Status of Rivet and Professor MC validation & tuning tools

Andy Buckley, Jon Butterworth, Hendrik Hoeth, Heiko Lacker, James Monk, Holger Schulz, Jan Eike von Seggern, Frank Siegert

The Rivet [1] package for MC generator validation and the Professor [2] system for generator tuning have become established tools for systematically verifying event simulations and optimising their parameters, where required and physically sensible. In this short report, we summarise the status and development of these tools.

1. Rivet

Rivet is an MC *validation* tool: it encodes MC equivalents of an increasingly comprehensive set of HEP collider analyses which are useful for testing the physics of MC generators. Rivet does not itself produce tunings, but provides a standard set of analyses by which to verify the accuracy of a given generator with a given tuning.

Several fundamental design principles have been derived from the experience on Rivet's predecessor system, HZTool [1], and from iteration of the Rivet design:

- No generator steering: Rivet relies entirely on being provided, by unspecified means, with events represented by the HepMC [3] event record.
- No generator-specific analyses: all Rivet analyses are specifically not allowed to use the generator-specific portions of the supplied event records. Apart from a few limited (and deprecated) exceptions, all analyses are based solely on physical observables, i.e. those constructed from stable particles (those with status 1) and physical decayed particles (those with status 2).
- Rivet can be used either as a C++ library to be interfaced with generator author or experiment analysis frameworks, or as a command line tool (which itself makes use of the library interface). This is an example of the general philosophy to keep things simple and flexible, since we do not *a priori* know every task to which our tools will be employed.

Internally, Rivet analyses are based on a comprehensive set of calculational tools called *projections*, which perform standard computations such as jet algorithms (using FastJet [4]), event shape tensors, and a variety of other standard tasks. Use of projections makes analysis code much simpler, encapsulates any complexities arising from the ban on use of event record internal entities (the summation of photon momenta around charged leptons during Z-finding is a good example), and is more efficient than just using library functions, due to a complex (but hidden) system of automatic result caching.

Users can write their own analyses using the Rivet components and use them via the Rivet API or command-line tool without re-compiling Rivet, due to use of an analysis "plugin" system. Separation between generator and Rivet on the command-line is most simply achieved by using the HepMC plain text `IO_GenEvent` format via a UNIX pipe (a.k.a. FIFO): this avoids disk access and writing of large files, and the CPU penalty in converting event objects to and from a text stream is in many cases outweighed by the general-purpose convenience. For generator-specific use of Rivet, the programmatic interface allows HepMC objects to be passed directly in code, without this computational detour. A sister tool, AGILe [5], is provided for convenience control of several Fortran-based generators, with command-line and parameter file based run-time steering of generator parameters. This is a convenient tool when exploring generator parameter space as part of a tuning.

Reference data for the standard analyses is included in the Rivet package as a set of XML files in the AIDA format. After several years of re-development as part of the CEDAR [5] project, the HepData [6] database of HEP experimental results can be used to directly export data files usable by Rivet from its Web interface at http://hepdata.cedar.ac.uk/. Analysis histograms are directly booked using the reference data as a binning template, ensuring that data and MC histograms are always maximally consistent.

Rivet is in use within the MC generator development community, particularly in general-purpose shower MC programs, and the LHC experimental community, for MC validation and MC analysis studies which do not require detector simulation.

1.1 Recent developments

Rivet 1.1.3 was released during the first week of this Les Houches workshop, in June 2009: this release includes many new analyses and fixes to existing analyses. Since the workshop, a huge number of extra improvements and developments have taken place in the run-up to the 1.2.0 release. Aside from many technical improvements, and the addition of a large number of QCD analyses (primarily for minimum bias and multi-jet physics) the major conceptual developments have been an emphasis on automated testing and validation of Rivet code, and the removal of hard-coded cross-section normalisations whenever possible. This latter step required more development than may be expected, due to the separation of generator and analysis: the HepMC record had to be enhanced to store cross-section information in a way which can be passed to Rivet. This has now been done, and recent versions of major generators such as Herwig++, SHERPA, and Pythia 8 support this HepMC feature "out of the box". AGILe's generator interfaces have also been updated to write cross-section information into their HepMC output. Determining scaling K-factors where required is now performed via post-processing scripts, which automatically support common approaches to constraining this remaining degree of freedom.

A technical development, but one worth mentioning, has been the emphasis on making Rivet analyses "self-documenting": each analysis has a structured set of metadata specifying name, authors, run conditions, a description, etc., which is used to provide interactive help, HTML documentation, and a reference section in the Rivet manual.

At the time of writing, the final stage of systematic validation of Rivet for the 1.2.0 release is underway. The validation scripts used for this checking will henceforth be included in automatic build tests, to ensure that future developments do not unexpectedly change existing analysis functionality. The final major stage of development is the upgrade of Rivet's histogramming and data analysis code, which is currently rather basic. The upgrade will enable statistically accurate combination of runs, allowing for greater parallelisation of Rivet analyses which require large event statistics.

2. Professor

The Professor system builds on the output of MC validation analyses such as those in Rivet, by optimising generator parameters to achieve the best possible fit to reference data. The main description of Professor's details is found in reference [2], and we will not significantly replicate it here, except in the most high-level sense.

