UCL/HEP 2000

ATLAS Software and Higgs Searches via Weak Vector Boson Scattering at the LHC

1^{st} Year Transfer Report

Richard Steward
Department of Physics and Astronomy
University College London

Abstract

This report summarises the analysis, design and implementation of a new, object-oriented ATLAS fast detector simulation. The software is implemented using the ANSI C++ programming language, and will be executable within ATHENA (the ATLAS software architecture framework).

Also presented is a physics study of Higgs detection via WW fusion at the LHC, using the decay mode $H_0 \rightarrow WW \rightarrow e^{\pm}\nu\mu^{\pm}\nu$. The Pythia version 5.7 event generator initialised for pp collisions at 14TeV is used. The detector simulation and analysis is carried out using Root/ATLFAST++.

1 Introduction

1.1 ATLAS Software

Following a collaboration wide decision to develop all software using object-oriented methods, the ATLAS Software Architecture group have provided an application framework which will be usable for implementing the full range of offline computing tasks. This framework is called Athena and is based on the LHC-b framework, Gaudi[1].
Fast detector simulation is an important first step in offline physics analyses. ATLFAST[2] was the original ATLAS fast detector simulation, written using the Fortran programming language, with event generation from the Pythia[3] monte carlo generator. This was followed by ATLFAST++, the C++ version which is executable within Root[4], again using Pythia.
The UCL ATLAS software group[5] is currently designing and implementing a new fast detector simulator, based on the previous ATLFAST simulators which will be executable from within the new ATLAS framework.

1.2 Higgs Searches at the LHC

In the Standard Model of particle physics, electroweak gauge symmetry is broken via the Higgs mechanism, leading to the existence of one neutral scalar Higgs particle.
Although the Higgs mass is not predicted by theory, direct searches at LEP2 set a lower bound of $M_h > 95$ GeV[6], while vector boson scattering violates unitarity if the Higgs mass exceeds $M_h \approx 800$ GeV[7]. Precision electroweak data constrain the Higgs mass to $M_h = 76^{+85}_{-47}$ GeV.[6]
Weak vector boson scattering is a promising channel for the discovery of an intermediate mass (80GeV $< \mathrm{M_h} <$ 200GeV) Higgs boson at the LHC. Although gluon fusion dominates, weak boson fusion has the second largest production cross-section for the SM Higgs boson in hadronic interactions[8].

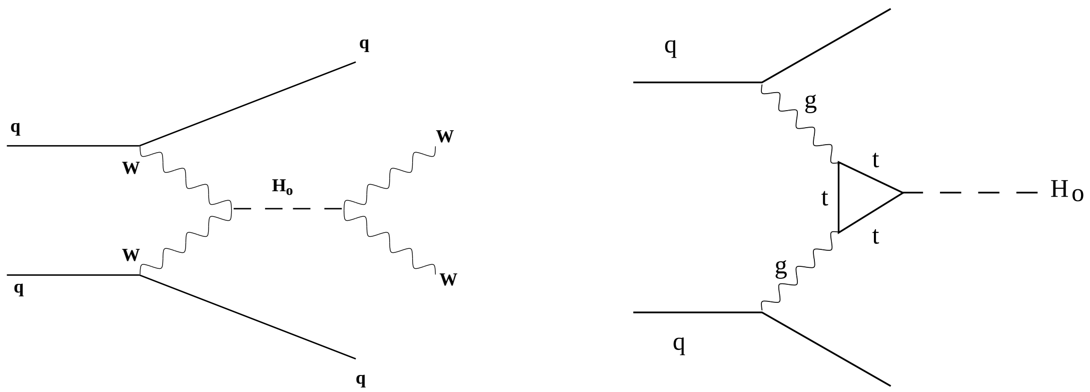

Figure 1: *The Dominant Higgs Production Processes in Hadronic Interactions.*

While higgs production via gluon fusion has substantial irreducible background, weak boson fusion events contain information in observable quark jets, as well as suppressed hadronic activity in the central region.

