Department of Physics & Astronomy
Experimental Particle Physics Group
Kelvin Building, University of Glasgow
Glasgow, G12 8QQ, Scotland
Telephone: ++44 (0)141 339 8855 Fax: +44 (0)141 330 5881

UNIVERSITY
of
GLASGOW

Comparator Toolkit Functional Specification

C. Collins-Tooth, A.T. Doyle
Department of Physics and Astronomy
University of Glasgow, Scotland
Email: c.collins-tooth@physics.gla.ac.uk

Abstract

This document provides a description of the functionality of the proposed Comparator Toolkit, which will allow comparison to be made between the full (Geant4) simulation/reconstruction and the fast simulation (ATLFAST). We discuss typical use cases of a physicist, and indicate how these use-cases can be provided for within the existing framework, whilst identifying gaps in the current methods adopted. An important part of this document will be to examine how existing software can be re-used and also to identify gaps in functionality which must be filled.

Comparator Toolkit
Functional Specification

Version 0.99
C. Collins-Tooth, A.T. Doyle
University of Glasgow, Kelvin Building, Glasgow, G12 8QQ, Scotland.

REVISION HISTORY

Version	Date	Initials	Comment
0.1	11/11/04	CCT	New
0.2	17/12/04	CCT	Doc structure changed, references to end, ATLFAST functionality, use cases and comparator functionality added.
0.3	21/12/04	CCT	Pre-publication revisions: add the proposed comparator scope; sizes of ESD/AOD; 'Basic' use-case;
0.99	22/12/2004	CCT	Version distributed for comments to CEB

ABSTRACT

This document provides a description of the functionality of the proposed Comparator Toolkit, which will allow comparison to be made between the full (Geant4) simulation/reconstruction and the fast simulation (ATLFAST). We discuss typical use cases of a physicist, and indicate how these use-cases can be provided for within the existing framework, whilst identifying gaps in the current methods adopted. An important part of this document will be to examine how existing software can be re-used and also to identify gaps in functionality which must be filled.

CONTENTS

1. INTRODUCTION

1.1. Comparator scope

The proposed comparator sits within the complex ATLAS software framework. It will enable the investigation of parameters used in the ATLFAST [1] package, by allowing direct comparison of fully simulated and fast quantities emanating from the same generated events. A simple comparator use case might start with a user who wishes to investigate the validity of ATLFAST parameters in the light of revised detector geometry. Fig. 1 provides a simple outline of how the comparator might perform in the most basic sense. An additional piece of functionality has also been considered as a useful addition to ATLFAST. It is proposed that ATLFAST should be simultaneously able to hold several sets of parameters and corresponding parameterisations for any particular physics process, with these parameterisations being selected at run-time.
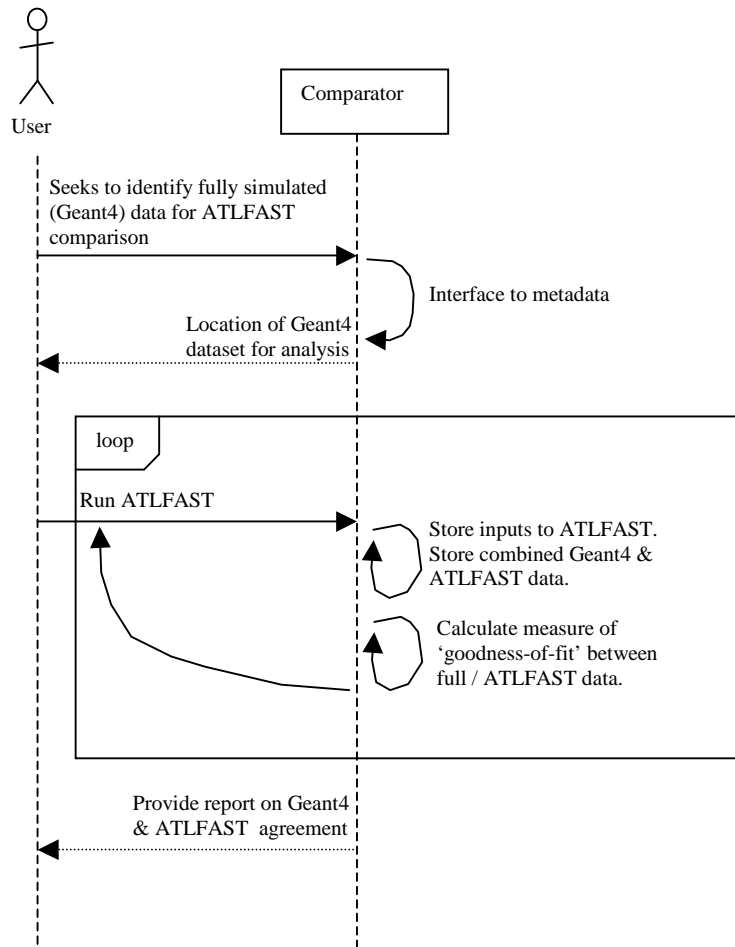
Fig.1. *The 'basic' comparator use case, showing how parameters inside ATLFAST might be investigated.*

The comparator is *not* intended to provide guidance as to when or when not to use the fast simulation. Nor is it intended to provide a method to refine the fast simulation, for example by producing a new parameterisation. Developing a new parameterisation for use inside ATLFAST is an investigation in itself, which must be performed via ad-hoc code changes and testing, followed by a documentation and code release loop. The comparator is intended to allow the choice of existing parameterisations and the variation of parameter values.

1.2. Current analysis procedure

It is important to initially provide an end-to-end overview of the current procedure typically used for the analysis of ATLAS data. Most of the individual use-cases will follow this procedure.
In the future, ATLAS physicists may typically only use a section of this procedure.

Simulated datasets are usually generated (and hadronised) using a variety of tools such as Pythia, MC@NLO , Alpgen, AcerMC, Herwig, Jimmy etc. Most users will not be required to perform this generation step within the formal ATLAS framework (as the various physics groups generally handle generation requests internally and subsequently make the datasets available to the rest of the ATLAS community), but may wish to generate small samples themselves for testing. It is assumed that data files intended to be available for general use will be available via a central catalogue, namely AMI [2]. Another assumption made is that once the LHC is operational, data files gathered using the ATLAS detector will also be centrally catalogued in AMI.

Following generation, there are (at least) two outputs. The first is a small ntuple containing basic ATLFAST [1] quantities, created and smeared with the default values supplied inside ATLFAST. These defaults are brought forward from the previous ATLFAST implementation (Fortran) and the parameter provenances are summarised in [3] and also in [4], though the current parameterisations do not always appear to be in exact agreement with past documentation. A summary of various current ATLFAST smearing parameters is provided in section 1.3.

