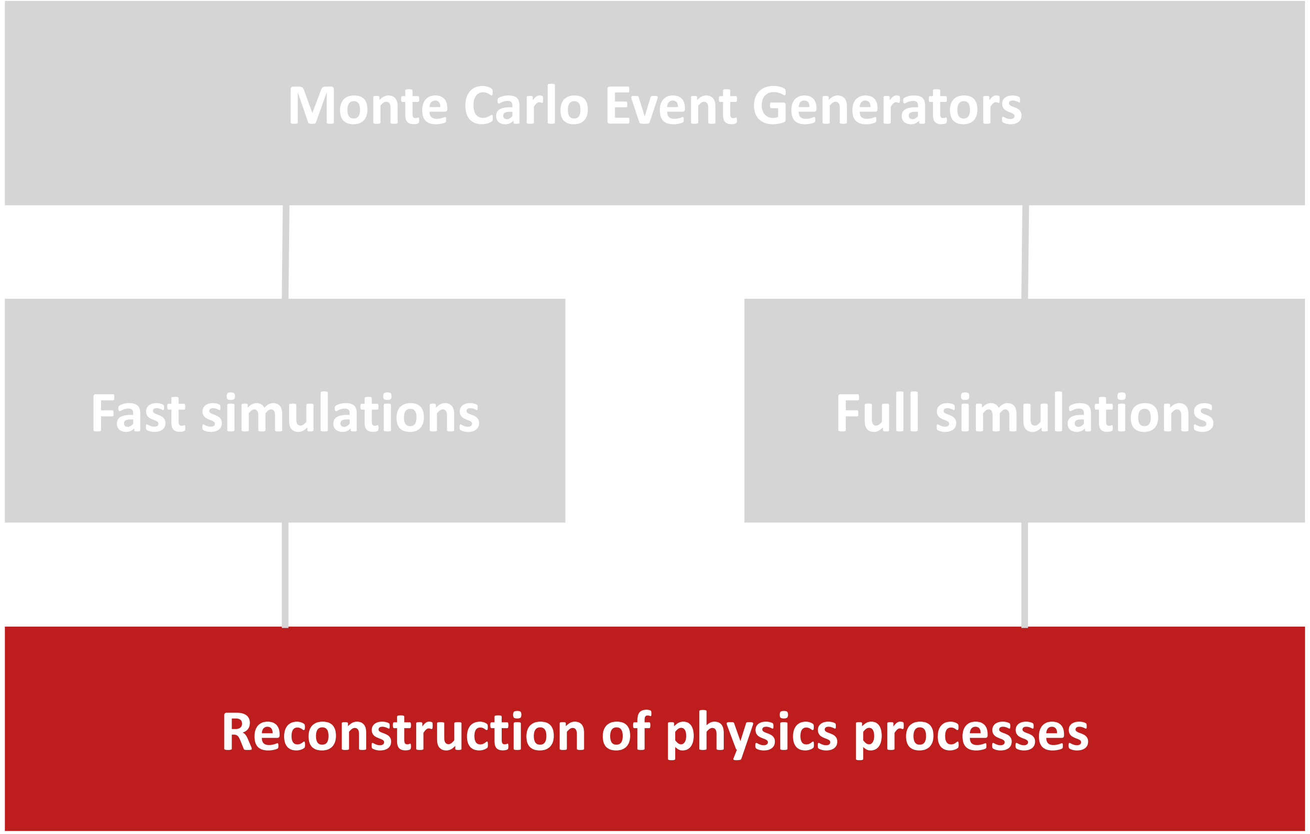


Simulations of physics processes and detector responses

Simulation of physics processes

Simulation of detector responses

Physics analysis



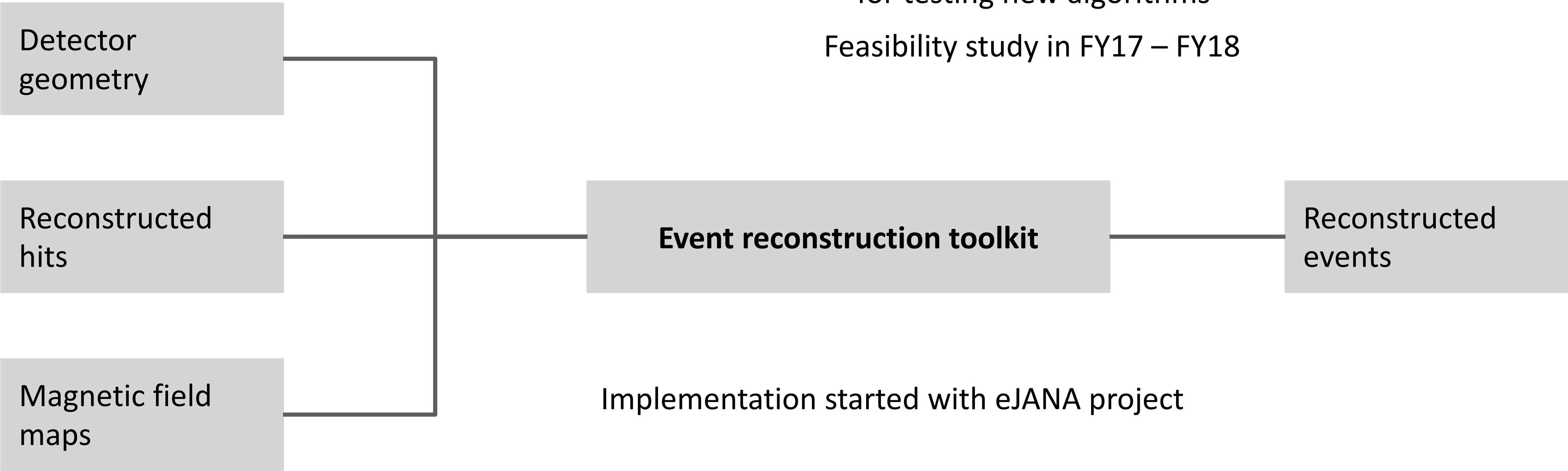
Community reference reconstruction

Charge *"The EICUG Software Working Group's initial focus will be on simulations of physics processes and detector response to enable **quantitative assessment of