Cross section studies of the Z and neutral supersymmetric Higgs bosons decaying to tau leptons at CMS

Gordon Ball

High Energy Physics

Blackett Laboratory
Imperial College London

A thesis submitted to Imperial College London
for the degree of Doctor of Philosophy
and the Diploma of Imperial College

Abstract

This thesis contains a summary of work done while working with the Imperial College High Energy Physics group and other collaborators, using the Compact Muon Solenoid experiment located on the CERN Large Hadron Collider. Four areas are covered:

- Using 2010 CMS data (integrated luminosity = 36.0 pb^{-1}) to study decays in the $Z \rightarrow \tau\tau$ channel, covering the electron, muon and hadronic jet combinatorial final states. The cross section measured using the combination of these channels was $990 \pm 120 pb$, which is compatible with both theoretical predictions and measurements using the $Z \rightarrow ee$ and $Z \rightarrow \mu\mu$ processes.

- Using 2010 CMS data to set limits on the cross-section $\sigma(pp \rightarrow \Phi \rightarrow \tau\tau)$ of the MSSM neutral Higgs bosons, and hence constrain the $m_A \times tan\beta$ plane of this model. For a light ($m_A < 130$ GeV) Higgs, the space $\tan\beta > 30$ is excluded at the 95% confidence level.

- Development of workspaces for computing monitoring as part of the CMS Overview component.

- Development of the Data Aggregation System, a service designed to mediate and cache queries across multiple CMS information systems.

Declaration

This document presents work in the period 2007 – 2011. Chapter 6 and 7 describe my own analysis, which builds on other work by members of the Imperial College CMS Group and wider CMS Collaboration. Chapter 8 describes several monitoring environments, which are my own work, and the wider Overview framework, to which I contributed. Chapter 9 describes the entirety of the Data Aggregation System; the parsing, key-learning and analytics components, and the performance study are my own work, and I have contributed to other aspects of the system. The work of others has been referenced where used.

Gordon Ball
June 2011

Acknowledgements

To work at CERN on the Large Hadron Collider has been an ambition for as long as I knew it existed, and for the privilege of being able to do so I am very grateful to everyone who made it possible; those who taught me physics and inspired my interest in the subject, the Imperial College group and STFC for providing the opportunity, and the great many who have contributed to the field of High Energy Physics and ultimately the realisation of the LHC.

While at Imperial and CERN I have met many good friends, despite some of them working on other experiments. I will remember the good times for a long time to come.

Thanks are due to my supervisor, David Colling, and all the other members of the Imperial Tau group, for their assistance and expertise.

I would like to thank my family for their support during my PhD, and special thanks go to Claire for understanding during my prolonged sojourn in Geneva, and her support and forebearance during the late nights, frustrations, burnt dinners and just occasionally the euphoric moments of understanding this PhD has entailed.

Finally, I should also acknowledge Sir Tim Berners-Lee and Robert Caillau for an invention from which I have learned a great deal, some of it at least tangentially related to Particle Physics, and which nearly prevented this thesis from ever being finished.

Contents

List of figures

List of tables

“FOR SCIENCE”

Chapter 1

Introduction

"May you live in interesting times."
— Origin uncertain

The Large Hadron Collider project has, from inception to collisions, occupied a longer period than my own lifetime[1]. During the years $2007 - 2011$ covered by this thesis, the collider and associated experiments were completed, beams circulated, and despite various set-backs data-taking was begun in earnest. The LHC in general, and the CMS experiment in particular, provide an excellent opportunity to advance our state of knowledge about the nature of matter, whether by vindicating existing theories or dismissing them.

This document contains:

- Chapter 2 *"Background"*: The Standard Model and the role of the Higgs boson, the current status of searches for the Higgs and the Minimally Supersymmetric Standard Model.
- Chapter 3 *"The Large Hadron Collider"*: A description of the Large Hadron Collider.
- Chapter 4 *"The Compact Muon Solenoid Experiment"*: A description of the detectors making up the Compact Muon Solenoid experiment.
- Chapter 5 *"Reconstruction"*: The methods used to reconstruct physics objects in the CMS experiment, covering electrons, muons, hadronic tau jets and missing transverse energy, as will be used by the subsequent chapters.

- Chapter 6 *"Measuring the $Z \to \tau\tau$ production cross section"*: Using early data from the LHC to measure the cross section $\sigma(pp \to Z \to \tau\tau)$, using the decay channels $Z \to \tau\tau \to e\tau_{jet}$, $\mu\tau_{jet}$ and $e\mu$.
- Chapter 7 *"Limits for MSSM $\Phi \to \tau\tau$ "*: Limits are set on the light, neutral, MSSM Higgs boson production cross section, using the same decay channels, for masses in the range 90 GeV – 300 GeV. The cross section limits are used to set limits in the m_A, $tan\beta$ plane.
- Chapter 8 *"Computing Monitoring Pages"*: Describes the design and implementation of a monitoring system for CMS Monte-Carlo production.
- Chapter 9 *"Data Aggregation System"*: Describes the CMS Data Aggregation System.

The data used in Chapter 6 and Chapter 7 comes from Runs 136033 – 149442, which covers the *Run2010A* and *Run2010B* eras. The total integrated luminosity (during which all CMS systems were fully available) is 36.0 pb^{-1}.