2 The ATHENA architecture framework

The purpose of the ATLAS architecture framework is to provide software which fulfils the requirements of the physicist developer and physicist analysist in the construction and use of programs for performing common tasks, such as reconstruction, detector simulation, visualisation etc.

ATHENA has an object-oriented design, guaranteeing ease of use, flexibility and well-specified methods for using the services that the architecture provides.
The application framwork consists of a number of component software blocks which have functionality provided by well-defined interfaces. This interface is the component's link to the outside world. Communication to and from this object can only take place in accordance with the rules specified by the methods defined on the interface.
Thus, the functionality of each software component is 'encapsulated' within the object, and any user of this object therefore does not, and should not, know how the object is performing its task.
This increases ease of use, as only the methods specified on the interface need to be known to make use of the object . Indeed, the functionality may change without the user ever realising.

Almost all software written for use within ATHENA will be in the form of specialisations of a few specific components; derived objects with enhanced functionality, yet the same interface. This is achieved using C++'s public inheritance mechanism. The base classes provided by ATHENA are

- Data Object
- Algorithm
- Converter

This report will only cover the first two classes, as these are relevant to the redesign of the ATLAS fast simulator.

2.1 Data Objects and Algorithms

Most physics tasks consist of manipulating mathematical and physical quantities such as vectors, matrices, energies and momenta by algorithms specified usually in the form of equations. The ATHENA application framework makes a clear distinction between 'data' objects and 'algorithm' objects.
Quantity-like entities, such as hits, vectors, cells or even electrons should be implemented with a derived class which inherits from the *DataObject* base class.
Procedural type entities which act on data objects, such as energy smearers, cell depositors or trackfinders are implemented with a derived class which inherits from the *Algorithm* base class.

Principally, an algorithm takes input data, manipulates it and produces new output data. The figure shows how a concrete algorithm object interacts within the application framework. There are five main services that *Algorithm* objects use:

- The event data store
- The detector data store
- The histogram service
- The message service
- The job options service

These services are provided by software components and are accessed via an inteface. These interfaces are C++ pure abstract base classes; they have no data members and contain no implementation of the methods that they declare.
Figure 2 shows the interfaces that algorithm objects use to access these services.
The event and detector data store services are used to transfer data to and from a persistent store or other algorithm using the *IDataProviderSvc* interface.
The histogram service is used for storage of statistical objects via the *IHistogramSvc*.
The message service provides a way in which algorithms can make reports on its progress, for example error messages or debug notices. This is accessed using the *IProperty* interface.
Parameters such as convergence criteria, cut-off values and algorithm ordering may be specified at run time using the job options service. This makes use of the *IProperty* interface.
As shown in the figure, the *Algorithm* base class makes use of the *IAlgorithm* and *IProperty* interfaces. Therefore any concrete algorithm class must implement these interfaces.

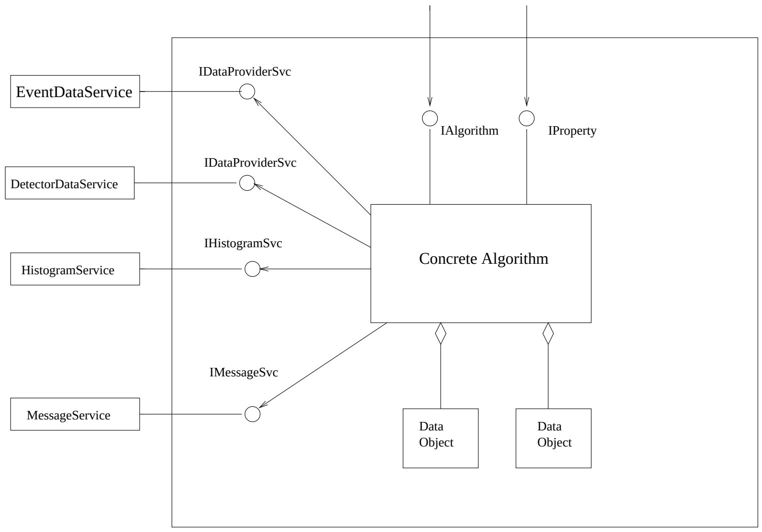

Figure 2: *Framework as seen by an Algorithm Object*

3 ATLFAST design and functionality

The ATLFAST fast detector simulation package simulates the key aspects of the ATLAS detector response in a parametrised way. Using a list of monte carlo particles from the pythia event generator, it produces the following output information:

- reconstructed jets
- isolated electrons and photons
- isolated and non-isolated muons
- missing transverse energy
- simple trigger response
- charged track parameters

In addition ATLFAST-B has been implemented which randomly simulates b-jet tagging, τ-tagging, τ-veto and performs a P_T dependent recalibration of jet energies.

ATLFAST executes using independent modules which communicate via a standardised object bus.
The ATLFast class controls the execution flow and I/O. Only one instance of this class is created and can be accesed via a global gATLFast pointer.
This class executes a pipeline of modules called 'makers', which mainly perform the monte carlo particle deposition in the ATLAS detector. The makers derive from an ATLFMaker object, and their output can be stored in a *Root* tree for access by the global pointer to the ATLFast object.

- MCMaker - Organises generation of the physics process via the Pythia event generator.
- ClusterMaker - Deposits the energy of generated particles into $\Delta\eta$ x $\Delta\phi$ calorimeter cells and reconstructs clusters for a given cone and minimum E_T.
- ElectronMaker - Uses list of clusters to store only isolated electrons. Calorimeter clusters associated with isolated electrons are then discarded. Electron energy is smeared using parametrized resolutions.
- PhotonMaker - Uses list of clusters to classify photons as isolated or non-isolated. Calorimeter clusters associated with isolated photons are then discarded. Photon energy and angular position is smeared using parametrized resolutions.
- MuonMaker - Uses list of clusters to classify muons as isolated or non-isolated, which are then stored separately. Energy is then smeared using parametrized resolutions.
- JetMaker - Remaining clusters are smeared with parametrized resolutions and classified as jets. b, c and τ-jet labelling is performed.

- TrackMaker - Using the list of generated particles, helix parameters of charged particles are reconstructed and stored. Smearing is implemented using parameterised resolutions.

- MiscMaker - Calculates missing transverse energy from all reconstructed objects. This object also includes an event summary and the number of reconstructed objects.

- TriggerMaker - Validates physics event with a simple trigger procedure, using muons, photons, isolated electrons and reconstructed jets.

4 Re-design of ATLFAST within ATHENA

To begin with, the fast detector simulator will be implemented within ATHENA with the functionality remaining as close as possible to the Root version, in order to perform comparative testing of the new package.
At present, the package is little more than a smeared monte carlo simulation, although the new design will be flexible enough to allow simulation of the detector response to an arbitrary degree of complexity, while still remaining faithful to the interface provided.
In the context of the ATHENA framework, it is clear that the *Maker* objects from ATLFAST should be implemented as concrete *Algorithms*, while their output classes are *DataObjects*.

The input for ATLFAST will be the standard format provided by the HepMC classes[9]. This is an object-oriented event record written in C++ for Monte Carlo generators.
The HepMC::Particle class instances from the MCEvent will be stored in ATHENA's Transient Data Store, which *Algorithm*s can access via the *IDataProviderSvc* interface mentioned previously.
The reconstructed entity makers are then implemented as instances of the same object which access the MCEvent in order to create entities such as electrons, muons, clusters etc.
A class diagram of the entity makers and entities themselves is shown using the Unified Modelling Language in Figure 3.

Reconstructed entities such as isolated electrons and cells are instances of the *ReconstructedParticle* object. These implement the *IKinematic* interface which all four-vector like entities obey. They also derive from *ContainedObject* which is the ATHENA data object base class which can be stored in STL-like containers in the Transient Data Store.

Entity makers are instances of the *DefaultReconstructedParticleMaker*. These implement the *IReconstructedParticleMaker* interface. From the UML diagram, it can be seen that the methods on the interface show the processes that all entity makers perform.