measurement capabilities and their physics impact**. This will be pursued in a manner that is **accessible**, **consistent**, and **reproducible** to the EICUG as a whole.*

Modular reconstruction based on EIC tracking tools (ANL, BNL, JLAB)

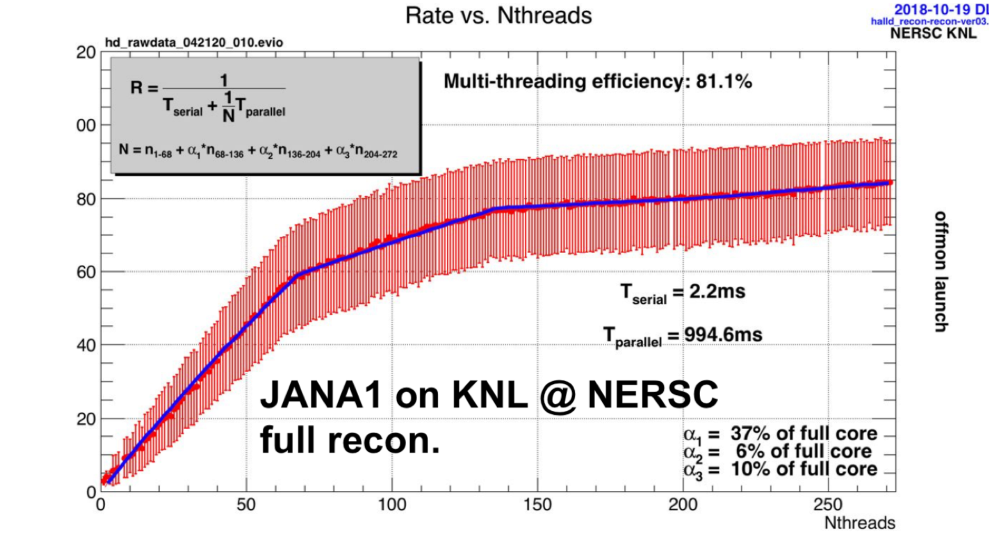
- for detector concepts and testing new algorithms (e.g., (D)NN for track finding)
- for comparing / validating EIC results
- for testing new algorithms