Natural units are used throughout this document. Factors of c have been omitted from units for mass and momentum quantities.

Chapter 2

Background

The arguable motivation behind High Energy Physics is to develop a viable "theory of everything". The *Standard Model* (SM) developed during the 20^{th} century has proved a successful description of particles that have been observed thus far and their interactions. However, the Standard Model does not provide a complete prediction of all observed forces (gravity is conspicuously absent), nor have all Standard Model predictions been experimentally verified (the Higgs boson remains elusive), and contradictions such as neutrinos having non-zero masses have now been observed. This chapter contains a brief discussion of the Standard Model and supersymmetric theories which extend it, with a focus on the Electroweak and Higgs sectors that will be examined in Chapter 6 and 7.

2.1 The Standard Model

The "Standard Model"[1] is a description of all known fundamental particles and their interactions, synthesised from a wide variety of experimental and theoretical work. Three generations of quarks and leptons (spin $\frac{1}{2}$ fermions) are described, along with their associated antiparticles and the force-carrier particles of the electroweak and strong forces (spin 1 bosons).

It has proved a highly successful model, surviving verification by a wide variety of experiments[4]. The properties of a number of particles were correctly predicted by the Standard Model before their experimental observation; such as the W[5] and Z[6] bosons observed at CERN in 1983 for which the masses were predicted to better than 2% accuracy[7].

[1] Many full mathematical descriptions exist, eg [2, 3]; here we present a brief descriptive overview.

The Standard Model is described by a quantum field theory Lagrangian, which is invariant under local symmetry transformations of the gauge groups $U(1)_{EM}$, $SU(2)_{isospin}$ and $SU(3)_{colour}$[2]. Each of these groups represents one of the three forces, and the conserved quantities of each symmetry transformation are realised as quantum numbers of the particles. Examining local gauge invariance allows us to understand the interactions between these fields.

We will focus primarily on the $SU(2) \times U(1)$ weak and electromagnetic sector, which describes the interactions of the electroweak and Higgs boson(s) of interest, and set aside the largely independent $SU(3)$ strong sector.

The Dirac wavefunction of a fermion has invariant Lagrangian density under a number of global gauge transformations (such as translation or rotation), but not under gauge transformations where the transformation is a function of the spacetime position. The fermion wavefunction can be made invariant under a local $U(1)$ transformation by the addition of a vector field. The vector field is chosen to cancel out the extra terms from the fermion wavefunction under local gauge transformations.

The newly added field acts as a boson which couples to the fermion field, which we recognise in the context of the electromagnetic force as the photon. It is not possible to add a gauge invariant mass term for this field. The conserved quantity of this transformation is the coupling constant between the fermion and photon fields, which we recognise as charge.

In the same conceptual way that the addition of the photon field conserves $U(1)$ symmetry, the weak and strong forces are generated by local gauge invariance under the $SU(2)$ and $SU(3)$ groups. Unlike $U(1)$, these groups are non-Abelian, which has the consequence that the Lagrangian may also contain interaction terms between the gauge bosons (although they still cannot have mass). Each generator of the symmetry group[3] results in a gauge boson; three for the weak force (conserving weak isospin I and I_3) and eight for the strong force (conserving colour charges r, g and b).

Fermions can be expressed as a sum of left-handed and right-handed chiral terms. For massless particles, this is equivalent to helicity, the projection of spin along the particle's momentum vector. Left-handed particles transform as a doublet under $SU(2)$, and right handed particles as a singlet.

[2] $U(n)$ is the group of $n \times n$ unitary matrices, and $SU(n)$ the group of $n \times n$ unitary matrices with determinant 1.

[3] $SU(n)$ has $n^2 - 1$ generators.

However, the weak gauge bosons are experimentally determined to have mass. Massive bosons are incorporated into the weak force (while retaining local gauge invariance and renormalisability) by spontaneously breaking the gauge symmetry[8, 9, 10], discussed further in Section 2.1.1. The electromagnetic and weak interactions are mixed into a combined gauge group $SU(2) \times U(1)$. Two of the weak fields are mixed to produce the $W^{\pm}$ bosons, which only interact with left-handed fermions, and the remaining weak and electromagnetic field are mixed to produce the photon and the Z boson, which couple to both left and right-handed particles. The resulting gauge fields are:

$$\begin{aligned} W_{\mu}^{\pm} &= \frac{1}{\sqrt{2}}(A_{\mu}^{1} \pm A_{\mu}^{2}) \\ Z_{\mu}^{0} &= \frac{1}{\sqrt{g^2+q^2}}(qB_{\mu} - gA_{\mu}^{3}) \\ A_{\mu} &= \frac{1}{\sqrt{g^2+q^2}}(qB_{\mu} + gA_{\mu}^{3}) \end{aligned}$$

where q and g are the electromagnetic and weak coupling constants, B_{μ}, $A_{\mu}^{1..3}$ the electromagnetic and weak gauge fields and $\frac{g}{\sqrt{g^2+q^2}} = \sin\theta_W$, the weak mixing angle.

