Iguana

https://github.com/JeffersonLab/iguana

Christopher Dilks

CLAS Collaboration Meeting, November 2024

Student and Postdoc Session

Purpose

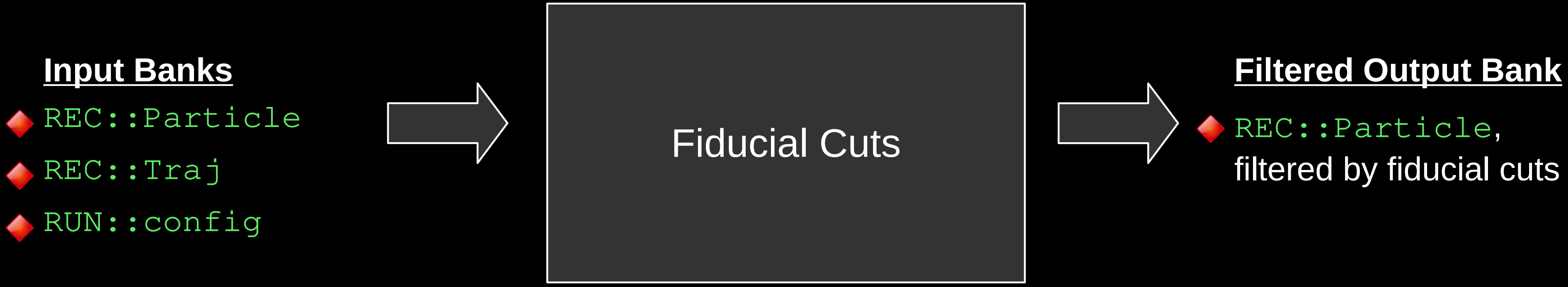
Encapsulate, centralize, and preserve common needs in **Iguana Algorithms**

- Methodology preservation (*cf*. data preservation efforts)
- Reproducibility
- Allow for focus on the important parts of an analysis
- Centralization increases the number of code reviewers
  - Lower probability of bugs
  - But if there are bugs, they impact *all* users
  - Validation is critical
- More details from the last CLAS Collaboration Meeting: https://indico.jlab.org/event/829/contributions/14072/attachments/10720/16241/iguana.pdf

What do we mean by Algorithm?

We define “Algorithm” as a function that maps a set of input banks to a set of output banks

Filter Algorithm: accepts/rejects rows of bank(s)



Input Banks

- `REC::Particle`
- `REC::Traj`
- `RUN::config`

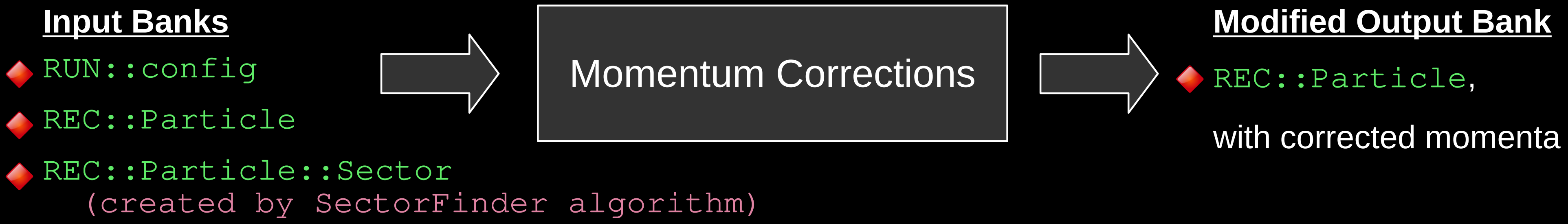
Fiducial Cuts

Filtered Output Bank

- `REC::Particle`, filtered by fiducial cuts

What do we mean by Algorithm?