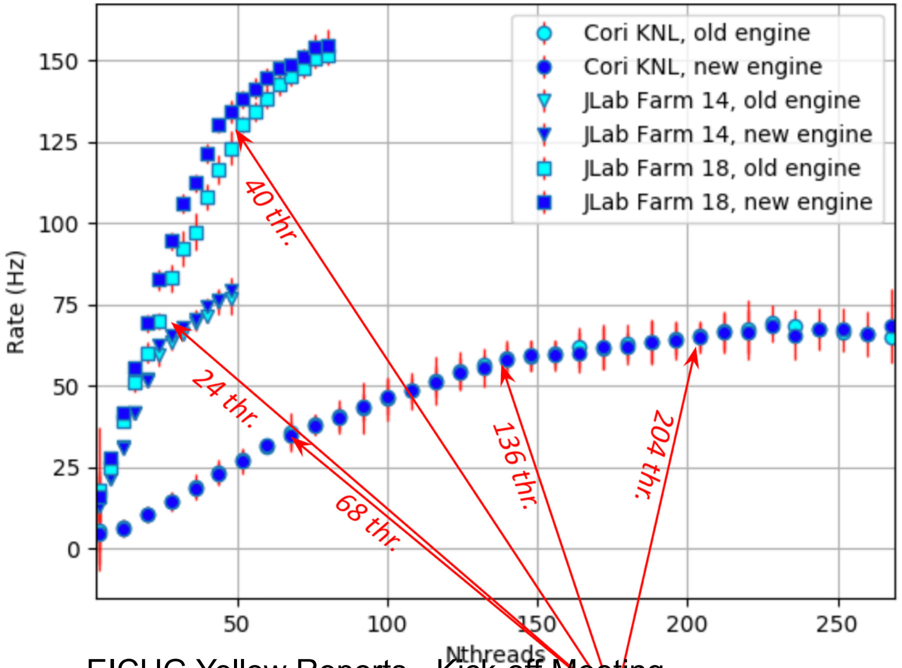
Feasibility study in FY17 – FY18



Implementation started with eJANA project

JANA2





JANA C++ event processing framework

- **factory model** on demand interface, user-centered design
- multi-threaded with > 10 years experience
- **plugin support** provide mechanism for many physicists to contribute, multi-threading external to contributed code (parallelizer)

JANA2 active development (JLAB LDRD)

- take advantage of new C++ standards
- Python interface
- part of Streaming Readout Grand Challenge at Jefferson Lab (C++ streamed events processing framework)

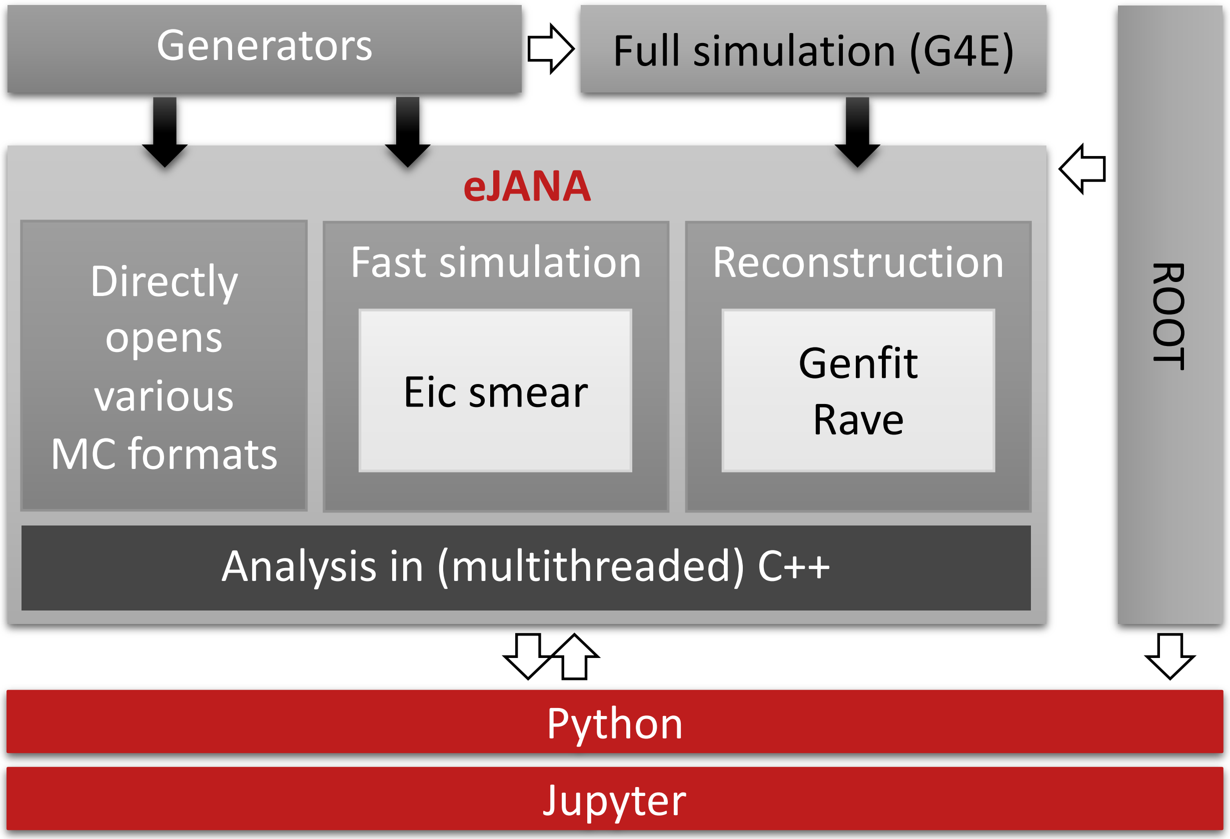
e^{JANA}: JANA2 for EIC

JANA2 + plugins

- EIC data reconstruction
- EIC data analysis

But also

- tools to manage dependencies and run eJANA in different environments
- integration with python and extensions to Jupyter Lab *(ejpm, edock, pyjano, and others..)*