2.1.1 The Higgs Mechanism

Proposed by Higgs and others[11, 12], the Higgs mechanism explains the existence of massive electroweak gauge bosons. A scalar $SU(2)$ field, Φ, is introduced

$$\Phi = \begin{pmatrix} \phi^{+} \\ \phi^{0} \end{pmatrix}$$

with a potential

$$V(\Phi) = \frac{\mu^2}{2}\Phi^{*}\Phi + \frac{\lambda}{4}(\Phi^{*}\Phi)^2$$

where μ is the Higgs mass parameter and λ is the Higgs self-interaction. The Vacuum Expectation Value (VEV) of a field is the minimum of $V(\Phi)$. In the case $\mu^2 < 0$, this potential has a continuum of Φ for which the VEV is a minimum, rather than a single point. The complete set of all possible vacuum states preserves $SU(2) \times U(1)$ symmetry, but the choice of any individual one does not, hence spontaneous symmetry breaking. The vacuum states occur at $|\Phi\Phi^\dagger| = \sqrt{\frac{2\mu^2}{\lambda}}$. We can choose the ground state to be

$$\langle\Phi\rangle_0 = \left(\begin{array}{c} 0 \\ \frac{v}{\sqrt{2}} \end{array} \right)$$

where $v = \frac{\mu}{\sqrt{\lambda}}$. This choice ensures that the $U(1)$ (ie, electromagnetic) charge of the vacuum is zero. Goldstone's Theorem[13] states that each spontaneously broken symmetry generator results in a massless scalar boson. By expanding the Higgs potential close to the VEV we find one massive scalar boson (the Higgs) and three massless Goldstone bosons.

By carefully choosing the gauge when expanding the Higgs potential, the Goldstone bosons are "eaten" by the electroweak gauge bosons, producing their masses

$$m_{W^\pm} = \frac{gv}{2}, m_Z = \frac{gv}{2\cos\theta_W}, m_\gamma = 0$$

The Higgs gives mass to all fermions by the Yukawa coupling between the scalar and fermion fields. The coupling has strength proportional to the Higgs VEV, producing a mass term $m_f = \frac{g_f v}{\sqrt{2}}$. The coupling constant g_f is not predicted and hence becomes a free parameter equivalent to the fermion mass.

The base Higgs mass is $m_{H_0} = \sqrt{-2\mu^2}$, however this receives corrections from loop diagrams with all fermions and other bosons. Since the strength of the coupling is proportional to the particle mass, this contribution is dominated by the heaviest particles. The correction to the Higgs mass, Δm_H^2, from fermion loops is approximately the square of whatever cut-off energy scale (Λ) is chosen, between the current experimental exclusion of new physics and the Planck scale.

To obtain a physical Higgs mass near the masses of other particles either requires a very low cut-off scale, very large base mass or some mechanism for systematically cancelling the fermion loop contributions.

In addition to the exclusions obtained from direct searches for the Higgs, the mass can be constrained by theory. Since it interacts with the electroweak sector, precision measurements of electroweak gauge bosons can be used to fit the most likely Higgs mass. Figure 2.1 shows the results of fitting the top, W and Higgs mass, and Figure 2.2

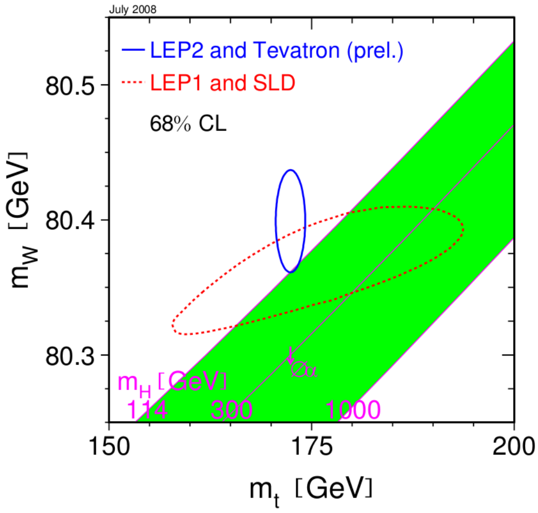


Figure 2.1: Best fits for the masses of the top quark and W boson, showing limited overlap at the 68% CL between the fitted area and the Higgs-compatible areas. [14]