Quantities in the basic ntuple (e.g. number and momenta of cone-jets, electrons, muons etc) are listed in Appendix A. These ntuples are only likely to be of use for simple validation. The comparator user (wishing to compare fully simulated and reconstructed events with ATLFAST output) must work much further down the analysis chain, after reconstruction has been performed. The other generator output for each dataset is a group of 'evgen' files (in AMI, these are currently called 'partitions'), which together form the generated dataset. These 'evgen' files are used as input to the next stage of the process: either detector simulation, or direct Analysis Object Data (AOD) production ('direct' in the sense that intervening persistifications are skipped, see Fig. 2).

After a dataset has been generated and catalogued, the physics groups usually proceed to run the detector simulation (Geant4) and write out the simulated hits in the ATLAS detector into 'simul' files. The 'simul' files are then digitised (producing 'RDO' files), which may be done either with or without pile-up. Both 'simul' and 'RDO' files for general use are also usually catalogued in AMI.

Assuming that the user has located the 'RDO' files (digitised partitions), the next step is to produce ESD (typically ~500kb /event [5]), and subsequently AOD (typically ~100kb /event [5]). If ESD is *not* needed then AOD can be produced directly, but typically ESD will be required (e.g. by a future comparator user). This is because producing ESD or AOD from 'RDO' files is time-consuming (usually >2min /event - full reconstruction is performed at this stage). However, once ESD is available, AOD can be rapidly produced from it (usually << 1min /event), which is useful, because AOD production may need to be repeated several times (e.g. for ATLFAST parameterisation changes). In contrast, ATLFAST can run over many generated events per second, which should provide continued motivation for refinement.

From a comparator perspective, the ESD acts as a start-point. The quantities which the comparator seeks to compare must all be contained inside the AOD (since the comparator would run ATLFAST during AOD creation). The contents of the AOD and the larger ESD (Event Summary Data) are provided in [6], however it is useful to reproduce the main contents here (see Fig. 3).
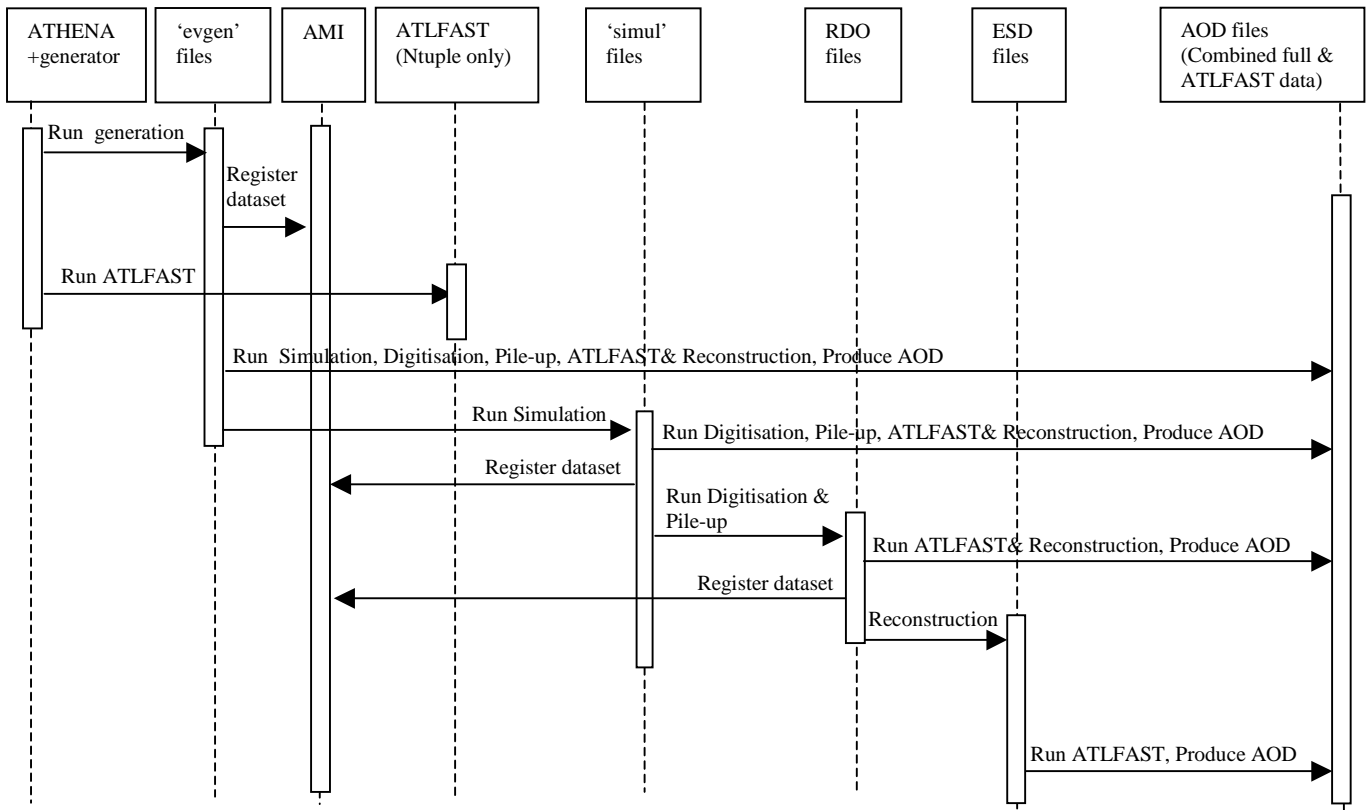
Fig. 2. *A sequence diagram showing the current analysis procedure and the stages involved in producing AOD, which can then be used for a physics analysis. It should be noted that ATLFAST information can be produced and stored in the AOD from the 'evgen', 'simul', 'RDO' or ESD files.*
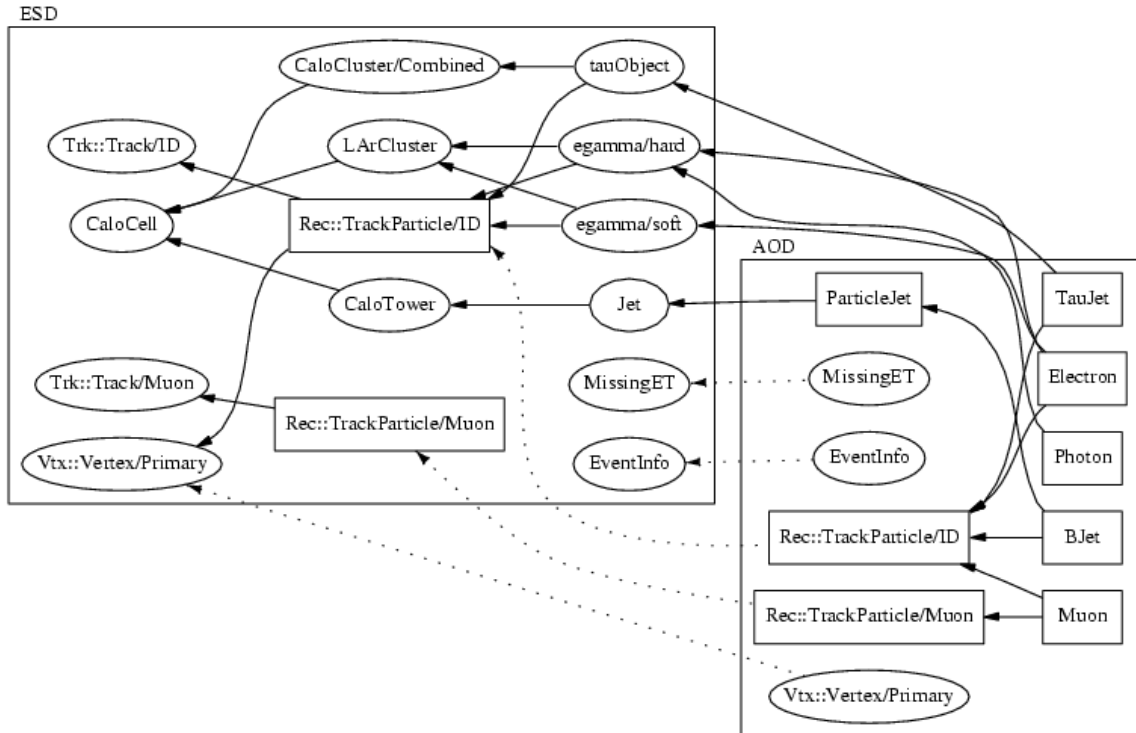
Fig. 3. *Main AOD and ESD objects available in StoreGate using ATHENA release 9 (reproduced from [6]). Some auxiliary objects are omitted. Solid arrows indicate a "pointed to" relationship. Dotted arrows indicate a "copied from" relationship. All objects in square boxes point to the primary vertex, the corresponding arrows are omitted for clarity.*