- getMC - accesses TDS for MCEvent data

- create - creates relevant *ReconstructedParticle* objects

- smear - smears energies of *ReconstructedParticle* objects
- isAcceptable - criteria for storage (eg. isolated for electrons)
- store - stores container of *ReconstructedParticle* objects in TDS

Concrete instances of *ReconstructedParticleMakers* are, of course, algorithms and must therefore implement the *Algorithm* base classes methods.
These are simply a specialised constructor and three methods.

- Initialise initialises any algorithm specific parameters.
- Execute performs the main function of the algorithm ie. self-messaging the implementation of the *IReconstructedParticleMaker* interface methods.
- Finalize performs any post-execution functions.

These methods all return a StatusCode object which reports the progress of the method; SUCCESS or FAILURE.
MuonMaker, CellMaker and ClusterMaker instances are in the process of implementation.
All code can be found at **http://www.hep.ucl.ac.uk/~ rms/phd.html**.

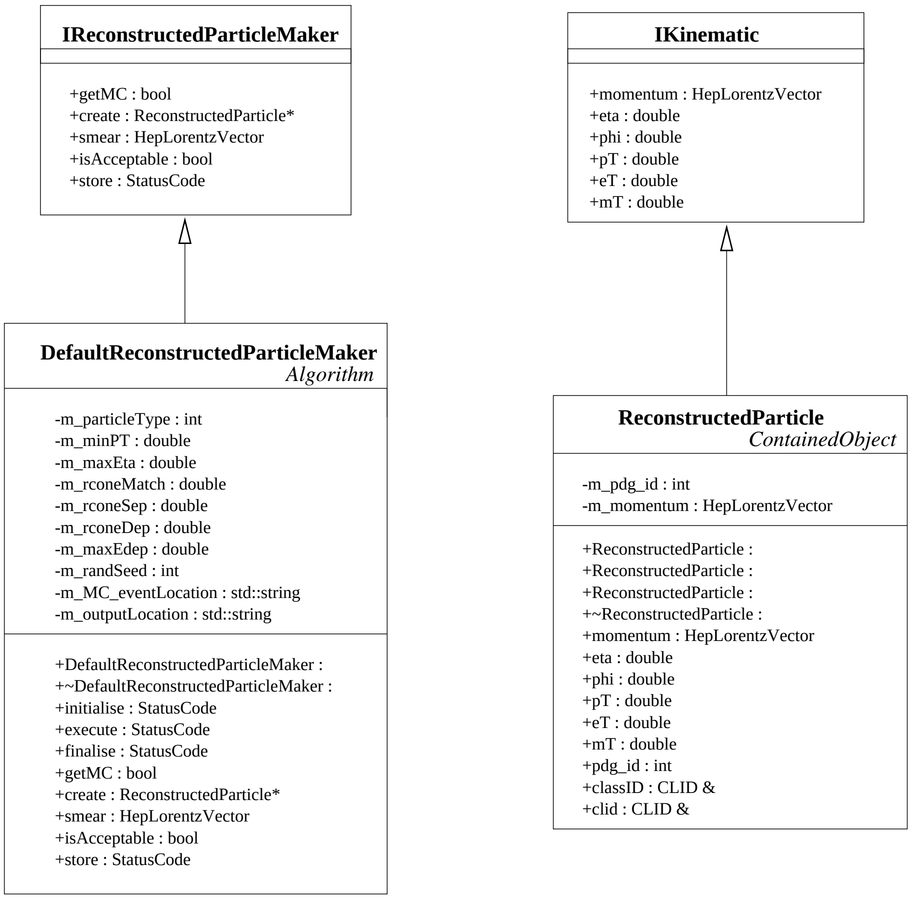

Figure 3: *UML Diagram for new ATLFAST Classes*

5 Higgs Searches using the $H_0 \to WW \to e^{\pm}\nu\mu^{\pm}\nu$ decay mode

Experimental evidence has, for some time, suggested a small Higgs boson mass. It is for this reason that searches for the Higgs boson in the intermediate mass region 155 $GeV < M_h < 180$ GeV is particularly important.[10]
Outside this mass region, Higgs detection is possible by the observation of narrow mass peaks using the decays $H_0 \to \gamma\gamma$ and $H_0 \to Z^0Z^0 \to 2l^+2l^-$. However, in the intermediate mass range, the Higgs decay to W bosons is close to unity, thus the latter gold-plated signature is heavily suppressed.
Despite larger branching ratios, identifying W or Z^0 decays into jets is difficult due to the hadronic background at the LHC. Although it is not possible to reconstruct a Higgs boson mass peak due to the decay into two neutrinos, the $H_0 \to WW \to e^{\pm}\nu\mu^{\pm}\nu$ decay ratio is approximately 100 times greater than the gold-plated channel, thus an excess of events may be observed, and used to identify the presence of a Higgs boson.[10]

5.1 Background Processes

For $W^{\pm}$ and Z^0, isolated high P_T electrons and muons are typically clean and detectable, although there are a number of substantial background processes.[11]

- Continuum W production constitutes the dominant irreducible background, with a σ x BR of between 5 and 7 times larger than the signal.
- The largest reducible background with isolated leptons in the final state comes from $t\bar{t}$ and Wt. This background has a σ x BR between 40 and 50 times larger than the signal.
