Rivet monthly dev meeting

1 June 2022

Recent activity / TODOs

- **3.1.6 release done!** **Review MRs**
  - Lots of activity for a patch release. Thanks to Chris for driving and running the release
  - AB to follow up on remaining bits of release process — review now. Dockers are done
  - Highlighted by Frank K: our “distutils deprecation protection” apparently wasn’t enough for Ubuntu 22.04. Need a 3.17 *genuine patch* release for Py 3.10 compatibility ⇒ AB
- **YODA 2.0 developments**
  - Chris G very active, e.g.
    - honing C++ interface for “mixed axis” types,
    - “projection” schemes for e.g. Histo<N> → Histo<N-1> or Profile<N-1>
    - Discuss main points
  - UCL Sci Computing MSc student Yaru started late May on developing YODA2 Python API & designing/implementing HDF5 I/O
  - Separately, Jamie & Yoran active in migrating matplotlib plotting replacement into Rivet.
    - next step will be identifying core, non-Rivet-specific elements ⇒ move to YODA
    - YODA releases are easier than Rivet ones: can provide & refine functionality before *needed* by Rivet 3.2.0
  - **Report+discuss both…**

Misc + AOB

- **gitlab.com reducing CI time for non-premium projects: need official applications to Open Source Programme to regain full CPU quota**
  - Application done, maybe too late. Did everything stop working today?!
  - ⇒ raise issue of CERN gitlab inaccessibility via MCnet + other routes
- **MCnet event-weights standard: doc on arXiv, with SciPost, reports in**
  - Reviews reviewed by Andy & Chris, actions/responses identified… to-do
- **Events, schools, talk requests? + working meetings…**
  - Sussex reinterpretation workshop; Grenoble visitor programme; MCnet in Graz
  - Chris to investigate an autumn Rivet/Contur/reinterpretation workshop via IPPP
    - truth-level issues ("promptness" with LLPs/weak showers, EW pileup)?
  - Other routes? Expt-specific workshops? MCnet school
- **Google Summer of Code / Docs:**
  - Anjelo Narendran on new Rivet tutorials and website
  - Kalp Shah on event-file manipulation and event filtering

BACKUP

Major-release tasks

- **In parallel: work toward v3.2.0 — baseline without YODA2**
    - ~~CPU-saving no-copy of the HepMC event, with API constness change: merged~~
    - ~~Finish and merge thread-safety branch (important for Gambit → Tomek Procter)~~
    - ~~Add early versions of automatic “object flattening”, and no-width scaling~~
    - ~~Merge HDF5 analysis data, and live/dead-conversion branches → AB~~
    - Plotting merge (+ CHC patches) → Jamie, Yoran, AB, CB, etc.: **meeting needed**
    - Jet clustering of any ParticleBase: some reclustering devel, nuanced due to need to propagate constituents / recluster, maybe needs a proj subclass → **AB**
    - Deprecation clean-out and enum rationalisation (started)
        - including “enumification” of the DISKinematics options arg :-/ → AB
    - Primary particles definition / enforcement
      → mix of PIDs and decay time; Leif started tech discussion

Path towards YODA2

- **Plan for major version release around summer!**
- ~~Finalise translation of the usual YODA 1.9 objects into YODA2-style objects in time for Easter~~
  - ~~Support all usual histogram/profile/scatter object types + new (continuously) binned Estimates~~
- Spend some time after Easter on (more validation and) syntactic sugaring of discretely binned axes
  - If this cannot be incorporated with reasonable turn-around (e.g. too complicated or other distractions get in the way), propose to postpone user-friendly support of discrete binning to a later YODA 2.1 release (autumn/winter?)
    My current feeling is this won't be necessary, though … 🤞
- Outstanding ToDos:
  - ~~Finalise Estimate implementation~~
  - Syntactic sugaring for ~~BinnedStorage<Estimate, BinnedAxis> (+~~ discretely binned axes~~)~~
  - Reduce operations (e.g. for live-to-dead conversion)
  - Update Python API (😬)
  - Need new I/O reader and writer
  - Update docs with practical examples
  - Update build tests, tweak CI if necessary + validate, validate, validate, …