Alternative reconstruction options

A Common Tracking Software (ACTS)

ATLAS software → generic, framework- and experiment-independent track reconstruction software

Collaboration of LBNL NP and HEP (Y. S. Lai et al.):

- Study of geometry import with a TPC (currently not supported in ACTS)
- momentum/vertexing of full events with ACTS

EiCRoot

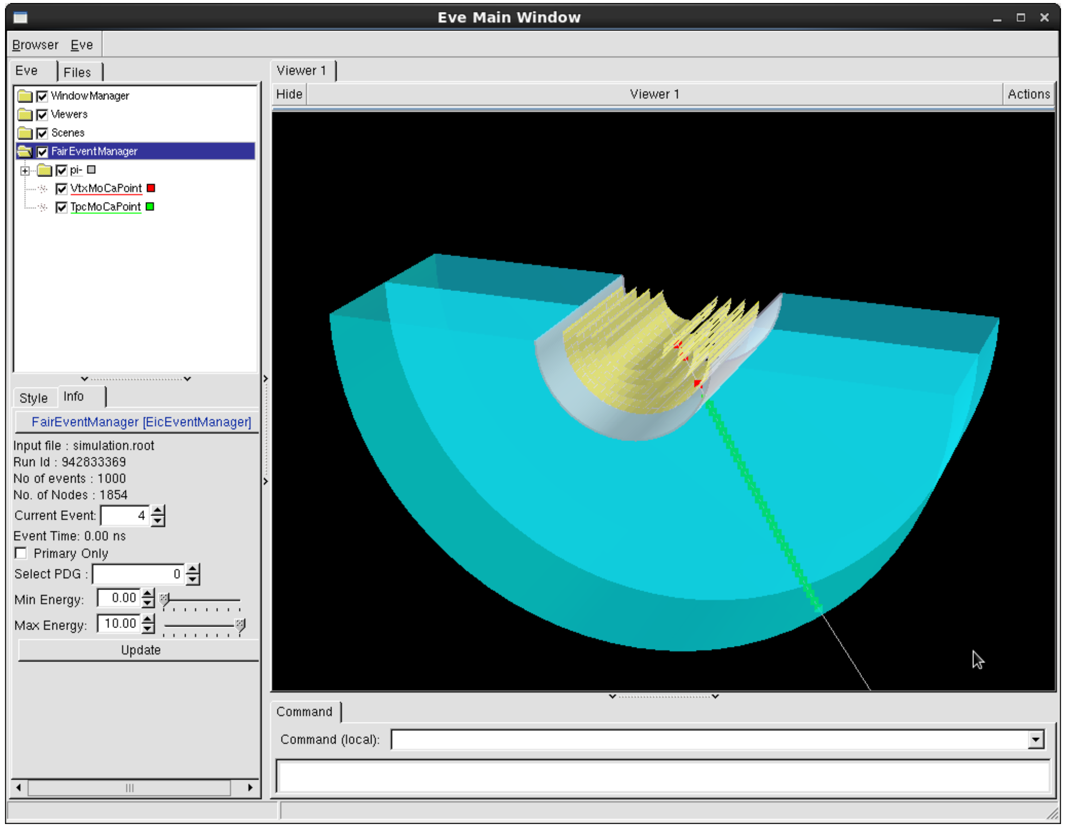
(slide 35)