- The largest reducible background with non-isolated leptons in the final state comes from $Wb\bar{b}$ and direct $b\bar{b}$ production, containing one or two leptons from semi-leptonic b-decays. This background has a σ x BR over 130 times larger than the signal. These can be greatly suppressed by P_T and isolation cuts.

5.2 Signal and Background Discrimination Criteria

Despite the large contributions from background processes, there are a number of suggested selection criteria to increase greatly the signal to background ratio for the leptonic final state.[10] These selection criteria are not optimized, but will be used in the following analysis.

- The event should contain two leptons with opposite charge with a minimal P_T of 10 GeV
- The pseudorapidity $|\eta|$ of each lepton should be less than 2

- The di-lepton mass should be smaller than 80 GeV
- The missing P_T should be larger than 20 GeV
- The di-lepton opening angle in the plane transverse to the beam should be smaller than 2.5 radians
- Jet pseudorapidity $|\eta|$ should be greater than 2.4

The first two cuts mainly select decays of W and Z bosons. The remaining criteria reduce lepton pairs from single Z production and τ lepton decay. $t\bar{t}$ and Wtb are strongly reduced by the final cut jet cut. The remaining continuum, $t\bar{t}$ and Wtb background can be suppressed with the following cuts.

- The di-lepton polar angle $\theta_{l^+l^-}$ reconstructed from their vector sum should fulfil $|cos\theta_{l^+l^-}| < 0.8$ - the continuum background originates from valence-quark sea-antiquark scattering, resulting in a boosted W^+W^- system
- The di-lepton opening angle ϕ should be between 0.2 and 1.0 radians - this exploits the spin correlation between the W^+W^- pair

5.3 Physics Analysis using ATLFAST++

The analysis was conducted using Pythia version 5.7 and the ATLFAST++ fast detector simulator within Root.
Signal was chosen as Higgs production via W or Z fusion with the leptonic decay restricted to an electron and a muon. All other subprocesses were switched off and the Higgs mass was set to 160 GeV. The primary tag was chosen as the jet with the highest P_T. The second tag was chosen in the opposite hemisphere as the jet furthest from the primary tag in pseudorapidity. Background processes were $pp \rightarrow t\bar{t}$ and $gg \rightarrow t\bar{t}$.
Generator and analysis code can be found at **http://www.hep.ucl.ac.uk/~ rms/phd.html**.
Plots were then generated which confirm these event topologies, and illustrate the possibilities for optimising the selection criteria.
Filled histograms correspond to the Higgs signal.