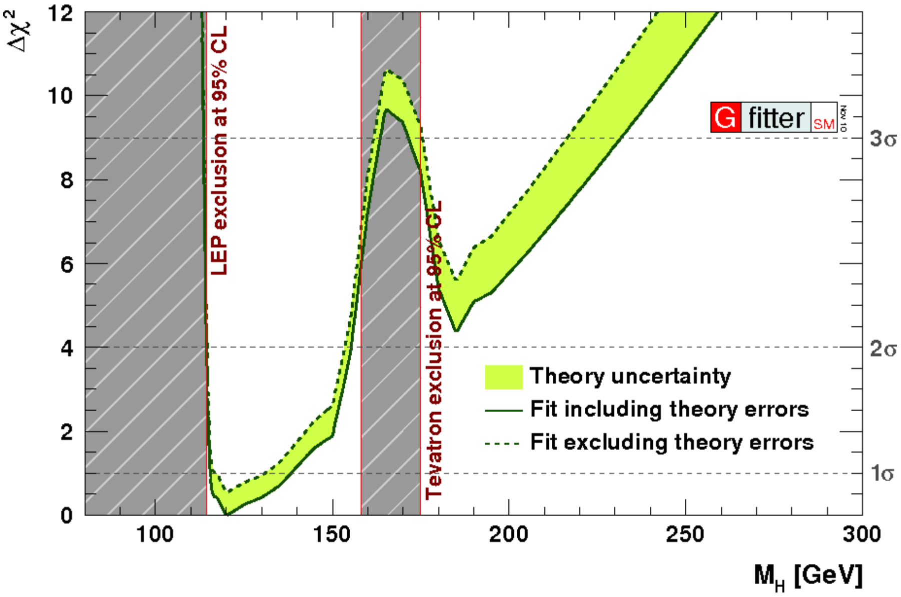


Figure 2.2: Exclusion limits on the Standard Model Higgs as of November 2010, and the χ^2 fit to the Higgs mass including both precision electroweak and direct search constraints. [15]

shows the current experimentally-excluded ranges and the χ^2 fit of the Higgs mass in the remaining regions.

An upper limit on the Higgs mass may be obtained from the requirement that cross sections remain physical. The Higgs self-interaction parameter λ is given by

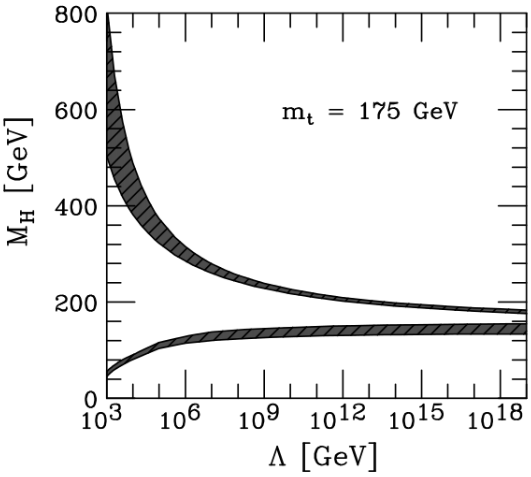


Figure 2.3: Theoretical upper and lower limits on the Standard Model Higgs mass, as a function of the cut-off energy scale. [16]

$$\lambda = \frac{m_H^2}{2v^2} = \frac{G_F m_H^2}{\sqrt{2}}$$

The value of the fermi coupling G_F runs with the energy scale (as higher order processes become important), and hence λ varies as a function of the cut-off energy scale. For very large values of λ, cross sections for Higgs decays to weak bosons become divergent, and for very small values coupling to heavy fermions (ie, the top quark) become negative. The constraint on m_H is a function of the cut-off energy scale, but the upper limit on m_H is $\approx$ 800 GeV. The upper and lower limits are shown in Figure 2.3. With sufficient integrated luminosity, the LHC can probe (and, if necessary, exclude) the entire permitted region for the Standard Model Higgs.

2.1.2 Problems with the Standard Model

Despite the compelling agreement between the Standard Model and experimental results, a number of issues remain unresolved:

- No description of the gravitational force is provided, which is clearly unsatisying for any unified theory.
- Non-zero neutrino masses[17] are not accommodated by the Standard Model.

- Even without neutrino masses, the model contains 18 free parameters[4] which have had to be experimentally determined. This gives an appearance of inelegance.
- Observations from cosmology are not adequately explained; no explanation exists for dark matter (no weakly interacting massive particle (WIMP) exists in the standard model), and it is clear that observed CP violation is not sufficient to explain the remaining amount of baryonic matter in the universe.
- Extrapolating the EM, weak and strong coupling constants to very high energy scales does not yield a common point of intersection, which might be expected if they could be described by a single gauge group and coupling.
- The hierarchy problem; loop corrections to the Higgs mass are divergent, and should result in a mass close to the Planck scale without extremely fine-tuning to counter them.

Supersymmetric theories offer solutions to some of these problems.

2.2 Supersymmetry

A possible solution to the Higgs fine-tuning problem is to find a way of cancelling the fermion loop corrections to the Higgs mass. Fundamentally, supersymmetric[18] models introduce an additional symmetry between fermions and bosons, with a supersymmetry (SUSY) operator

$$Q\left|fermion\right\rangle \rightarrow \left|boson\right\rangle, Q\left|boson\right\rangle \rightarrow \left|fermion\right\rangle$$

Each fermionic or bosonic field in the Standard Model gains a superpartner. The conserved charge of the supersymmetric operator is $R = (-1)^{2s+3B+L}$[5], where all Standard Model particles have $R = 1$ and superpartners $R = -1$.

At the most basic level, superpartners should have similar properties and behaviour to their Standard Model counterparts. The fact that no SUSY particles have been observed with mass equal to their partners implies that supersymmetry (if it exists) is a broken symmetry.

[4] Masses of $u, d, c, s, t, b, e, \mu, \tau, Z, H$; EM, weak and strong couplings α, G_F, α_S; three CKM mixing angles and CP violating phase.

[5] Where s is spin, B baryon number and L lepton number.

Supersymmetry solves (to some extent) three of the problems with the Standard Model. The addition of superpartners provides a solution to the hierarchy problem by cancelling each fermion loop with a boson loop of opposite sign, eliminating divergent corrections to the Higgs mass. The running of coupling constants under supersymmetry is such that electromagnetic, weak and strong coupling constants converge at a common point[19]. Finally, if R is a conserved quantity it implies that the lightest supersymmetric particle (if neutral) must be stable, massive and weakly-interacting, which makes it a good dark matter candidate[20].