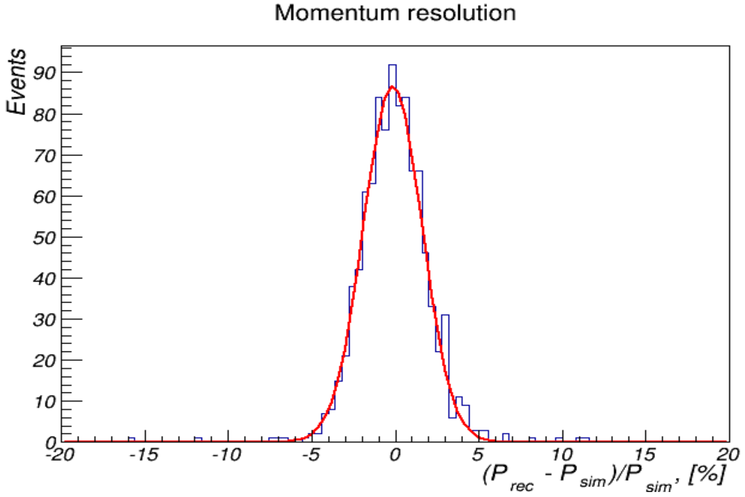
fun4all

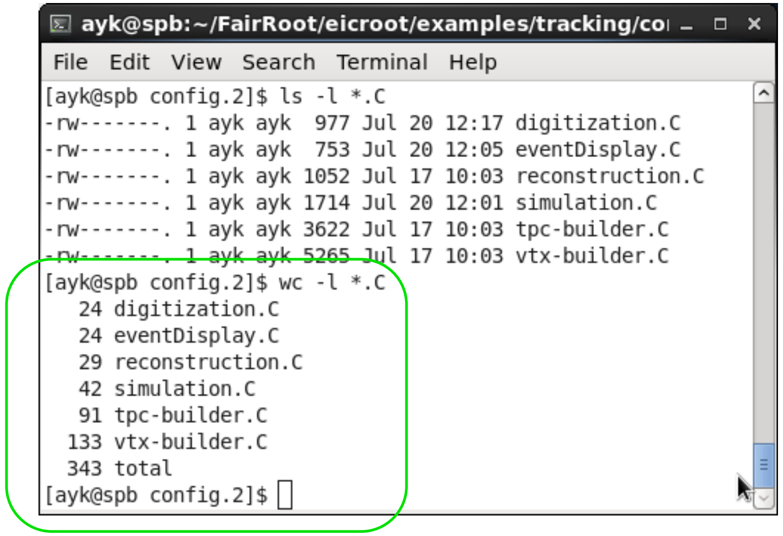
(slides 36-37)

EicRoot: Example tracking study

Consider vertex tracker + TPC in 3T field; shoot 10 GeV/c pions at θ=75°





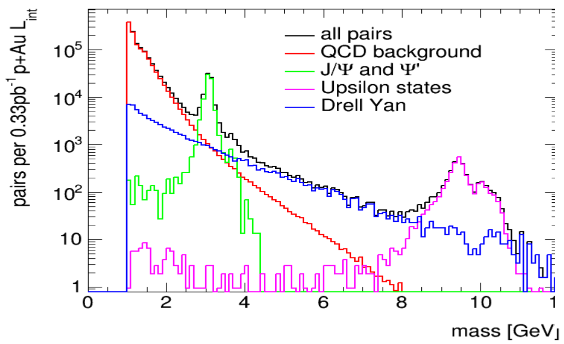


-> see examples/tracking/config.2 directory for details

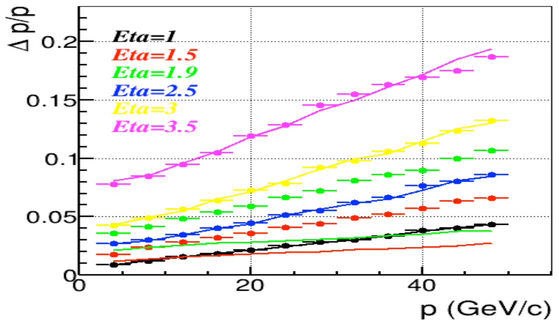
- Once Docker image is downloaded it takes <5 minutes to generate this plot

