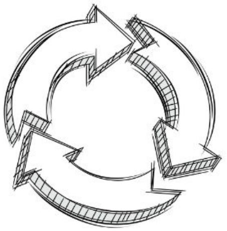
Rivet (for jet substructure)

Andy Buckley
University of Glasgow

BOOST 2014, London, 20 August 2014

Introduction

- **Big developments in how the (LHC) experimental and theory communities interact in last few years**
- Especially around QCD, lots of direct collaboration to improve methods and modelling. Like BOOST itself…
- **Rivet** analysis library is part of that: a lightweight way for exchanging analysis details and ideas
- Implementing a Rivet analysis to complement the data analysis is increasingly expected of CMS and ATLAS analyses. **Everyone benefits!**
- **This talk: description/discussion + demo** More about the philosophy and recent/relevant developments than detailed technicalities (we have a manual and a mailing list for that)



Introduction

Rivet is an analysis system for MC events, and *lots* of analyses

289 built-in, at today's count! 38 are pure MC, 41 unvalidated

- *Generator-agnostic* for physics and practical reasons
- A quick, easy and powerful way to get physics plots from lots of MC gens
    - Only requirement is that gens write **HepMC** objects/format
    - (ASCII format is usually used, but direct in-memory access is faster)
- Rivet has become the LHC standard for archiving LHC data analyses
    - Focus on *unfolded* measurements, esp. QCD, rather than searches
    - But there are BSM studies using it!
    - Key input to MC validation and tuning – increasingly comprehensive coverage
    - **Add your analyses, too!**

Design philosophy / pragmatics

Rivet operates on HepMC events, intentionally unaware of who made them…so don't "look inside" the event graph.

$\Rightarrow$ reconstruct resonances, dress leptons, avoid partons, etc.

cf. q/g jet discrimination: LO picture is an implementation-dependent cartoon; a useful motivator but incomplete and ill-defined until after hadronization

This "hard work" way is actually simpler – fewer gotchas.
Makes you think about physics & helps find analysis bugs/ambiguities

Tech stuff:

- C++ library with Python interface & scripts
- "Plugins" $\Rightarrow$ write your analyses without needing to rebuild Rivet
  Trivial from user / analysis author point of view
- Tools to make "doing things properly" easy and default
- Computation caching for efficiency
- Histogram syncing: *keep code clean and clear*

+ helpful developers! New contributors always welcome

Running Rivet

Easy to install using our *bootstrap script*:

```
wget http://rivet.hepforge.org/hg/bootstrap/raw-file/2.1.2/rivet-bootstrap
chmod +x rivet-bootstrap
./rivet-bootstrap
```

Latest version is 2.1.2; but I'll demonstrate today with 2.2.0beta1

- **`rivet`** command line tool to query available analyses
- Can be used as a library (e.g. in big experiment software frameworks)
- Can also be used from the command line to read HepMC ASCII files/pipes: very convenient
- Helper scripts like **`rivet-mkanalysis`**, **`rivet-buildplugin`**
- Histogram comparisons, plot web albums, etc. very easy

Example run on jet shape analyses

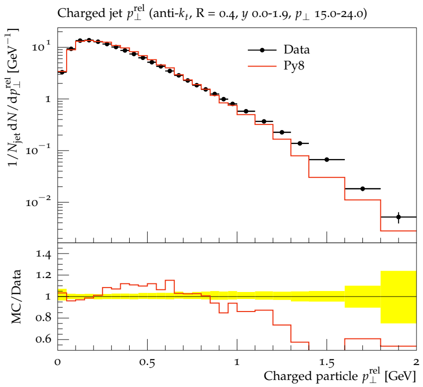
```
$ mkfifo fifo.hepmc # NOT ON AFS! Use /tmp/$USER on lxplus

$ run-pythia -e 7000 -c HardQCD:all=on -c PhaseSpace:pThatMin=15
  -c PhaseSpace:bias2selection=on -c ParticleDecays:limitTau0=on
  -n 10000 -o fifo.hepmc &

$ rivet -a ATLAS_2011_S8924791   # classic jet shapes
  -a ATLAS_2011_I919017   # track jet props
  -a ATLAS_2012_I1094564   # mass and substr
  -a ATLAS_2012_I1119557   # high pt shapes
  -a CMS_2013_I1224539_DIJET   # jet masses
  fifo.hepmc

$ rivet-mkhtml -a Rivet.yoda:'Py8'
```