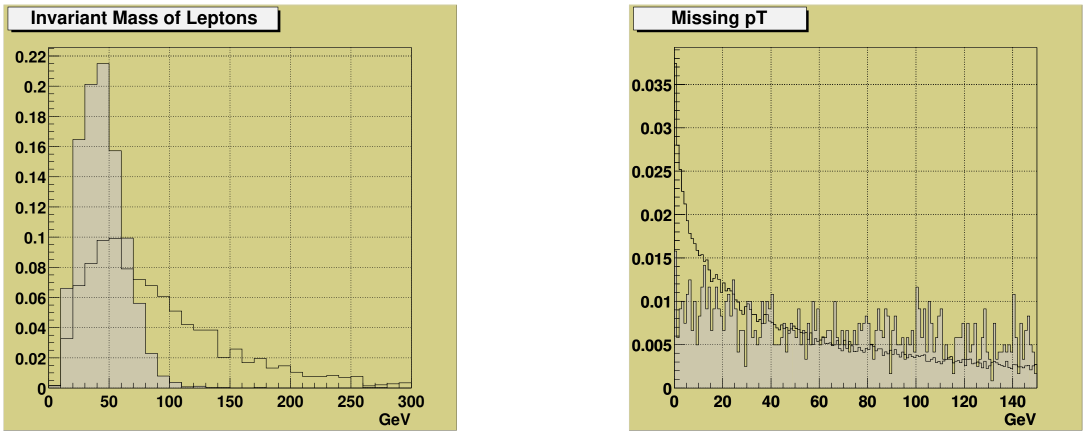

Figure 4: *Lepton Mass and Missing PT*

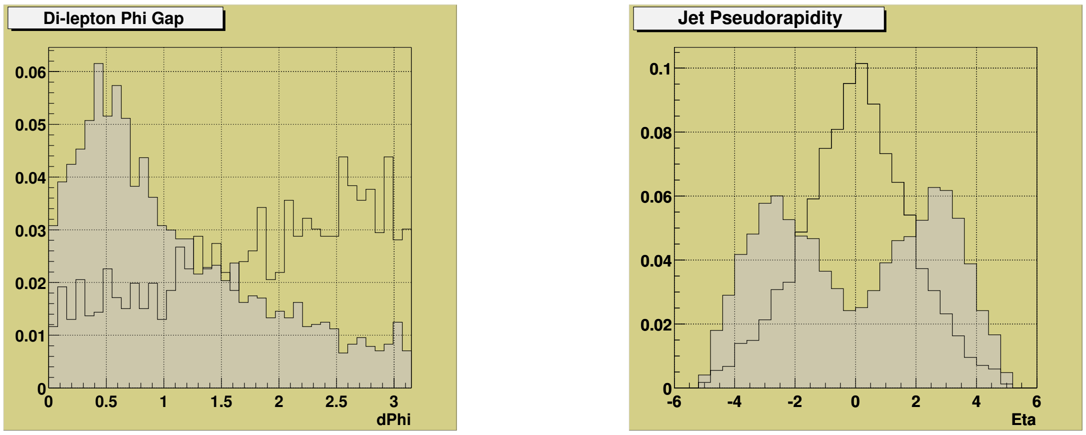

Figure 5: *Lepton Phi-Separation and Jet Pseudorapidity*

Figure 6 and 7 show further possibilities for background rejection. Signal efficiencies are in the process of being calculated for the selection criteria mentioned above and the quantities plotted below.