To produce ESD and AOD, the user supplies commands such as:

```
>athena optRecExToESD.py RecExCommon_topOptions.py > alogfile.log &
```

(produces ESD from 'RDO' files)

```
>athena optESDToCombAOD.py RecExCommon_topOptions.py > anewlogfile.log &
```

(produces combined G4+ATLFAST AOD from ESD)

These can be placed into shell scripts for submission to the LSF batch at CERN. Ganga [7] is able to submit and monitor these shell scripts. Additionally, Ganga should soon have the functionality to implement job submission via the ATHENA Startup Kit (ASK) [8] using metadata harvested from AMI.

Once the AOD exists, a physics analysis can be run against it. A package called 'UserAnalysis' (part of the 'PhysicsAnalysis' [9] suite) has been designed to provide a starting point for novice users. This package has been modified and tested to ensure that ATLFAST and full reconstruction output can indeed be compared using histograms created from the same combined AOD file. Simple back-navigation from AOD to ESD, and back-navigation from AOD to multiple ESD files has also been tested. Recently the Physics Analysis Tools group have begun to provide an interactive analysis environment, which purports to allow on-the-fly histogram creation and fitting [10].

A concrete example of a simple analysis producing ESD and AOD with combined full reconstruction and ATLFAST information is provided in Appendix B.

1.3. Current ATLFAST smearing parameters.

A survey of the locations and some of the values of the smearing parameters provided inside ATLFAST for ATHENA v9.0.2 (ATHENA-ATLFAST) will now be presented. It is intended that these parameters will be used inside the comparator, along with other parameters (e.g. cone sizes for jet-finders etc). As has already been mentioned, the ATHENA-ATLFAST smearing parameterisations derive from [3,4], though they do not always appear to be in exact agreement. An example given here is for the electron momentum smearing in ATHENA-ATLFAST. In contrast, [3,4] do not appear to describe any η dependence at low or high luminosity. It should be noted that many of these parameters are not currently changeable without code recompilation. For the comparator to be useful, code should not need to be recompiled. ATLFAST will have to be modified to enable parameters to be changed at run time. This could be accomplished by 'promoting' the hard-coded parameters into (for example) AtlfastStandardOptions.py. This corresponds to functionality item 5.1, in section 5. (For the sake of tidying up, amalgamating AtlfastStandardOptions.py with AtlfastKStandardOptions.py should be considered).

ElectronSmearer.cxx - parameterises electron energy resolution according to:

Low Luminosity: $|\eta| < 1.4$

$$\frac{\delta E_e}{E_e} = \frac{0.12}{\sqrt{E_e}} \oplus \frac{0.245}{E^T_e} \oplus 0.007$$

$|\eta| \geq 1.4$

$$\frac{\delta E_e}{E_e} = \frac{0.12}{\sqrt{E_e}} \oplus \frac{0.306 \cdot (2.4 - |\eta| + 0.228)}{E_e} \oplus 0.007$$

High Luminosity: $|\eta| < 0.6$

$$\frac{\delta E_e}{E_e} = \frac{0.12}{\sqrt{E_e}} \oplus \frac{0.245}{E_e^T} \oplus 0.007 \oplus \frac{0.32}{E_e^T}$$

$0.6 < |\eta| < 1.4$

$$\frac{\delta E_e}{E_e} = \frac{0.12}{\sqrt{E_e}} \oplus \frac{0.245}{E_e^T} \oplus 0.007 \oplus \frac{0.295}{E_e^T}$$

$|\eta| \geq 1.4$

$$\frac{\delta E_e}{E_e} = \frac{0.12}{\sqrt{E_e}} \oplus \frac{0.306 \cdot (2.4 - |\eta| + 0.228)}{E_e} \oplus 0.007 \oplus \frac{0.27}{E_e^T}$$

PhotonSmearer.cxx – Photon position θ smeared depending on $|\eta|$. Photon energy resolution then smeared based on new $|\eta|$.

MuonSmearer.cxx – parameterisation for muon momentum resolution. NB much of this code is executed using old Fortran routines.

TrackSmearer.cxx – smears track parameters. Uses different manager for electrons, muons & pions. Five correlated gaussian variables are used to smear the true parameters. A large study was performed for muon tracks in [11].

CellSmearer.cxx – smears hadronic jets and cells according to $|\eta|$.

JetSmearer.cxx – smears cluster energies in a similar manner to CellSmearer.cxx, but has high luminosity option, which uses cone sizes to determine an additional value to smear by.

Other parameters *already changeable* without code recompilation, such as:
AtlfastStandardOptions.py / AtlfastKStandardOptions.py – cone sizes, minimum energies, momenta etc.
AtlfastB.cxx – performs rudimentary b-tagging, mis-tagging of heavy and light jets. Probabilities appear to be set based on input parameters.