$$\Longrightarrow$$

Example output

```
# BEGIN YODA_HISTO1D /ATLAS_2011_I919017/d01-x01-y01
Path=/ATLAS_2011_I919017/d01-x01-y01
ScaledBy=1.123803092733942
Title=
Type=Histo1D
XLabel=
YLabel=
# Mean: 1.739146e+00
# Area: 6.863568e+03
# ID     ID      sumw    sumw2   sumwx   sumwx2  numEntries
Total           Total           6.863568e+03    7.381017e+02    1.193675e+04    7.104475e+04    364085
Underflow       Underflow       6.284459e+03    6.798003e+02    7.000148e+03    1.201060e+04    312937
Overflow        Overflow        2.565131e-02    1.308516e-06    3.226243e+00    4.322514e+02    4249
# xlow   xhigh   sumw    sumw2   sumwx   sumwx2  numEntries
4.000000e+00    5.000000e+00    1.518135e+02    1.686008e+01    6.752514e+02    3.016264e+03    8144
5.000000e+00    6.000000e+00    9.066276e+01    9.612548e+00    4.966447e+02    2.728790e+03    5233
6.000000e+00    7.000000e+00    6.658654e+01    7.348775e+00    4.321264e+02    2.809855e+03    3522
7.000000e+00    8.000000e+00    4.687470e+01    4.987798e+00    3.499815e+02    2.617007e+03    2548
8.000000e+00    9.000000e+00    3.889044e+01    4.144272e+00    3.298475e+02    2.800817e+03    2111
9.000000e+00    1.000000e+01    3.183864e+01    3.094788e+00    3.017020e+02    2.861477e+03    1742
1.000000e+01    1.100000e+01    2.550131e+01    2.594758e+00    2.678468e+02    2.815272e+03    1461
                                                .
                                                .
# END YODA_HISTO1D
```

Example output



Back to philosophy

The important stuff:

- **Lots of data:** it's only through "global" observable coverage that we can test both data and models. Phase spaces rarely line up exactly between measurements, and you don't know that a model is good/bad until you've really tried to *tune* it to lots of data.
- **Unfolding:** experiments are the only people who can be expected to understand the detector & reco details. Effects may be small – maybe – but theorists can't be sure. Unfolding is hard, but crucial. *"Unfold detector effects, but don't extrapolate acceptance or unfold to partons"!!*
- **Authorship:** 1000% better when original analysts write the analysis routine. You wouldn't believe how hard it is to write a paper that's as watertight as a short piece of code.

PS. Also make sure to send your data to **HepData**. Makes it *much* easier to get into Rivet (and other pheno tools)

Thinking about physics object definitions

Rivet has also proven to be a useful theatre for developing theoretically and practically better-behaved truth object definitions.

Examples: **Charged lepton dressing:**

- Long debate about correct handling of MC electrons (and muons).
- QED FSR means that stable lepton MC particles have less energy than at “Born” level. Representation of QED FSR implementation-specific: PHOTOS vs. Herwig++ vs. Pythia…
- Calorimeter will see the sum of electron + collinear photons within roughly a calo cell radius $\Rightarrow$ cluster all photons within $\Delta R \sim 0.1$ of bare ch lepton?
- Iterated to a “negative definition” first implemented in Rivet: don’t want to pick up decay contributions from $\pi \to \gamma\gamma$ $\Rightarrow$ cluster photons *not* identified as from a hadron decay by a recursion up status = 2 decay chain: very close performance to MC-specific Born level

Thinking about physics object definitions

Rivet has also proven to be a useful theatre for developing theoretically and practically better-behaved truth object definitions.

Examples:

"Promptness":

- More generally, can we identify particles "from the hard process" or "from a EW boson decay"?
- Direct connection in event record *very* flaky and unportable
- Use the negative "not from hadron" recursion again
- New `PromptFinalState` projection implements this. *Will never return true for a jet or hadron!*
- What about photons? How can we tell the difference between a "prompt" photon and one from QED FSR near a lepton? In reality we couldn't tell…

Thinking about physics object definitions

Rivet has also proven to be a useful theatre for developing theoretically and practically better-behaved truth object definitions.