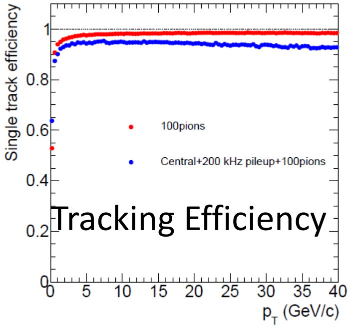
Fun4All Reconstruction

Forward Quarkonia

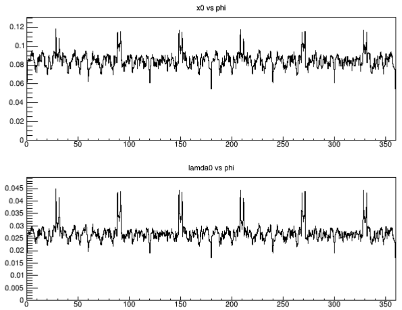


Forward Momentum Resolution



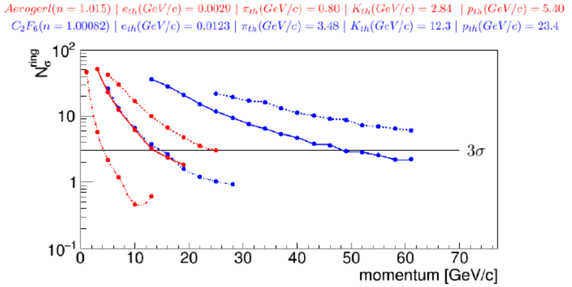


Tracking Efficiency

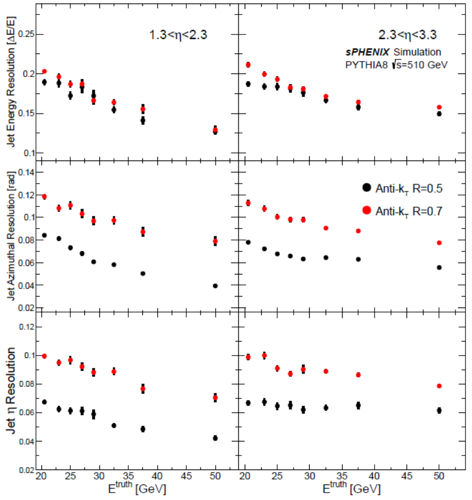


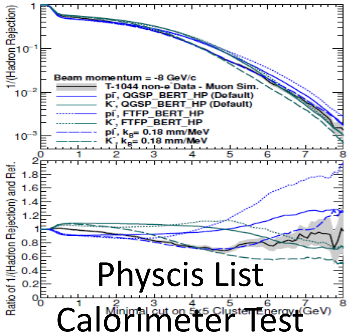
Material budget of inner tracker

Rich PID

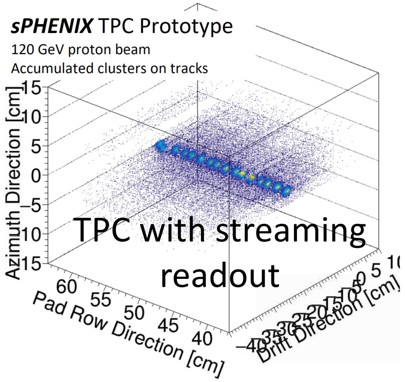


Forward Jet Energy Resolution





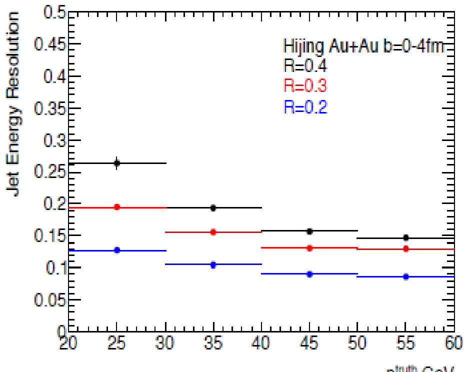
Physcis List Calorimeter Test



TPC with streaming readout

- Modularity allows easy re-use of existing components
- Many existing reconstruction modules exist (digitization, granularity, clustering, tracking, jet finding, secondary vertices…)
- Interface to raw data from rcdaq (used in test eRD beams)
- Single Chain from event generator/raw data to final user output, no switching frameworks and impedance mismatches
- Snapshots of chain can be saved, chain can pick up from there
- Large and active user base