However, supersymmetry per se introduces a vast parameter space, with $O(100)$ free parameters compared to the Standard Model. We will focus on a supersymmetric model which contains the minimum extra behaviour necessary to solve these Standard Model problems.

2.2.1 The Minimally Supersymmetric Standard Model

The Minimally Supersymmetric Standard Model (MSSM) is the simplest SUSY theory that solves the hierarchy problem while avoiding phenomenological difficulties such as fast proton decay or extensive CP violation in the lepton sector. We will limit the discussion here to the MSSM Higgs sector. For a fuller discussion of the model, see [21].

While the majority of particles map directly to a superpartner in the MSSM, the Higgs sector requires two $SU(2)$ doublets, Φ_u, Φ_d, which couple to up- and down-type quarks respectively, each with an independent VEV (v_u, v_d). Similarly to the Standard Model, applying $SU(2) \times U(1)$ symmetry breaking to each of these doublets produces four scalar fields. Three of these eight are eaten by the electroweak gauge bosons, and the remaining five form the MSSM Higgs spectrum.

The mass of the W bosons is given by

$$m_W^2 = \frac{g^2}{2}(v_u^2 + v_d^2)$$

which allows us to constrain v_u and v_d as a single parameter

$$\tan\beta = \frac{v_d}{v_u}$$

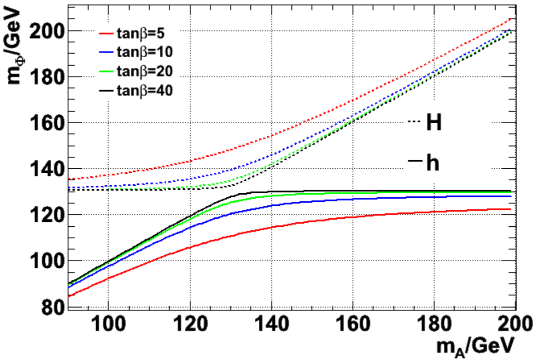


Figure 2.4: Masses of the h and H supersymmetric Higgses as a function of m_A, for several values of $\tan\beta$. The A can be seen to be degenerate with the h below $\approx$ 130 GeV and with the H above.

The Higgs sector contains two charged Higgs, $H^{\pm}$, an CP-odd neutral Higgs, A and two CP-even neutral Higgses, h and H. The masses of these can be expressed (at leading order) entirely as a function of the mixing angle β and a single mass parameter, m_A:

$$m^2_{H,h} = \frac{1}{2}(m^2_A + m^2_Z \pm \sqrt{(m_A + m_Z)^2 - 4m^2_Z m^2_A \cos 2\beta})$$

$$m^2_{H^{\pm}} = m^2_A + m^2_W$$

This implies a constraint on the upper mass of the h, $m_h < m_Z|\cos 2\beta|$, although this is modified by higher-order corrections. The masses of the neutral Higgses as a function of $\tan\beta$ are shown in Figure 2.4.

For large values of $\tan\beta$, $\cos 2\beta \to 1$, and hence the mass spectrum becomes $m_A \approx m_h, m_H \approx m_{hmax}$ for $m_A < m_{hmax}$ and $m_A \approx m_H, m_h \approx m_{hmax}$ for $m_A > m_{hmax}$ respectively. Including loop corrections, $m_{hmax} \approx 130$ GeV. The coupling to down-type fermions is also enhanced by a factor of $\tan^2\beta$, meaning that associated production with bottom quarks and decays to bottom quarks and tau leptons are signficantly enhanced compared to the Standard Model Higgs.

2.3 Particle background

With the focus on $Z \to \tau\tau$ processes in Chapter 6, it will be useful to provide a brief digression on the properties and history of these particles.

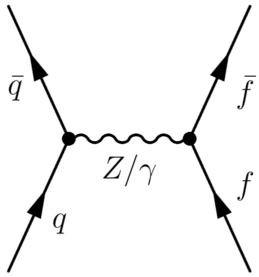


Figure 2.5: Drell-Yan process

2.3.1 Z boson

The “neutral current” was first observed indirectly (by the appearance of isolated vertices yielding only hadrons) in 1974 in the Gargamelle bubble chamber[22]. It was subsequently observed directly (by the invariant mass of lepton pairs) in the UA1 and UA2 experiments on the $Sp\bar{p}S$ in 1983[6], along with the W bosons. Precision measurements have since been performed by experiments on the LEP and Tevatron colliders[4].

In a proton-proton collider, the leading Z boson production mechanism is the Drell-Yan[23] process, shown in Figure 2.5.

Unlike in $p\bar{p}$ collisions, the antiquark must be a sea rather than valence quark, causing a corresponding reduction in cross section.

The Z decays to $f\bar{f}$[6] pairs. Since $m_Z = 91.1876$ GeV $\gg m_f$[24] for all known quarks and leptons except the top quark, the branching ratio and kinematics of decays to fermion-antifermion pairs of each species are approximately the same (accounting for a colour factor in decays to quark pairs). The decay fractions $Z \to ee, \mu\mu, \tau\tau$ are each 3.36%. The Z has a very narrow width (2.4952 GeV), producing a characteristic visible mass peak in the di-lepton spectra.