Examples:

Overlaps:

- Next issue to consider is the *ordering* of such definitions
- Find (hard) electrons first then remove them (and their clustered photons) before jet finding?
- Prompt photons again? flavour or q/g labelling?

More definition issues to be explored!
Important for precise studies of complex final states ($t\bar{t}$ is current testbed) and ones where asymptotic ideas like jet=parton break down (e.g. fat jets, q/g)

Current substructure analyses

Current unfolded substructure-relevant analyses in Rivet:

ATLAS_2011_S8924791	jet event shapes	link
ATLAS_2011_I919017	track jet properties	link
ATLAS_2013_I1243871	$t\bar{t}$ jet shapes	link
ATLAS_2012_I1094564	mass and subst	link
ATLAS_2012_I1119557	shapes and masses	link
CMS_2013_I1224539_DIJET/WJET/ZJET	jet masses	link
…	*MORE?*	

"FSR" performance generally good – if you can hit jet shapes, much of the rest also falls in line. Departures on the 20% level common, but data also statistically limited in so-far available analyses.

Need a systematic generator comparison – include matching/merging – on all these plots… again.

`mcplots` was the hope for this, but seems to have died – human effort in configuring/running gens is typically the bottleneck in e.g. BOOST report studies.

Recent (jet handling) improvements

Rivet 2.2.0 improvements for jets / substructure

Cuts:

Limition of e.g. `jetalg.jetsByPt(1, -2, 2)` – how to express disjoint cut ranges? Or η rather than y?

If every arg is a double, how to distinguish / remember ordering?

New combinable `Cut` objects:

⇒ `FinalState(ptGtr(0.5*GeV & etaIn(-2.5, 2.5)))`

⇒ `fs.particles(rapIn(-5, -3.2) | rapIn(-3, 3) | rapIn(3.2, 5), sortByEta)`

Recent (jet handling) improvements

Rivet 2.2.0 improvements for jets / substructure

FastJet interaction:

Used to have to "drop out of Rivet" to do filtering, substructure, etc.

We see no point to "wrap" every FastJet feature – but can help to make the interoperation smooth.

Rivet's `Jet` and `Particle` classes now auto-convert to `PseudoJet`:
⇒ `d23 = cs.exclusive_subdmerge(jetproj.jetsByPt[0], 2)`

More to come, e.g. `JetAlg::pseudojets()`, `ParticleFinder::pseudojets(Cuts, Sorter)`, etc.

Recent (jet handling) improvements

Rivet 2.2.0 improvements for jets / substructure

Jet tagging:

Previously used a very inclusive tagging definition based on hadron parentage:

```
j.hasBottom()
```

Still an option, but now also automatically ghost-tag jets using *b* and *c* hadrons:

```
if (!myjet.bTags().empty()) ...
```

Could also add ways to tag with *anything* specified by user, if that would be useful…

Writing analyses: W' finder and JH top tagger

I'll demonstrate quickly how to do a (very generic) JH boosted top tagger analysis, and something like ATLAS' recent boosted W' search analysis.

Note for Nsubjettiness – need to hack the FJContrib Makefile to add **`-fPIC`** to **`CXXFFLAGS`**. Can we get that fixed?

Summary

- **Rivet helps to make data analyses permanently useful**
  - Designed to encourage *physical*, *portable* and *clear* analyses
  - Nice, quick testbed for analysis / pheno ideas!
  - Comprehensive, detector-unfolded, observable coverage important for modelling improvements
  - Actively developed and supported. Open to requests / contributions
- **A number of substructure-relevant analyses already implemented**
  - These ones are (all?) unfolded.
  - Super-high stats on jet shapes – much worse on masses and more exotic variables
  - A public archive of "contender" MC models convering shapes & structure (and some intrajet, too) would be valuable – a successor to mcplots?
- **Rivet 2.2.0 to be released soon**
  - More analyses, etc. as usual
  - New features for tagging and FastJet interoperability
  - Multi-weights, NLO counterevents, etc. on the (too long!) TODO list