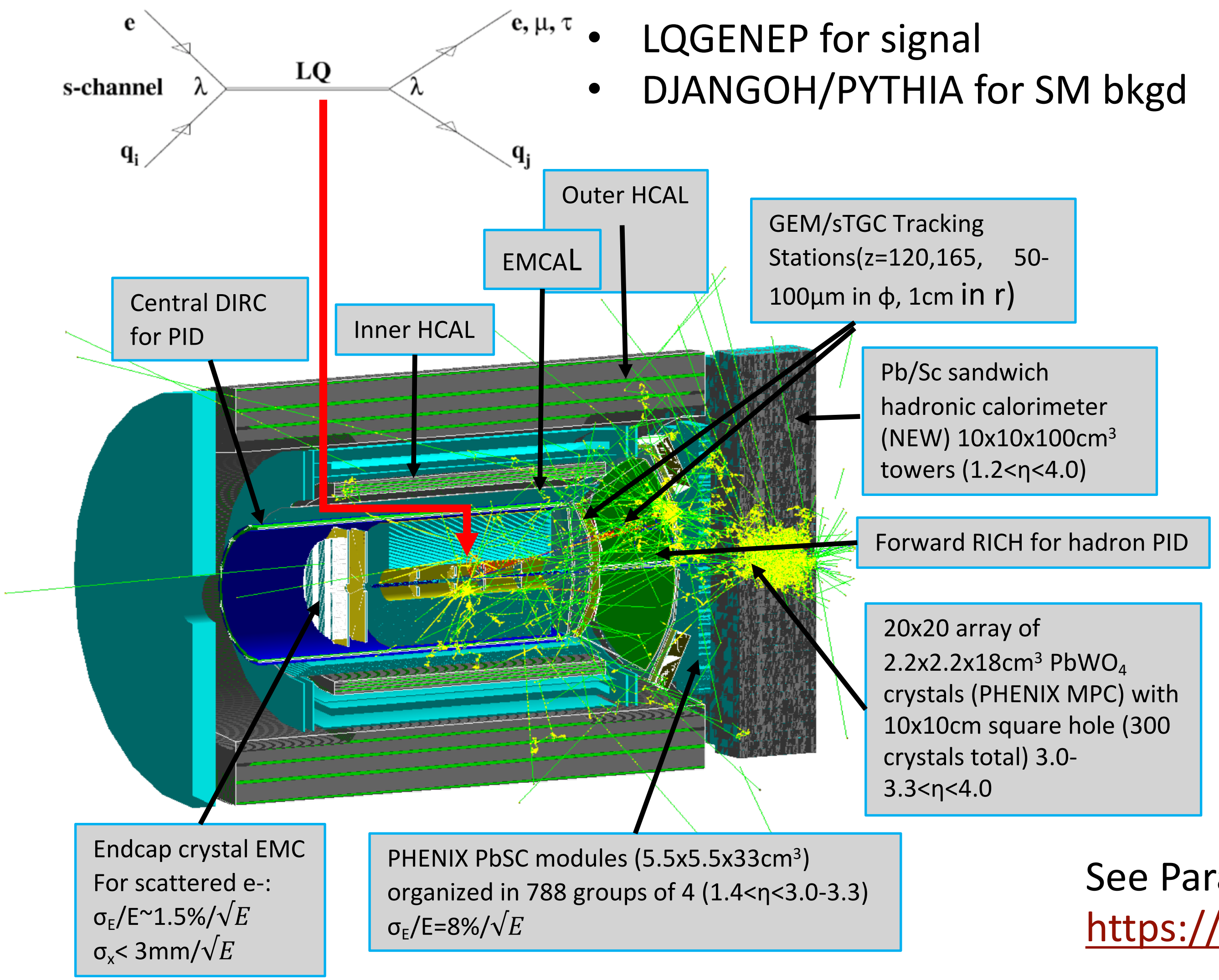
Central Barrel Jet Energy Resolution

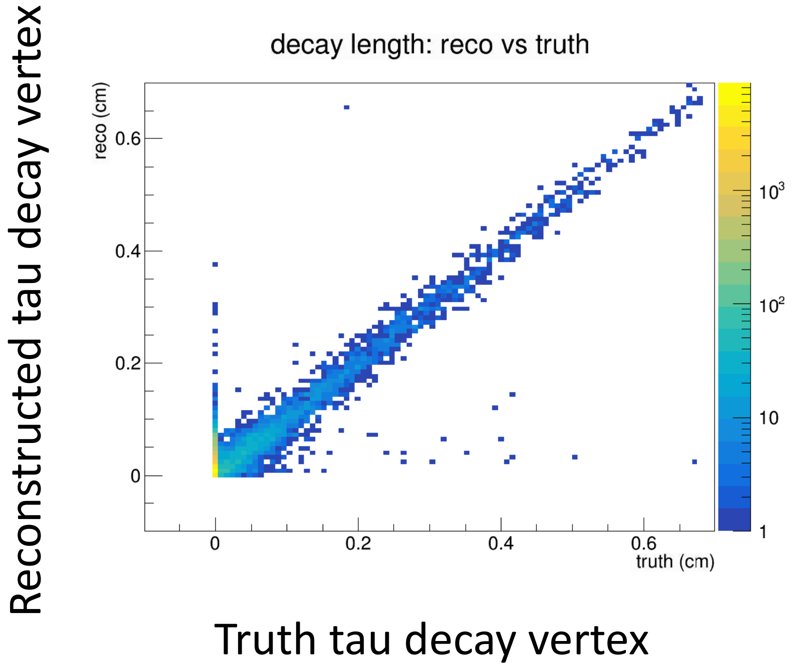


High level Example: Leptoquark Study for an EIC Detector Based on the Babar Magnet

Full detector **simulation, tracking, vertex and secondary vertex** finding using Fun4All infrastructure

- LQGENEP for signal
- DJANGOH/PYTHIA for SM bkgd





Courtesy of Jinlong Zhang, Stonybrook

See Parallel talk at EIC users meeting Paris
https://indico.in2p3.fr/event/18281/contributions/71617/

Section
Next steps

EICUG Software Working Group

43 members

Mailing list eicug-software@eicug.org
subscribe via Google Group
Repository http://gitlab.com/eic
Website https://software.eicug.org

Subscribe to Google Group for news and updates

Growing core group

Thanks for your support!


