EXPERIMENT INFRASTRUCTURE FOR SYSTEMATIC VARIATIONS

Enrico Guiraud

Analysis Ecosystem Workshop, 23/05/2022

WHAT

A look at current experiment software infrastructure for handling common sources of systematics (e.g. from reconstruction).

A *partial* summary of feedback collected at shorturl.at/uM237 (*thank you* to all responders).

WHY

As fodder for discussion.

I will try to highlight commonalities and opportunities.

Apologies for any bias towards CMS and ATLAS: mostly due to more standardized workflows.

RELEVANCE OF THIS CONTENT

Many use cases are not covered by the centralized code discussed in the survey.

However the patterns discussed could be replicated by specific analyses, *if* the centralized code used facilities that are also nice to use at the level of end-users.

DISCLAIMER

I might have gotten some things wrong.

But hey, *that* will certainly spark a discussion!

TRENDS & CHALLENGES

SYSTEMATICS AS CODE AND AS DATA

With some exceptions (e.g. LHCb) common systematics are handled by a combination of centralized code and centralized production of reduced data formats (PHYSLITE, NanoAOD).

THE SYSTEMATICS PIPELINE

- **full data**: experiment-wide, reconstructed objects
- **reduced formats**: experiment-wide, e.g. CMS NanoAOD, ATLAS PHYSLITE, targeting common analysis use cases
- **analysis-specific data**: produced by end-user code (flat ntuples, event weights) as an easy-to-handle intermediate format, might include systematics
- **on-the-fly systematics**: esp. to compute weights, but not only

THE SYSTEMATICS PIPELINE

- **full data**: experiment-wide, reconstructed objects
- **reduced formats**: experiment-wide, e.g. CMS NanoAOD, ATLAS PHYSLITE, targeting common analysis use cases
- **analysis-specific data**: produced by end-user code (flat ntuples, event weights) as an easy-to-handle intermediate format, might include systematics
- **on-the-fly systematics**: esp. to compute weights, but not only

Can we remove step 3?

PROBLEMATIC LOSS OF PRECISION OR MISSING INFORMATION IN REDUCED FORMATS

Tension between keeping files small and keeping information useful for analysis (e.g. for evaluation of systematics).

- **CMS**: typically not a problem for NanoAODs
- **ATLAS**: unclear, currently being debated for PHYSLITE
- **LHCb, Belle II**: reduced formats are not a thing

AHEAD-OF-TIME OR ON-THE-FLY?

Currently we have a mix of ahead-of-time computations of variations, then stored in auxiliary ntuples, and tools that let analyses compute variations “on the fly”, i.e. as part of the main analysis.

General interest in seeing more done on-the-fly in order to save storage, reduce I/O.

At the same time, no general prescriptions/patterns for analysis-specific systematics.

C++ OR PYTHON?

- **C++**: ATLAS' CP tools
- **C++ with Python bindings**: CMS' correctionlib
- **Python**: PID calibration in Belle II, LHCb

As usual, Python picked for end-user ergonomics, C++ for performance and to write event-at-a-time logic. We can have both!

VARIED HISTOGRAMS

General interest in a user-friendly representation of a “varied histogram”, e.g. boost-hist with categorical axis.

- it could provide common useful visualizations
- it could be a standardized way to communicate information to statistics packages
- a chance to pool efforts from different experiments? interaction with future RHist?
- storage/retrieval to/from ROOT files?

EVENT LOOP OR COLUMNAR EXPRESSIONS?

Tension between writing tools for event-by-event or column-at-a-time paradigms.

CMS’ correctionlib (C++ with Python bindings) shows that we don’t necessarily have to choose.

OBJECT SYSTEMATICS, COLUMNAR STYLE

Several responders in doubt on how to express object-/event-level systematics in a columnar style.

Solution: (re)definition of quantities with smart tracking of dependencies between derived quantities and systematics, á la RDataFrame?

EOF