2. USE CASES

2.1. User wishes to compare full and fast quantities in pre-existing ESD files.

Combined Full (Geant4) and ATLFAST AOD is produced by the user. Full and ATLFAST histograms are produced from the combined AOD via a user-initiated job, which forms part of the comparator. For additional specific physics investigations, it may be possible to develop functionality to allow the inclusion of simple user-defined and interactively created histograms to complement the standard set (defined in section 4). Additionally, assuming that new parameterisations have been developed inside ATLFAST to complement the default set, it may be possible for the user to select which parameterisation (and corresponding set of parameters) they wish to use.

Once a comparison of ATLFAST and fully reconstructed data has been made, the user may wish to increase the ATLFAST sample size to estimate the output when a larger fully reconstructed sample is created. This is likely most expediently accomplished using the ATHENA framework outside the comparator, using job options files such as FastSimToAOD.py.

Functionality Implications:
- 2.1.1. Producing combined AOD must be facilitated.
- 2.1.2. AOD outputs and log files to be named by user and stored.
- 2.1.3. Facilitate production of 'standard comparator' histograms for simple comparison and place into a (user-named) file.
- 2.1.4. A 'comparison job' (e.g. with Kolmogorov or chi-squared tests) should be made available to the user.
- 2.1.5. User-defined and interactively created histograms: The user may wish to write and compile additional analysis based on combined AOD as input. Also, may wish to perform some interactive analysis (see [10], which suggests using PI, on the fly root histogram creation via pyROOT and other interactive tools).
- 2.1.6. User-defined and interactive analysis code and outputs must be stored. (It is unclear as yet how interactive analysis code might be stored).
- 2.1.7. 'comparison job' must be flexible enough to allow user-defined and interactive output to be compared for full and fast outputs.
- 2.1.8. Functionality to enable the choice of parameterisation and a corresponding set of parameters.

Metadata Implications:
- 2.1.9. Cataloguing ESD, AOD, Standard comparator/User-defined/Interactive histogram output and logs. This should include recording the 'parentage' of output files (allows one to retrieve the 'children' of input files).
- 2.1.10. Configuration files should be stored/catalogued where necessary (e.g. ParticleEventAthenaPool/runFastSim.py contains the instance of the options: AtlfastStandardOptions.py, which contains the ATLFAST smearing switches). It may be sufficient to simply store the full job options file in the catalogue, along with the record of the AOD, since the configuration file essentially provides metadata pertaining to the AOD.
- 2.1.11. Require annotation of metadata, such as might be held in the AMI dataset comment field (or equivalent). The aim of this is to allow other users of the comparator to see and search for the results of a comparator study.

2.2. User wishes to tune ATLFAST

ATLFAST is typically tuned by varying input parameters. From the perspective of the user of the comparator, ATLFAST code would usually be run during the step which AOD is produced, hence to

vary ATLFAST input conditions (e.g. smearing parameters), the combined AOD must be re-created from the preceding persistification (ESD).

Functionality Implications:
In addition to various items mentioned in use case 2.1;
2.2.1. ATLFAST inputs (e.g. smearing parameters, cone sizes, cuts etc.) must be easily changeable in the comparator. Upon examination, some of these parameters appear to be contained within ATLFAST code. This would need to be changed so that the code does not need to be recompiled in order to vary the physics inputs.
2.2.2. Multiple AODs will probably be produced. These must be named by the user as is appropriate.

Metadata implications:
2.2.3. Cataloguing multiple AODs (again, should include recording the 'parentage' of output files and the 'children' of input files).
2.2.4. Configuration files used to create the various AODs (e.g. AtlfastStandardOptions.py) should be stored/catalogued with these AODs.

2.3. Automated comparison – a tuning loop.
Use cases 2.1 and 2.2 might be considered to form a loop if a user wishes to find the most suitable value of an ATLFAST parameter. To do this, many iterations of AOD production, comparison and tuning may be required. As such, it would be advantageous to automate the process. It should therefore be possible for a user to select a range of values and a step-size for a parameter. At each intermediate parameter value, the 'goodness-of-fit' of two (one ATLFAST, one fully reconstructed) user-selected comparable quantities should be characterised (e.g. with a chi-squared test) and a report provided to the user on the results.

It is hoped that the loop functionality will enable the best values of existing parameterisations to be chosen. It should also eventually highlight areas where the ATLFAST simulation could be improved, however it is not within the scope of this project to make these improvements. A group interested in the fast simulation of a particular item would, most likely, perform an individual investigation. These investigations should feed back into the ATLFAST software in the form of new parameterisation models.

No significant advantage would be gained by allowing the comparator to produce purely ATLFAST AOD for comparison with some centrally available full-chain AOD, since the reconstruction is already performed prior to ESD production, and additionally the AOD is small in size.

Functionality Implications:
2.3.1. The comparator must provide the user with a list of possible parameters which can be varied.
2.3.2. The user must be able to input a spread of parameter values and a step size, to define a region of interest.
2.3.3. The comparator must be able to edit the parameter value and submit the AOD builds for the desired spread of values.
2.3.4. Once the AOD builds are complete and error free, a specialised comparison job must be run.
2.3.5. The comparison job must be able to:
   o Accept the user defined histograms.
   o Accept **the set** of AODs to process.
   o For each AOD in the set, compare the standard or user-defined histograms and post

the result to be collated with the results for the other AODs.

    o   Produce a plot of the chi-squared values; hopefully providing a suitable minimum.

Metadata Implications:
2.3.6. See use cases 2.1/2.2.
2.3.7. There will be many AODs.
2.3.8. The collated chi-squared fit results must be catalogued and stored, again maintaining a list of the parents of this file (the set of AODs used for the fit).

2.4. Instead of ESD files, user wishes to consult a Tag Database to select events.

Event tags and their relationship to ESD/AOD are detailed in [6]. Assuming that a tag database exists (creation of a tag database should happen when AOD is produced, and so should form part of the general analysis procedure) then a user would wish to process a collection of tagged events. This functionality becomes critical for real data gathered using the ATLAS detector, since events of interest may be spread throughout the available ESD/AOD. Ref. [6] states that the tag collection might be used in two ways: either the 'ATHENA event selector' will follow pointers to deliver events to [the user's] job, or, a run extraction utility takes the collection as input and the events are saved in a personal file. In either case, it is the user who runs a job based on a collection supplied to it.

Functionality Implications:
2.4.1. The comparator must not conflict with general ATLAS functionality to produce tag databases.
2.4.2. The comparator must be able to perform analysis for AOD accessed via a tag database.
2.4.3. The system resource costs for processing events in a tag database could be extremely high, since individual events could be spread over many logical (and physical) files. It would be advantageous if the user were to receive an indication of the resources required for a particular job. Functionality to provide for this may already be in development [6], though this has not yet been verified.

Metadata Implications:
2.4.4. Currently, it seems that AMI will not hold event level metadata. In [6], tag collections are envisaged as POOL [12] collections in local files produced when AOD is created.
2.4.5. Tag data must be able to be treated as a dataset for input to the ATHENA framework in its own right. This should allow the simple processing of a tag collection.
2.4.6. Tag collections will change as more data is added, therefore it is important that the tag database be accessible in a way which allows both new data incorporation (if desired) and also allows a static 'baseline' of tagged events to be used for analysis.