The LEP collider was initially operated with the centre of mass energy tuned for Z production. However, proton structure makes such tuning impossible in a hadron collider, so the $q\bar{q} \to l^+l^-$ spectrum includes a signficant contribution from γ interference. The $Z \to f\bar{f}$ cross section (Figure 2.6) is defined as $\sigma(pp \to f\bar{f})$ where $60 < m_{f\bar{f}} < 120$ GeV, the region in which the Z contribution is dominant.

[6] Where $f \to$ any fermion.

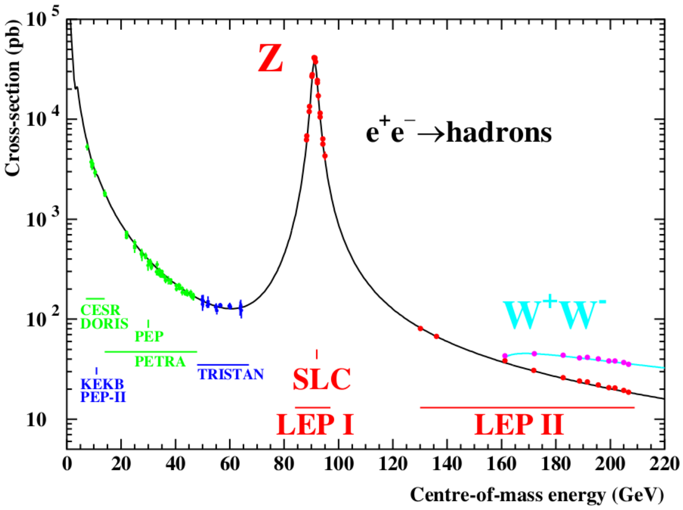


Figure 2.6: Cross section $\sigma(e^+e^- \to q\bar{q})$ as a function of centre of mass energy, showing the region $\sqrt{s} \in [60, 120]$ in which the Z dominates. This is the inverse process to $q\bar{q} \to Z \to ll$ of interest at the LHC, but the cross section profile is the same. [4]

2.3.2 Tau lepton

The third-generation tau lepton was first observed in 1974 with the process $e^+e^- \to e^\pm\mu^\mp$ in 4.8 GeV collisions with the SPEAR[7] ring at SLAC[25].

The mass is 1.777 GeV[24] and the average lifetime is 291 fs, making direct observation in collider experiments difficult. Tau processes must be deduced from their decay products, which is complicated by final states that always contain one or more neutrinos.

Taus decay by W emission, producing a tau neutrino and either an electron or muon (and associated neutrino) or one or more hadrons. The leptonic decay fractions are 17.8% and 17.4% respectively, for the electron and muon. Of the hadronic modes, 76.4% (49.5% of all tau decays) involve a single charged pion or kaon ("one-prong"), 23.5% (15.2% of all tau decays) three charged hadrons, and $< 0.1\%$ five or more.

[7] **S**tanford **P**ositron **E**lectron **A**ccelerating **R**ing

Chapter 3

The Large Hadron Collider

3.1 Introduction

The Large Hadron Collider (LHC) is a 26.7 km circular particle accelerator, constructed in the former LEP[1] tunnel (see Figure 3.1) underneath the Franco-Swiss border near

[1] **L**arge **E**lectron **P**ositron Collider - see Appendix A.3

Figure 3.1: Interior of the 3.8 m diameter LHC tunnel between the ATLAS and ALICE experiments. [26]

Geneva. The design goal was for proton-proton collisions at energies of 7 TeV per beam and heavy-ion collisions (using lead ions) at 2.7 TeV per nucleon.

Planning for the LHC and associated experiments was started in 1984[1], with construction beginning in 2001 following the dismantling of the LEP. Particles were first injected into the LHC in September 2008 but, following a serious magnet quench accident at the end of that month (see Appendix A.4), further operation was delayed for a year while repairs were carried out and additional instrumentation and safety equipment installed.

The LHC was re-started in November 2009, initially at the injection energy of 450 GeV per beam. This was quickly followed by acceleration to 1.18 TeV per beam, at which point the LHC surpassed the Tevatron as the highest-energy operational accelerator.

The beam energy was ramped up in 2010, with the first collisions at 3.5 TeV per beam achieved in March 2010.

Four major experiments are constructed on the LHC ring:

- **ALICE**[2], studies quark-gluon plasmas resulting from heavy ion collisions
- **ATLAS**[3], a general-purpose detector
- **CMS**[4], a general-purpose detector
- **LHCb**[5], studies B-meson physics

In addition, the TOTEM[6], LHCf[7] and MoEDAL[8] experiments are located near the interaction points used by the above experiments.