Transformer Algorithm: modifies bank(s)



Input Banks

- `RUN::config`
- `REC::Particle`
- `REC::Particle::Sector`
  `(created by SectorFinder algorithm)`

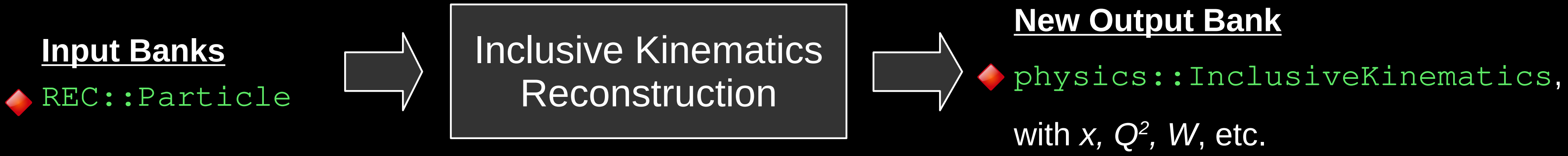
Momentum Corrections

Modified Output Bank

- `REC::Particle`,

with corrected momenta

Creator Algorithm: creates new bank(s)



Input Banks

- `REC::Particle`

Inclusive Kinematics Reconstruction

New Output Bank

- `physics::InclusiveKinematics`,

with *x*, Q^2, *W*, etc.

Available Algorithms

FTEnergyCorrection	forward tagger energy corrections
FiducialFilter	fiducial cuts (Pass 1)
MomentumCorrection	momentum and proton-E-loss corrections
PhotonGBTFilter	enhanced photon PID
SectorFinder	determine the sector for each particle
ZVertexFilter	vertex filter
InclusiveKinematics	calculate inclusive kinematics (x, Q^2, etc.)
SingleHadronKinematics	calculate SIDIS ep->ehX kinematics
DihadronKinematics	calculate SIDIS ep->ehhX kinematics

Iguana Usage Options

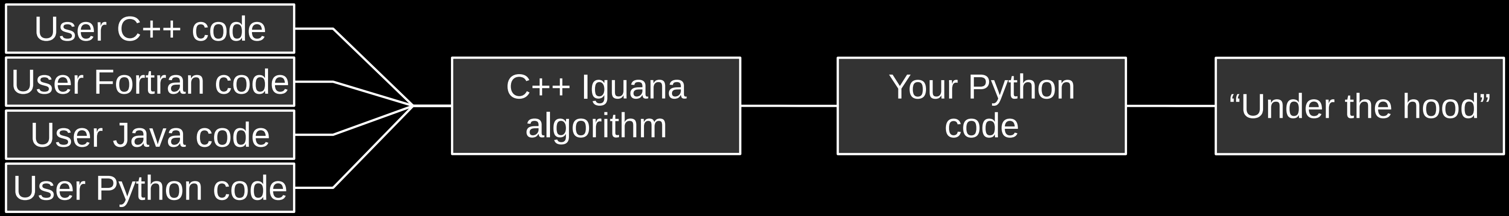
- Iguana algorithms are in C++
- Bindings are available for other languages
  - Python (via cppyy)
  - Fortran
  - Java (TODO!)
- You may also use Iguana from clas12root

Where can I find Iguana?

- **On `ifarm`**
  - `module avail iguana`
  - `module load iguana`
- **Build it yourself**
  - Follow https://github.com/JeffersonLab/iguana/blob/main/doc/setup.md
  - All dependencies are available on `ifarm`
    - A bit more work if you want to build on your personal computer
- **Use a Docker/Apptainer image from ‘clas12-containers’**
  - See my talk later this afternoon!

Can I put an algorithm in Iguana?

- Yes, please!
- If you need a new dependency, ask and we’ll try to add it
- Your algorithm *must* be in C++, so that it integrates well with other existing algorithms, tests, and language bindings
  - If your algorithm *cannot* be ported to C++, then we’ll need a simple C++ “wrapper” algorithm that would call your code and handle its output
    - For example, if you *really* need your algorithm to be in Python:



Contributions are Welcome

- We follow the usual GitHub workflow
  - Issues: planned work, bugs, feature requests, …
  - Pull Requests: new code, fixed code, …
- You may also contact the CLAS Software Group
  - Via email
  - My email: `dilks AT jlab DOT org`
  - Post in the CLAS Discourse: https://clas12.discourse.group/
- New algorithms and ideas are welcome!

https://github.com/JeffersonLab/iguana