2.5. User wishes to include real data (gathered from the ATLAS detector)

It is envisaged that real data gathered from the ATLAS detector will appear as either 'RDO' or, more likely ESD files. It is unlikely that users will be able to process all available ESD in an analysis job, and events will be likely be spread throughout the ESD. More likely, a tag database (see use case 2.4) will be used to pre-select events of interest in the AOD.

Factors such as use of the conditions database is likely to restrict the data sample being processed into AOD. User-defined cuts and cleaning procedures are also likely to be applied to the real data during the post-AOD analysis/histogramming phase. The comparator should provide users with the ability to produce histograms from real data AOD, in order that they can be compared to the simulations.

Functionality Implications:

2.5.1. The comparator must be able to use real data events selected from AOD (using a tag database) to create histograms.

2.5.2. Real data histograms must be available for comparison with a defined set of simulated data (ATLFAST or full chain simulation) histograms.

2.5.3. It is unlikely that the real data and simulated data (full/ATLFAST) will reside in the same AOD.

Metadata Implications:

2.5.4. When a comparator analysis job is run on real data AOD (using a tag database), the output histograms must be catalogued and stored.

2.5.5. The filter selection used to provide a real data sample (in fact, all the selection criteria applied to the tag database) must be stored in the metadata for the output histograms.

2.6. User wishes to compare ATHENA with pure stand alone generator quantities.

Although this use case does not strictly form part of a comparator comparing full reconstruction and ATLFAST information *per se*, it is envisaged that such functionality would greatly assist the physicist producing their own datasets when they are tuning input generator quantities.

Various generators, hadronisation routines and other add-ons have been configured to work inside the ATHENA framework. These utilities can also be run in a stand-alone manner, often by simply downloading the package, installing it, and providing a set of inputs. Table 1 provides some background to the various packages available. This is not intended to be an exhaustive survey.

The primary outputs of a generator run inside ATHENA are the 'evgen' file, and a log file. The event record is held within the 'evgen' files, and contains information about particles produced. The event record can be processed to give distributions (for example, of charged particle transverse momenta, multiplicities etc.). These distributions will vary based on the inputs to the generator. By comparing the output of a stand-alone generator with the results from the packages inside ATHENA, differences between package versions and inputs can be quickly exposed and rectified. Therefore, it would be advantageous to be able to compare ATHENA based results to the corresponding stand-alone quantities. Typical inputs and instructions for running the various packages are provided at [13].

The full event record is maintained as far as the ESD. The default AOD contains only a 'slimmed down' version of the record, holding only particles judged likely to be 'of interest' to the physicist (see [6]). Comparison of generator-level quantities (such as number of charged particles etc.) should ideally come from the full event record.

It is proposed that a 'pass-through' mode be provided in the comparator to enable the full event record to be created in the AOD, for comparison with stand-alone quantities. This would increase the size of the AOD for this comparator mode. The comparator might then aim to provide automated histogram output for a few typical quantities (see section 4.2).

Functionality Implications:

2.6.1. The comparator must provide a 'pass-through' mode to create AOD with a full event record.

2.6.2. The comparator should provide only simple histogram output in the pass-through mode, to enable a user to make an external comparison with their stand-alone generator.

Metadata Implications:

2.6.3. The 'pass-through' mode histogram output and log files must be stored and catalogued. This should include recording the 'parentage' of output files (allows one to retrieve the 'children' of input files).

Package	In ATHENA		
	Generator	Hadronisation	Other
Pythia	Y	Y +CompHEP +Alpgen +AcerMC	
Herwig	Y	Y +Alpgen, +AcerMC	
+Taola	N	N	Y Taus 'stable', treated by Taola (Pythia,Herwig)
+Photos	N	N	Y Final state QED (Pythia,Herwig)
+Jimmy	N	N	Y Multiparton int. (Herwig)
Alpgen	Y	N (event file passed to hadronisers)	
AcerMC	Y	N	
CompHEP	Y	N	
MadCUP	Y	N	
MC@NLO	Y	N (Hadronised using Herwig only)	

Table 1. *Typical packages used to create simulated data samples*

3. COMPARATOR USE CASES & RELATIONSHIPS TO METADATA USE CASES

Tables 2 and 3 aim to provide a cross-reference outlining the metadata use cases detailed in [14], and how the comparator use cases relate to them.

		Metadata Use Cases												
		Dataset Handling					Analysis			Job Handling				
		1.1	1.2	1.3	1.4	1.5	2.1	2.2	2.3	3.1	3.2	3.3	3.4	3.5
Comparator Use Cases	2.1	X		X	(1)	X	X	X(3)	X	X(2)	X		X	X
	2.2	X		X	(1)	X	X	X(3)	X	X(2)	X		X	X
	2.3	X		X	(1)	X	X	X(3)	X	X(2)	X		X	X
	2.4	X		X	(1)	X		X(3)	X	X(2)	X	X	X	X
	2.5	X		X	(1)	X		X(3)	X	X(2)	X	X	X	X
	2.6	X		X	(1)	X	X	X(3)	X	X(2)	X		X	X

Table 2. *Cross-reference of metadata [14] use cases vs. comparator use cases.*

(1) ATLAS DC2 data is currently held on Castor [15], a storage management system for files which may be migrated between disk and tape storage. ASK [8] symlinks to these files for batch runs.
(2) Presently, jobs are submitted to a batch queue (e.g. LSF at CERN). It is obviously desirable and a major aim of the Ganga [7] development team to allow ESD and AOD job submission to the Grid.
(3) Metadata use case 2.2 is expected be used to produce a tag database to provide a list of events to be processed (in Comparator use cases 2.4, 2.5). It is also envisaged that Metadata use case 2.2 [14] will be used in the creation of new datasets for physics analysis based on metadata-level queries (e.g. Comparator use cases 2.1, 2.2, 2.3, 2.6).