Backup

Writing a Rivet analysis

An example is usually the best instruction: take a look at the MC_GENERIC analysis via **http://rivet.hepforge.org/hg/rivet/file/tip/src/Analyses/MC_GENERIC.cc**)

Mostly "normal":

- Typical init/exec/fin structure
- Histogram booking normal here, but no titles, labels, etc.: use `.plot` file
- Rivet's own Particle, Jet and FourMomentum classes, but no (bad) surprises – actually, nice things like `abseta()`
- Use of *projections* for computations, with a bit of magic – this is where the caching happens
- Projections are registered with a string name, and later are executed using the same name
- Final state projections are central: compute from final state or physical decayed particles

Rivet histogramming and plotting

ROOT didn't meet our needs/aspirations :-(

bin widths and bin gaps unhandled, object ownership nightmare, thread-unsafety

Rivet 2 uses a new system called YODA – **http://yoda.hepforge.org**

Designed to be nice, correct and powerful – and I think we got this one right! Currently supports 1D and 2D histos and profiles, 2D and 3D "scatters". Unbinned objects and better overflows in the pipeline... and "general" 2D binning!

Main YODA data is a plain text format which stores all second-order statistical moments: can do full stat merging, including details like weighted focus inside bins. And general annotation system for metadata – styling, notes, whatever

Command line tools: `yodamerge`, `yoda2aida`, `yoda2root`, ... and `yodahist`

Plotting still via Rivet's `make-plots` script: nice output via LaTeX but slow and can be awkward to customize. Replacement slowly evolving! What you want to know:

```
rivet-mkhtml Rivet.yoda:'Pythia\,8 $t\bar{t}$'
```
!!

Using FSPs to get final state particles

```
void analyze(const Event& evt) {
  ...
  const FinalState& cfs =
    applyProjection<FinalState>(evt, "ChFS");
  MSG_INFO("Total charged mult. = " << cfs.size());
  foreach (const Particle& p, cfs.particles()) {
    const double eta = p.momentum().eta();
    MSG_DEBUG("Particle eta = " << eta);
  }
  ...
}
```

More complex projections like **`DressedLeptons`**, **`FastJets`**, **`WFinder`** implement expt-like strategies for dressing, tagging, mass windowing, etc.

We need more of those, e.g. a **`TauFinder`**, . . .

Recent developments

- More analyses, obviously: see **http://rivet.hepforge.org/anadiffs** 3 new CMS ones in last release, another 4 in current beta.
- Rivet contrib: **https://www.hepforge.org/archive/rivet/contrib/**
- Enhancements to ParticleBase, Jet, Particle, FourMomentum: heading toward interchangeability, now little need for `myparticle.momentum()`
- Also better utils like `sortByPt(CONTAINER<ParticleBase>&)`, `any(CONT, FN)`, `all(CONT, FN)`
- Handy `abspid()`, `abseta()`, `absrap()` functions: every little helps
- Math utils: `sqr(x)`, `inRange(x,l,h)`, `add_quad(x,y)`, `linspace`,`logspace`,`bwspace()`, `binIndex()` etc.
- 2D histograms re-enabled – YODA developments ongoing, esp. fully overflows

Plans

Generally:

- ATTACK THE UNVALIDATEDS! ATLAS is main culprit…
- ~~Tau finder~~, ~~prompt finder~~, ~~general particle finder interface~~, pseudo-top finder, ~~more decay chain tools~~

Version 2.2:

- `Cuts` system: replace army of `(double, double, int, enum, ...)` function variations with single combineable kinematic cut object
- ~~Better FourMom/Particle/Jet interaction with FastJet~~
- ~~Built-in jet tagging (b/c-hadron, tau)~~

Version 2.3:

- Multi-weight events and histogramming of syst bands. And NLO counter-events, etc.
- "Factorized `finalize`" – completely arbitrary data object stat merging! Maybe make the system usable outside Rivet…
- New plotting…no-one pays for this, so it's a private indulgence. But we have a nice prototype that combines Cairo with TeX: fast and high quality = WIN!