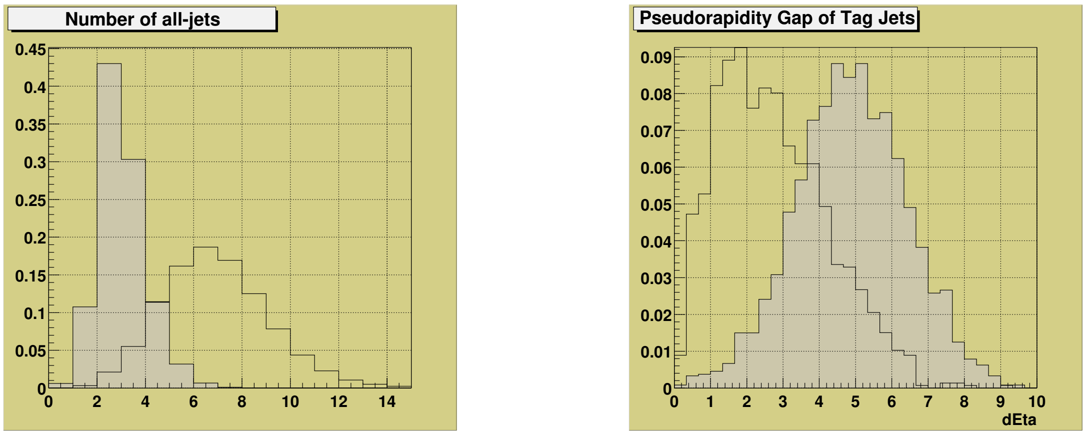

Figure 6: *Number of Jets and Jet Pseudorapidity Gap*

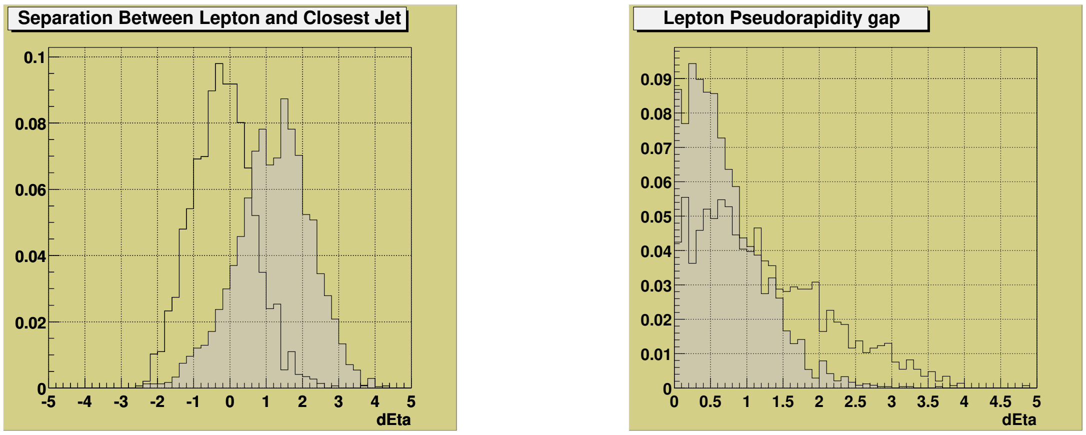

Figure 7: *Jet-Lepton Separation and Lepton Pseudorapidity*

References

[1] Gaudi User Guide: version 3/1 (1999); M.Cattaneo and P.Maley. LHC-b Software Architecture team, CERN, Geneva.

[2] Elzbieta Richter-Was. ATLFAST, http://root.cern.ch/root/Atlfast.html

[3] Tobjörn Sjöstrand. Pythia, http://www.thep.lu.se/˜ torbjorn/Pythia.html

[4] Rene Brun. ROOT, http://root.cern.ch/root

[5] UCL ATLAS Software Group: Peter Clarke, Hywel Phillips, Peter Sherwood, Richard Steward.

[6] Rainwater D and D Zeppenfield. Observing $H_0 \to WW \to e^{\pm}\nu\mu^{\pm}\nu$ in weak boson fusion with dual forward jet tagging at the CERN LHC. arXiv:hep-ph/9906218 (1999).

[7] Lee BW, Quigg C and Thacker. The Strength of Weak Interactions at Very High Energies and the Higgs Boson Mass. FERMILAB-Pub-77/22-THY (1977)

[8] Dawson S. The Standard Model Intermediate Mass Higgs Boson. arXiv:hep-ph/9703387 (1997).

[9] Matt Dobbs. HepMC, http://home.cern.ch/m/mdobbs/www/HepMC

[10] Dittmar M and Dreiner H. How to find a Higgs Boson with a Mass between 155-180 GeV at the LHC. arXiv:hep-ph/9608317 (1996).

[11] Jakobs K and Trefzger T. SM Higgs Searches for $H_0 \to WW \to e^{\pm}\nu\mu^{\pm}\nu$ with a Mass between 150-190 GeV at LHC, ATLAS Communications, ATL-PHYS-COM-99-031 (1999).