Fundamentally, generator tuning is an example of the more general problem of optimising a very expensive function with many parameters: the volume of the space grows exponentially with the number of parameters and the CPU requirements of even a single evaluation of the function means that any attempt to scan the parameter space will fail for more than a few parameters. Here, the expensive function is running a generator with a particular parameter set to recreate a wide range of analysis observables, using a package such as Rivet. The approach adopted by Professor is to parameterise the expensive function based on a non-exhaustive scan of the space: it is therefore an approximate method, but its accuracy is systematically verifiable and it is currently the best approach that we have.

The parameterisation is generated by independently fitting a function to each of the observable bin values, approximating how they vary in response to changes in the parameter vector. One approach to fitting the functions would be to make each function a linear combination of algebraic terms with n coefficients α_i, then to sample n points in the parameter space. A matrix inversion would then fix the values of α_i. However, use of a pseudoinverse for rectangular matrices allows a more robust coefficient definition with many more samples than are required, with an automatic least-squares fit to each of the

sampled "anchor points": this is the method used by Professor. By aggregating the parameterisations of all the observable bins under a weighted goodness of fit measure – usually a heuristic χ^2– a numerical optimisation can be used to create an "optimal" tune. In practice, many different semi-independent combinations of MC runs are used to provide a systematic handle on the degree of variation expected in tunes as a result of the inputs, avoiding the problem that a single "maximum-information" tune may not be typical of the parameter space.

The first application of Professor, due to its popularity and fairly well-understood steering parameters, was the Pythia 6 MC generator. This was tuned in reference [2], using both of the available parton shower and multi-parton interaction (MPI) models, to data from LEP, SLD, and Tevatron Runs I and II. It was found that the parameterisation method worked well in all cases, and a range of systematic methods and tools were developed to check the accuracy of the approximations, such as line-scans through the parameter space. It was found that a sensible maximum number of parameters to be included in a single tune was ~ 10, and hence, there being ~ 20 Pythia 6 parameters relevant to the studied observables, we separated the tune into an initial stage of final state shower and fragmentation tuning using e^+e^- observables, and then a second stage based on tuning initial state parton shower and MPI parameters to best describe hadron collider data. In these tunes, a quadratic parameterisation was used throughout, this being the simplest suitable function to account for parameter correlations.

2.1 Recent developments

The main development since the initial publication and use of Professor has been the application to more MC tunings. A first extra application was to use the same χ^2 weightings as for the Pythia 6 tune to obtain additional tunings of Pythia 6 with different PDFs. The results of this study, shown at the PDF4LHC meeting in April 2009, indicated that modified leading order PDFs, developed for use with LO MC generators and characterised by a larger than normal low-x gluon component, drive statistical generator tunings in a physically expected direction: the main effect was to increase the screening of MPI effects proportional to the size of the gluon PDF at low x values.

Moving away from Pythia 6, substantial tuning effort has been expended with the Jimmy MPI simulation (used with the Fortran Herwig code), and the Pythia 8 and Sherpa generators. In the case of Pythia 8, the default fragmentation settings are now those fixed by Professor, and use of Professor has identified a problem with description of underlying event (UE) observables, connected to MPI and initial state shower effects: this problem is being addressed. An extremely useful tool in this study was the `prof-I` GUI, which makes use of the Professor parameterisations not for minimisation, but for interactive mimicking of generator responses to parameter changes: this tool makes exploration of speculative tuning ideas easily testable, and helped to verify that no combination of parameters would achieve the desired effect in Pythia 8's description of UE observables.

Another major effort has been the use of Professor to tune and develop the Sherpa generator's simulation of hadronisation and soft initial-state QCD physics. This collaboration has helped to rapidly iterate model improvements and debugging, due to the fast turnaround of tune information.

Professor is additionally being used within the ATLAS, CMS, and LHCb LHC experimental collaborations for tuning studies of the main generators used for their MC simulation, and in plans for re-tuning to first LHC data at new centre of mass energies.

Development of the Professor framework and application to different implications of tuning continues: the next contribution details how ensembles of tunes created by Professor may be used for estimations of tune uncertainties in MC predictions. Other suggested extensions to optimisation of observable definitions or parameterisation of other expensive functions, e.g. observables in SUSY parameter space, remain open to exploration.

References

[1] B. M. Waugh et al., *HZTool and Rivet: Toolkit and framework for the comparison of simulated final states and data at colliders*, arXiv:hep-ph/0605034.

[2] A. Buckley, H. Hoeth, H. Lacker, H. Schulz, and J. E. von Seggern, *Systematic event generator tuning for the LHC*, Eur. Phys. J. **C65** (2010) 331–357, arXiv:0907.2973 [hep-ph].

[3] M. Dobbs and J. B. Hansen, *The HepMC C++ Monte Carlo event record for High Energy Physics*, Comput. Phys. Commun. **134** (2001) 41–46.

[4] M. Cacciari, *FastJet: A Code for fast k_t clustering, and more*, arXiv:hep-ph/0607071.

[5] A. Buckley, *CEDAR: tools for event generator tuning*, PoS **ACAT2007** (2007) 050, arXiv:0708.2655 [hep-ph].

[6] A. Buckley et al., *HepData and JetWeb: HEP data archiving and model validation*, arXiv:hep-ph/0605048.

Acknowledgements

The Rivet and Professor collaborations acknowledge support from the EU MCnet Marie Curie Research Training Network (funded under Framework Programme 6 contract MRTN-CT-2006-035606) for financial support and for many useful discussions and collaborations with its members. A. Buckley additionally acknowledges support from the Scottish Universities Physics Alliance (SUPA); H. Schulz acknowledges the support of the German Research Foundation (DFG).