3.2 Design

Unlike the contemporary (but never completed) SSC[9], the LHC was designed with the constraints of the existing LEP tunnel, limiting the beam energy by the magnetic

[2] **A** **L**arge **I**on **C**ollider **E**xperiment

[3] **A** **T**oroidal **L**HC **A**pparatu**S**

[4] **C**ompact **M**uon **S**olenoid

[5] **LHC** **b**eauty experiment

[6] **TOT**al **E**lastic and diffractive cross section **M**easurement

[7] **LHC** **f**orward

[8] **Mo**nopole and **E**xotics **D**etector **A**t the **L**HC

[9] **S**uperconducting **S**uper-**C**ollider

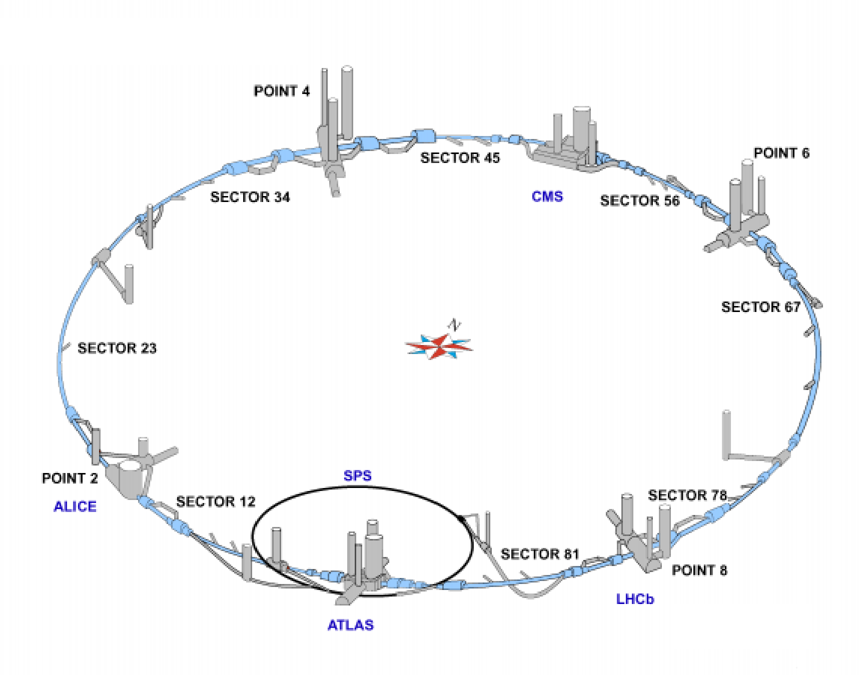


Figure 3.2: Diagram of the LHC and SPS underground complexes. The main CERN site is adjacent to the ATLAS cavern at Point 1. The LEP experiments were located at Points 2, 4, 6 and 8, two of which have been reused for ALICE and LHCb. ATLAS and CMS are located in newly constructed caverns. Points 4 and 6 house the RF cavities and beam dump, respectively. Adapted from [29].

field required to produce the fixed radius. The design called for the use of NbTi superconducting magnets, a proven technology already utilised by the Tevatron[27] and HERA[28] accelerators, although the LHC uses superfluid instead of liquid helium to allow a lower operating temperature (1.9 vs 4.2 K) and a stronger magnetic field (8 vs 4 T). This configuration allowed for 7 TeV per beam and $\sqrt{s} = 14$ TeV.

The LHC consists of two counter-rotating proton beams, circulating 19 cm apart in common magnetic and cryostat assemblies. The ring is divided into octants, four of which contain interaction regions around which experiments are built. The remainder contain two beam-cleaning assemblies, the accelerator RF cavities and the beam dump respectively. The installation is shown in Figure 3.2.

Each octant consists of a 528 m straight section (containing the interaction region or other facility) and a 2460 m arc. The arc consists of 23 magnetic cells, each containing six dipole and two quadrupole magnet assemblies.

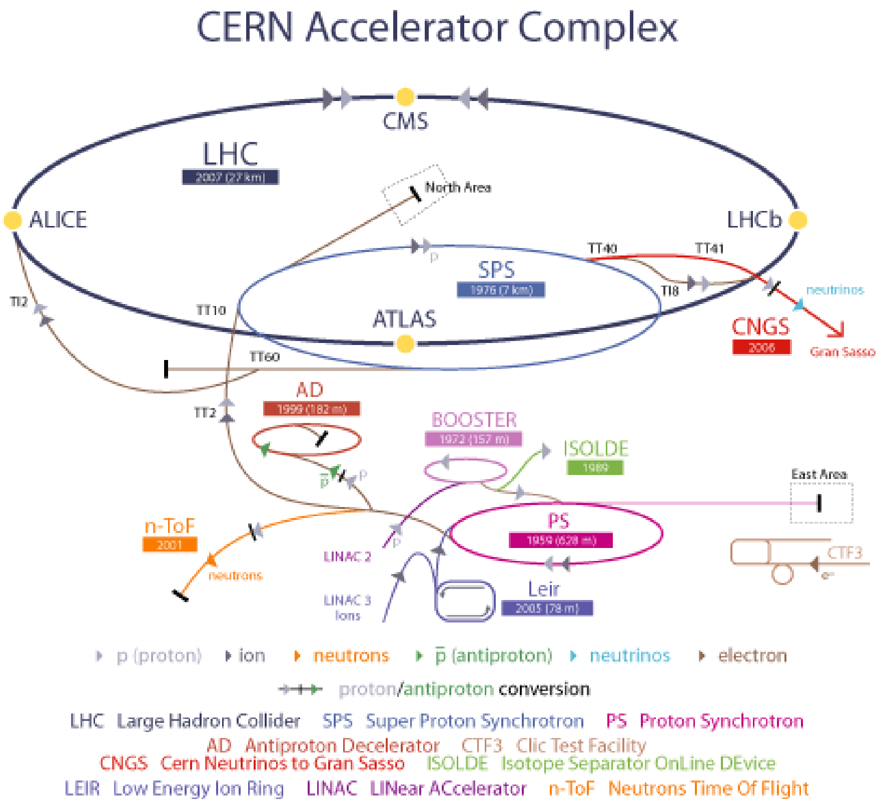


Figure 3.3: Diagram of the CERN accelerator complex, showing the complete injection chain for the LHC, as well as those used by other experiments[30]