Metadata Use Cases		Comparator Use Cases	
1.1	Read metadata for datasets	2.1	Comparing Full/Fast using ESD files
1.2	Update metadata for a dataset	2.2	User wishes to tune ATLFAST
1.3	Resolve physical data	2.3	Automated comparison – tuning loop
1.4	Download dataset to a local disk	2.4	User wishes to use tag database
1.5	Specify a new dataset	2.5	User wishes to use real data
2.1	Run a physics simulation program	2.6	Compare ATHENA with standalone gen.
2.2	Select a subset of a dataset		
2.3	Run an algorithm over an input dataset		
3.1	Submit a job to a grid		
3.2	Retrieve/access the output of a job		
3.3	Estimate system resource cost of job		
3.4	Monitor progress of a job		
3.5	Repeat a previous job		

Table 3. *Use case 'quick-reference' for metadata [14] and the proposed comparator.*

4. CORE COMPARISON QUANTITIES

4.1. Core comparator quantities

It is useful to define a set of core physics quantities on which the comparator toolkit might initially focus. Comparisons of ATLFAST output to the results of full reconstruction (Geant4) might include:

- Jet Distributions (for different jet algorithms e.g. kt, cone):
    - Pt, E
    - checking result is flat in azimuthial (phi) space,
    - eta distributions etc.
- Particle distributions (for electrons, muons etc)
    - Pt, E
    - phi,
    - eta
- Vertex distributions
- Track multiplicities
- B-tagging quantities
    - Impact parameter resolution
    - Tagging efficiency/ purity

4.2. Comparator quantities in 'pass-through' mode

By default the AOD contains only a slimmed down event record. In 'pass-through' mode, the AOD could be built with the full event record. Comparison of the ATHENA output at generator level with the stand-alone generator quantities might then include (at particle level):

- The first derivative of the number of charged particles with respect to eta and Pt

$$\frac{dN_{ch}}{d\eta} \text{ vs } \eta \quad , \quad \frac{dN_{ch}}{dPt} \text{ vs } Pt$$

- Cross section comparison - acts as a good cross check on the PDFs used as input.
- Momentum conservation – are the particles produced balanced in momentum terms?

5. PROPOSED FUNCTIONALITY

Fig. 4 shows how the proposed functionality might fit into the existing ATLAS software framework. The desired functionality will now be prioritised.

5.1. For each physics area inside ATLFAST (e.g. electrons, muons, tracks, photons, b-tagging, cell energies and jets), relevant parameters for the smearing must be extracted from the code and placed into job options files.

5.2. Comparison between full and ATLFAST quantities (those defined in section 4) should be enabled, and should output a measure of 'goodness of fit' between full and ATLFAST output.

5.3. Allow parameter variation to check/improve full and fast agreement.

5.4. Enable ATLFAST to hold multiple parameter sets and parameterisations; allow the comparator user to select which parameter set (and corresponding parameterisation) they wish to use.

5.5. Automate variation and comparison to allow best fit to be found. Use range/step size.

5.6. Publish metadata on results, possibly in AMI.

5.7. Publish output results – best fits etc. – possibly by default in a commonly accessible area.

Existing functionality to be considered:
- Ganga handles data selection (via AMI) and job submission/monitoring.
- Ganga JOE allows job-option files to be edited (will allow parameter variations).
- Ganga Athena Application Handler should allow package check-outs, modifications.
- AMI is now accessible (in query-only format) from Ganga.
- Muon Track smearing parameterisation in [11] appears to provide better description than existing ATLFAST functionality.
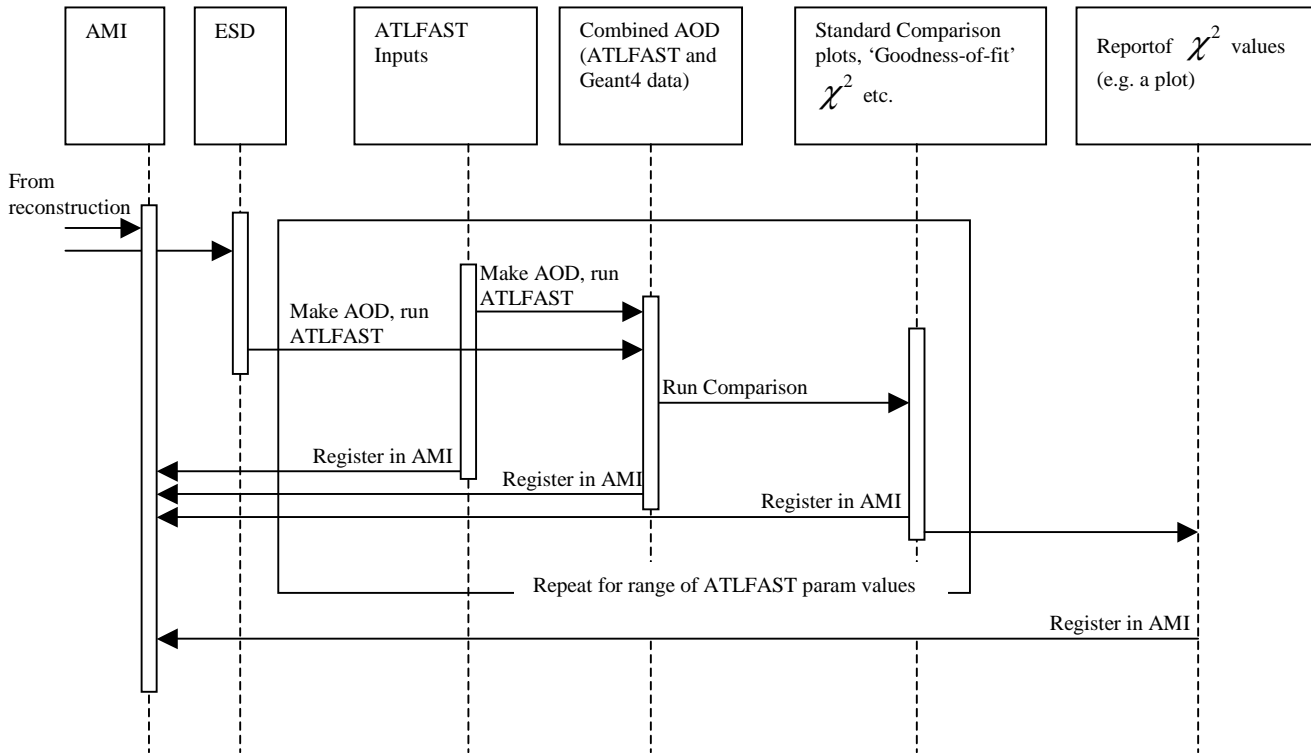

Fig. 4. *The proposed comparator functionality represented as a sequence diagram.*

6.   REFERENCES

[1] ATLFAST User guide
Available at: http://www.hep.ucl.ac.uk/atlas/atlfast/UserGuide.html

[2] Atlas Metadata Interface (AMI) User guide v.2.0.0
Available at: http://isnpx1158.in2p3.fr:8180/AMI/AMI/doc/pdf/AMIUserGuide_200.pdf

[3] ATLFAST documentation and parameter provenances
Available at: http://www.hep.ucl.ac.uk/atlas/atlfast/ATLFAlgorithm.ps

[4] ATLAS-PHYS-98-131, Atlfast 2.0 – A description of the Fortran implementation of ATLFAST.
Available at: http://cdsweb.cern.ch/

[5] ATL-SOFT-2004-007, The ATLAS Computing Model.
Available at: http://cdsweb.cern.ch/

[6] ATL-SOFT-2004-006, Final Report of the AOD/ESD Definition Task Force Revision 2
Available at: http://cdsweb.cern.ch/