Big picture tasks (near duplicate from April & May)

- **Stats objects are our major technical bottleneck**
  - Integrate and extend new plotting system
  - YODA type-extension (build on Nick R GSoC 2020 work)
  - HDF analysis data and new YODA format
  - post-finalize() always "flatten" stats objects to "binned measurement" type
  - [finish multiweight-fill optimisation (Aditya GSoC 2020)]
- **Scaling**
  - Analysis distribution system... again
  - Ref-data and analysis data particularly problematic: decouple data from code??
  - HD consistency
- **Standardising:**
  - MCnet weight-name/structure proposal: productive meeting on May 21, lots of agreement, AB to update and recirculate proposal
  - [Event-record content: excessive size and physicality…]

Major (stats) work plans

- **“Flattening”: convert finalize output to inert objects (scatters/binnedmeas)**
  - Final objects really will mean “what was plotted/listed in the paper”
  - Allow eager conversion to solve “no-bin-width issue”
  - Best that we wait for binned measurement YODA2 types: no more scatters!

- **HDF5 analysis data machinery (Holger)** *Status?*
  - Also interested in HepMC and YODA HDF5 formats
  - Holger to ping CMS, prototype interface

- **Plotting (Christian B et al)**
  - Plan: generate Python MPL scripts *without* TeX, .plot styles → YAML
  - Rivet labels tested: MathText fails due to missing std symbols. Can we extend?
  - Stalled for a while… restarting? Possible student help from David Grellscheid
  - Christian to prototype the Python-script generation
  - Chris to extract weight-handling logic from rivet-cmphistos

Performance in Rivet and YODA (Aditya Kumar, AB)

- **Profiling revealed bottlenecks: thanks Aditya!**
  - HepMC ASCII I/O (obviously) — taken out of tests by event-reuse
  - GenEvent copying — for sanitising, but hardly used: removed from Rivet.
    Could/should generators write smaller "essential" events by default?
    Awkwardness: we still normalise GenEvent units… so not quite analysing a const GenEvent.
    But can't justify an expensive copy for *unit conversion*…
  - PID functions — sped up charge lookups by special-cases. Marginal gain
  - Multiweight calls to histo fill() *very* expensive: ~40-50% CPU!
    100+ consecutive fills with same *x*: tried caching in YODA but no benefit:
    cache-check costs the same as linear bin lookup! *Maybe cache in Rivet?*

- **Thread-safety.** *"Just store a ProjectionHandler in AnalysisHandler: easy!"...?*
  - But then who do Projection constructors (recursively) register their contained projections with, before they themselves have been bound to a PH?
  - "Declare queue" implemented: not yet working (thx, unique_ptr), but should do
    What should *the Projection ownership be?!*

YODA generalised datatypes (Nick Rozinsky, LC, AB)

- **Long-understood limitations of YODA types and design**
  - Overreach in attempted non-factorisable binnings: composed 1D axes are fine
  - Complexity/mess in 2D overflows: need “infinity binning”
  - Need for binned “dead” data objects… or any type, actually
  - Want programmatic access to axis number and global/local bin indexing
  - Want labelled/discrete binnings as well as continuous
  - Code duplication, particularly in Cython interface building

- **Major YODA redesign using modern C++ magic. Thanks Nick!**
  - E.g. Histo1D → wrapper of a BinnedStorage<CAxis, Dbn<1>> + sugar
  - + arbitrary mixtures, e.g. 3D binnings of doubles, discretely labelled counters, …
  - Adaptors used to map fill/set behaviours. Can do the same for I/O read/write?

- **Path to a YODA2 release:**
  - Needs I/O adaptors and user-facing refinements. Tie in with HDF5 format?