M. Asai (SLAC)


N. Brei (JLAB)


A. Bressan (Trieste)


W. Deconinck (Manitoba)


M. Diefenthaler (JLAB)


J. Furletova (JLAB)


S. Joosten (ANL)


K. Kauder (BNL)


A. Kiselev (BNL)


D. Lawrence (JLAB)


C. Pinkenburg (BNL)


M. Potekhin (BNL)


D. Romanov (JLAB)


T. Wenaus (BNL)

Batch processing

- **Open Science Grid** (OSG) for batch processing (documentation in preparation)

- osg.EIC is project to run under:
  - international users can also use this but need to let us know so we can approve their OSG accounts if they don't have one with a US-based institution or national lab (which they are encouraged to use)

- EICUG docker images are mirrored nightly on /cvmfs/singularity.opensciencegrid.org/

Andrea Bressan (INFN, University of Trieste)
Markus Diefenthaler (EIC[2], Jefferson Lab)
Torre Wenaus (Brookhaven Lab)
eicug-software@eicug.org

Workflow environment for EICUG

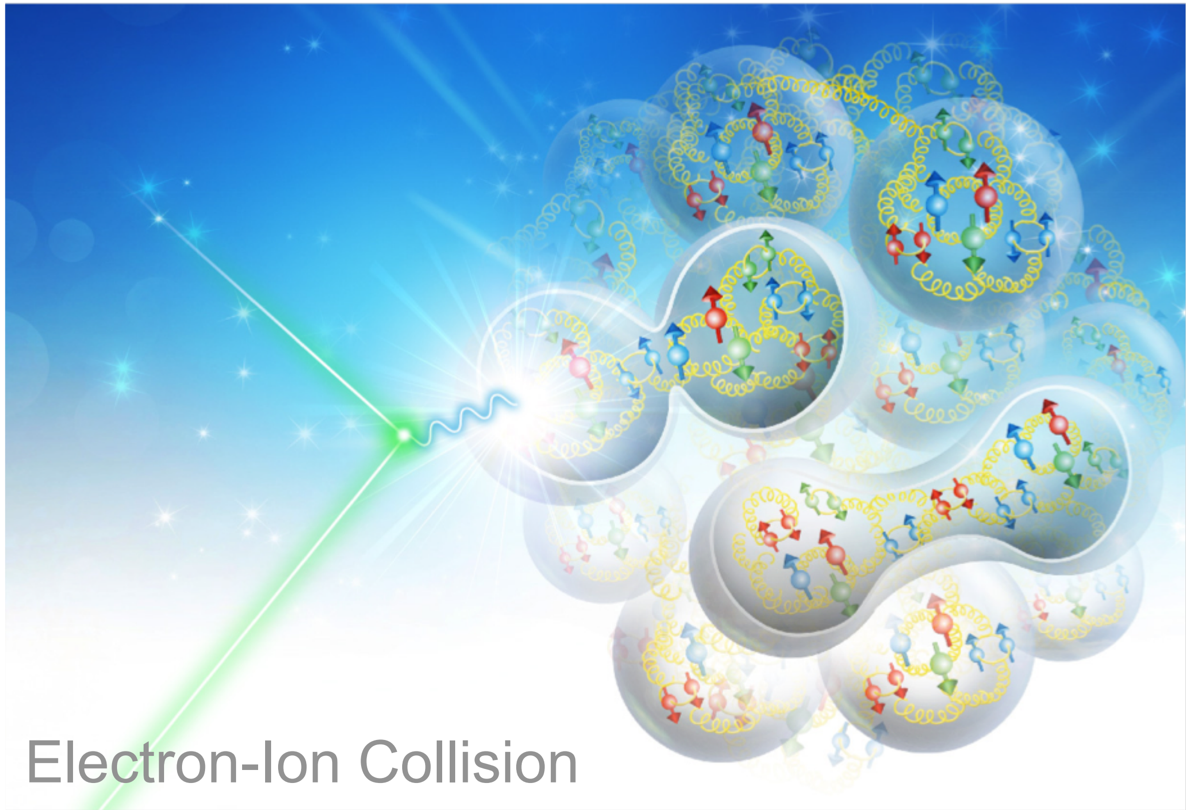
- **fast and full simulation** available and being extended with community input
- **documentation** started and being improved with community input
- **Support** available

Grow with user input

- excited to support EIC Physics and Detector Conceptual Development / Yellow Report

Path forward: Greenfield simulation software

- community-wide project on **greenfield simulation toolkit / framework** freeing us from the legacy of existing options while leveraging everyone’s experience



BROOKHAVEN NATIONAL LABORATORY

Jefferson Lab

UNIVERSITÀ DEGLI STUDI DI TRIESTE