[7] Ganga User guide v2.
Available at: http://ganga.web.cern.ch/ganga/user/index.html

[8] ATHENA Startup Kit manual
Available at: http://wlav.home.cern.ch/wlav/athena/athask/

[9] PhysicsAnalysis Tools Documentation
Available at: http://www.usatlas.bnl.gov/PAT/

[10] Physics Analysis Tools Presentations (16/11/2004)
Available at: http://agenda.cern.ch/fullAgenda.php?ida=a045142

[11] A Parameterization for Fast Simulation of Muon Tracks in the ATLAS Inner Detector and Muon System, A. Salzburger, Diploma Thesis (2003)
Available at: http://physik.uibk.ac.at/hephy/theses/dipl_as.pdf

[12] POOL Pool Of persistent Objects for Lhc
Information available at: http://lcgapp.cern.ch/project/persist/

[13] Ian Hinchcliffe's web-based information for Atlas event generation.
Description of packages: http://www-theory.lbl.gov/%7Eianh/monte/Generators/
Job Options used for various (DC2) samples:
http://atlassw1.phy.bnl.gov/lxr/source/atlas/Physics/GeneratorOptionsDC2/share/
Sample Job Options for packages: http://phyweb.lbl.gov/%7Eianh/monte/Generators/samples/

[14] Metadata Use Case Document
Available at: http://ppewww.ph.gla.ac.uk/~shanlon/Metadata/CoreUseCases_v5.pdf

[15] CERN Castor storage system.
Overview available at: http://it-dep-fio-ds.web.cern.ch/it-dep-fio-ds/Documentation/userguide.html

[16] ATHENA user guide and tutorial v2
Available at:
http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/OO/architecture/General/Tech.Doc/Manual/2.0.0-DRAFT/AthenaUserGuide.pdf

[17] Configuration Management Tool (CMT)
'Bootstrap' procedure for setup available at:
http://www.lal.in2p3.fr/cgi-bin/cvsweb/cvsweb.cgi/Atlas/doc/userguide/userguide.html?rev=1.5

[18] Personal record and method of production for ESD, AOD and analysis histograms
Available at: http://ppewww.ph.gla.ac.uk/~chrisc/logs/paedm881-pcgla02.html

APPENDIX A - ATLFAST Validation Ntuple Structure:

BLOCK	VARIABLE	DESCRIPTION
PLEPTONS	NELE	Number of isolated electrons
PLEPTONS	KFELE(NELE)	pdg_id of isolated electrons
PLEPTONS	PXELE(NELE)	X momentum of isolated electrons
PLEPTONS	PYELE(NELE)	Y momentum of isolated electrons
PLEPTONS	PZELE(NELE)	Z momentum of isolated electrons
PLEPTONS	EEELE(NELE)	Energy of isolated electrons
PPHOTONS	NPHO	Number of isolated photons
PPHOTONS	KFPHO(NPHO)	pdg_id of isolated photons
PPHOTONS	PXPHO(NPHO)	X momentum of isolated photons
PPHOTONS	PYPHO(NPHO)	Y momentum of isolated photons
PPHOTONS	PZPHO(NPHO)	Z momentum of isolated photons
PPHOTONS	EEPHO(NPHO)	Energy of isolated photons
PLEPTONS	NMUO	Number of isolated muons
PLEPTONS	KFMUO(NMUO)	pdg_id of isolated muons
PLEPTONS	PXMUO(NMUO)	X momentum of isolated muons
PLEPTONS	PYMUO(NMUO)	Y momentum of isolated muons
PLEPTONS	PZMUO(NMUO)	Z momentum of isolated muons
PLEPTONS	EEMUO(NMUO)	Energy of isolated muons
PMUXS	NMUX	Number of non-isolated muons
PMUXS	KFMUX(NMUX)	pdg_id of non-isolated muons
PMUXS	PXMUX(NMUX)	X momentum of non-isolated muons
PMUXS	PYMUX(NMUX)	Y momentum of non-isolated muons
PMUXS	PZMUX(NMUX)	Z momentum of non-isolated muons
PMUXS	EEMUX(NMUX)	Energy of non-isolated muons
PPJETS	NJET	Number of reconstructed jets
PPJETS	KFJET(NJET)	pdg_id of reconstructed jets tag
PPJETS	PXJET(NJET)	X momentum of reconstructed jets
PPJETS	PYJET(NJET)	Y momentum of reconstructed jets
PPJETS	PZJET(NJET)	Z momentum of reconstructed jets
PPJETS	EEJET(NJET)	Energy of reconstructed jets
PPJETS	PTcalo(NJET)	Transverse Momentum in calorimeter
PPJETS	PTbjet(NJET)	Transverse momentum of b-jets
PPJETS	PTujet(NJET)	Transverse momentum of u-jets
PPJETS	NJETB	Number of reconstructed jets calorbrated with AtlfastB
PPJETS	KFJETB(NJET)	pdg_id of reconstructed jets tag calorbrated with AtlfastB
PPJETS	PXJETB(NJETB)	X momentum of reconstructed jets calorbrated with AtlfastB
PPJETS	PYJETB(NJETB)	Y momentum of reconstructed jets calorbrated with AtlfastB
PPJETS	PZJETB(NJETB)	Z momentum of reconstructed jets calorbrated with AtlfastB
PPJETS	EEJETB(NJETB)	Energy of reconstructed jets calorbrated with AtlfastB
PHISTORY	NPART	Total number of status 3 particles
PHISTORY	KPPART(NPART)	"bar code" of particle
PHISTORY	KSPART(NPART)	status particle
PHISTORY	KFPART(NPART)	pdg_id of particles
PHISTORY	KPMOTH(NPART)	"bar code" of particles mother
PHISTORY	KFMOTH(NPART)	pdg_id of particles mother
PHISTORY	PXPART(NPART)	X momentum of particles
PHISTORY	PYPART(NPART)	Y momentum of particles
PHISTORY	PZPART(NPART)	Z momentum of particles
PHISTORY	EEPART(NPART)	Energy of particles
INFO	ISUB	Process
INFO	JETB	Number of b-tagged jets
INFO	JETC	Number of c-tagged jets
INFO	JETTAU	Number of tau-tagged jets
PMISSING	PXMISS	Measured missing X momentum
PMISSING	PYMISS	Measured missing Y momentum
PMISSING	PXNUE	X momentum of invisibles
PMISSING	PYNUE	Y momentum of invisibles

APPENDIX B - An Example Analysis

The process of producing ESD and AOD, and subsequently performing an analysis is complex, and may be better illustrated by a concrete example. The following instructions illustrate the *current* process of producing ESD, AOD and some analysis plots in the 'UserAnalysis' package (provided as part of the 'PhysicsAnalysis' [9] suite). It uses ATHENA v8.8.1[16], and relies on the user having set up CMT (Configuration Management Tool) [17] to produce a requirements file in a directory named 'cmthome'. This example also requires that the path corresponding to the user's test area (where all the code is checked out) corresponds to the environment variable '$TEST'.

1) Check out the ESD/AOD production code and compile it:

set up requirements file for ATHENA v8.8.1 in cmthome/requirements (see [17])
source cmthome/setup.sh -tag=opt
cd $TEST

check out packages:
cmt co -r EventInfo-00-02-07 Event/EventInfo
cmt co -r RecExCommon-00-02-72 Reconstruction/RecExample/RecExCommon

For performance reasons, disable CBNT and muonbox in:
$TEST/Reconstruction/RecExample/RecExCommon/RecExCommon-00-02-72/share/RecExCommon_flags.py

cd $TEST/Reconstruction/RecExample/RecExCommon/RecExCommon-00-02-72/cmt
do the "c,s,g" step to build the executables:
cmt broadcast cmt config
source setup.sh
cmt broadcast gmake

get distribution copies of some required files:
cd ../run/
source ../share/RecExCommon_links.sh

Make a retrospective change to RecExCommon_topOptions.py. At the end of the file add:
RDBAccessSvc = Service("RDBAccessSvc")
RDBAccessSvc.HostName = "pdb01"

set up poolFileCatalog.xml:
Firstly, get a copy of PoolFileCatalog.xml which contains the data you wish to process (you can add files using pool_insertFileToCatalog , but it is a lengthy process for large data files)
Put the copy of PoolFileCatalog.xml in an area of your homespace.
Set up as follows:
export   POOL_CATALOG=xmlcatalog_file:/afs/cern.ch/user/<location of Poolfile>/PoolFileCatalog.xml

You should still be in the /run directory. Now, get the following files:
get_files RecExCommon_topOptions.py
get_files optRecExToCombAOD.py
get_files FastSimToAOD_jobOptions.py
get_files optRecExToESD.py
get_files optESDtoCombAOD.py

and set optRecExtoESD.py to run over a data file (e.g.):
PoolRDOInput=["LFN:zee_RDO_extract.pool"]

(aside - you can also specify PFNs and outputs, if desired)
PoolRDOInput=["rfio:/castor/cern.ch/atlas/project/dc2/..."]
PoolESDOutput = "testESD.pool.root"
PoolESDInput = ["testESD.pool.root"]
PoolAODOutput = "testAOD.pool.root"

2) Run ATHENA to create ESD, AOD:

athena optRecExToESD.py RecExCommon_topOptions.py > toESD.log &
(produces ESD from RDO file)
athena optESDtoCombAOD.py RecExCommon_topOptions.py > toAOD.log &
(produces AOD from ESD)

Running ATHENA can also be accomplished from inside Ganga. This has the advantage that jobs are then monitored in the GUI.

3) Check out and compile the analysis package:

An analysis job can be run against AOD to produce physics output.
Check out the analysis package:
cd $TEST
cmt co -r UserAnalysis-00-01-08 PhysicsAnalysis/AnalysisCommon/UserAnalysis

Reconfigure the RecExCommon requirements file (RecExCommon/RecExCommon-00-02-72/cmt/requirements) by adding a line:
use UserAnalysis UserAnalysis-* PhysicsAnalysis/AnalysisCommon

Re-compile everything:
cd $TEST/Reconstruction/RecExample/RecExCommon/RecExCommon-00-02-72/cmt
cmt broadcast (to check all packages picked up)
cmt broadcast cmt config
source setup.sh
cmt broadcast gmake

cd ../run
get_files AnalysisSkeleton_jobOptions.py
(This will need changes – the users want it to use their own AOD as an InputCollection)edit
AnalysisSkeleton_jobOptions.py and change the InputCollections section:
  EventSelector.InputCollections = [
                      "AOD.pool.root"
    #   "AOD_Zee.root",
    #   "AOD_Zmm.root",
    #   "AOD_Ztt.root"
                                    ]

4) Run the analysis package:

athena.py AnalysisSkeleton_jobOptions.py > AnalysisSkeleton.log &
This produces AnalysisSkeleton.root, which contains some physics output from the full Reconstruction.

The UserAnalysis package has been modified to enable some simple ATLFAST and truth level quantities to be similarly stored in histograms for comparison. In addition, quantities from the ESD (such as Egamma energy) have been successfully retrieved via back-navigation from the AOD. Information on the changes made to the UserAnalysis package can be found in [18].
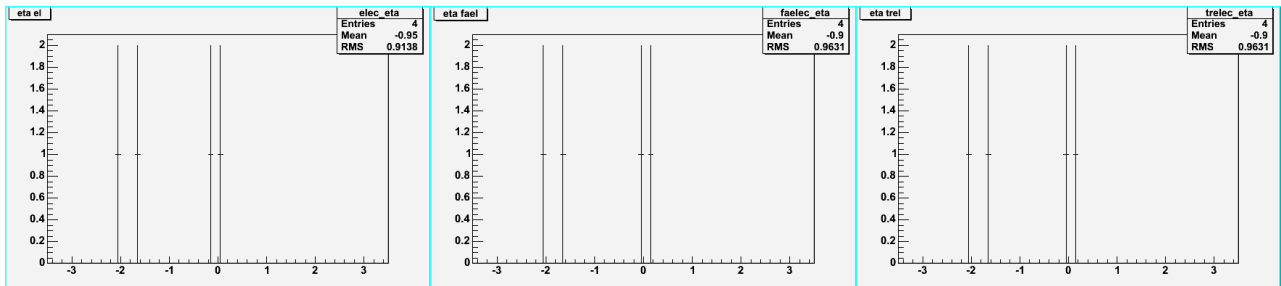

Fig. 5. Electron eta distributions for two Z->ee events, compared for full Geant4 reconstruction (left), ATLFAST (centre) and truth-level information (right).

APPENDIX C - Dependencies

The comparator aims to re-use existing code where possible. As such, it has a number of dependencies, which are summarised here.

ATLFAST – continued support and development.

ATHENA framework (+Pool/seal etc)

ASK – Athena Startup Kit

PhysicsAnalysis package

Ganga – in particular the ADA/DIAL compatibility for the future.

Atlas Metadata Interface – Comparator must be able to create datasets and assign parents. Also must consider storage of log-files and user comments.

Root – How to include user-defined histograms in the comparator.

GLOSSARY

AMI Atlas Metadata Interface

AOD Analysis Object Data (design:100k / event)

CMT Configuration Management Tool

ESD Event Summary Data (design: 500k/event; in practice ~1.4Mb/event)

LFN Logical File Name

PDF Parton Distribution Function – describes the densities of the partons inside a hadron as a functrion of the struck parton momentum fraction and the 4-momentum transfer of the event.

PFN Physical File Name

RDO Reconstructed Data Object files. Users may run ATLAS reconstruction against these files.