The interaction region is 130 m between the separation dipoles. Due to the small separation between bunches (nominally 7.5 m), to avoid secondary collisions in the interaction region a crossing angle of 200 µrad is required.

A large part of the CERN accelerator complex is required to inject protons into the LHC, as shown in Figure 3.3. Protons for the LHC are first produced from gaseous hydrogen by a duoplasmatron, and are accelerated to 50 MeV by the Linac2 linear accelerator. These pass through a transfer line to the Proton Synchrotron Booster (PSB), a 25 m radius ring which accelerates the bunches to 1.4 GeV ready for injection into the Proton Synchrotron (PS). Ion beams for *Pb* collisions from the Low Energy Ion Ring (LEIR[10]) enter the LHC injection chain at this stage from a separate linear accelerator. The PS accelerates the beam to 26 GeV and is used to produce the correct bunch structure for the rest of the chain. From the PS bunches pass into the Super Proton Synchrotron (SPS), with three PS fills required to fill the SPS before the beam is accelerated to 450 GeV. Finally, the beam is extracted into one of the two LHC rings.

[10] Built from the Low Energy Antiproton Ring (LEAR)

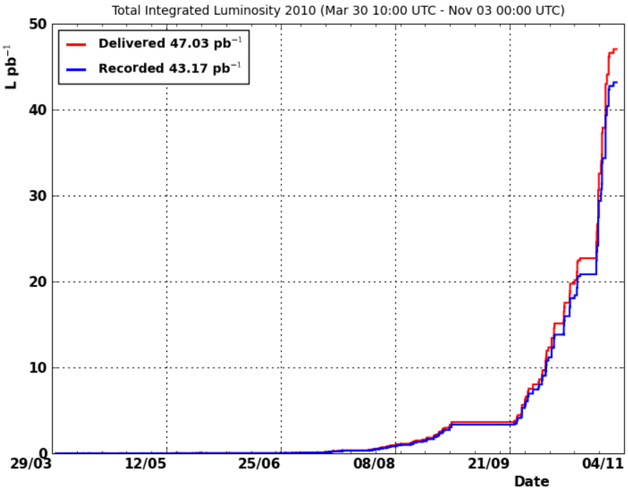


Figure 3.4: Graph of proton-proton integrated luminosity (seen at the CMS interaction point) for 2010[31].

The bunch structure in the LHC contains a series of gaps in the bunch train left by the rise time of the various kicker magnets used for extraction. To complete the nominal LHC fill of 2808 bunches per beam, 24 fills of the SPS and 72 of the PS are required. The complete LHC fill procedure and acceleration to collision energy takes about 30 minutes.

The design bunch size is 1.15×10^{11} protons[11], which under optimal emittance (3.75 µm), $\beta*$ (0.55 m) and bunch spacing (25 ns) conditions would result in approximately 20 proton-proton interactions per crossing and an instantaneous luminosity $L = 10^{34}$ cm^{-2} s^{-1}. This is expected to ultimately allow collection of O(100) fb^{-1}/year.

Collisions at 3.5 TeV per beam were taken during the latter half of 2010, with a total integrated luminosity (in CMS) of 45 pb^{-1} (see Figure 3.4). Increases in the size and number of bunches, and improvements to the focus in the interaction regions meant the majority of the data was taken within the last few weeks of the proton-proton physics run, with about $\frac{1}{8}^{th}$ of the recorded data taken in a single 14 hour fill. By the end of 2010 the beams consisted of approximately 400 bunches with a 50 ns spacing, with an instantaneous luminosity of $L = 10^{32}$ cm^{-2} s^{-1}.

[11] This has been increased to 1.5×10^{11} and may be increased further.

Chapter 4

The Compact Muon Solenoid Experiment

4.1 Introduction

The Compact Muon Solenoid (CMS)[32] is one of the two general-purpose particle detectors located on the LHC ring, located near the village of Cessy[1] in a cavern (Figure 4.1) 85 m underground.

4.2 Design

The original CMS proposal[33] called for a detector able to measure 100 GeV muons, electrons and photons with 1% precision. This was motivated by the primary purpose of searching for the Higgs boson in the 100 GeV - 1 TeV mass range, and the capability of distinguishing these events from a background of up to 10^9 inelastic collisions per second.

The broader goals of the CMS experiment are discovery or measurement of:

- The Higgs Boson
- Heavy Vector Bosons
- Supersymmetric Particles
- Precision Standard Model

[1] $46°18'34''N$, $6°4'37''E$

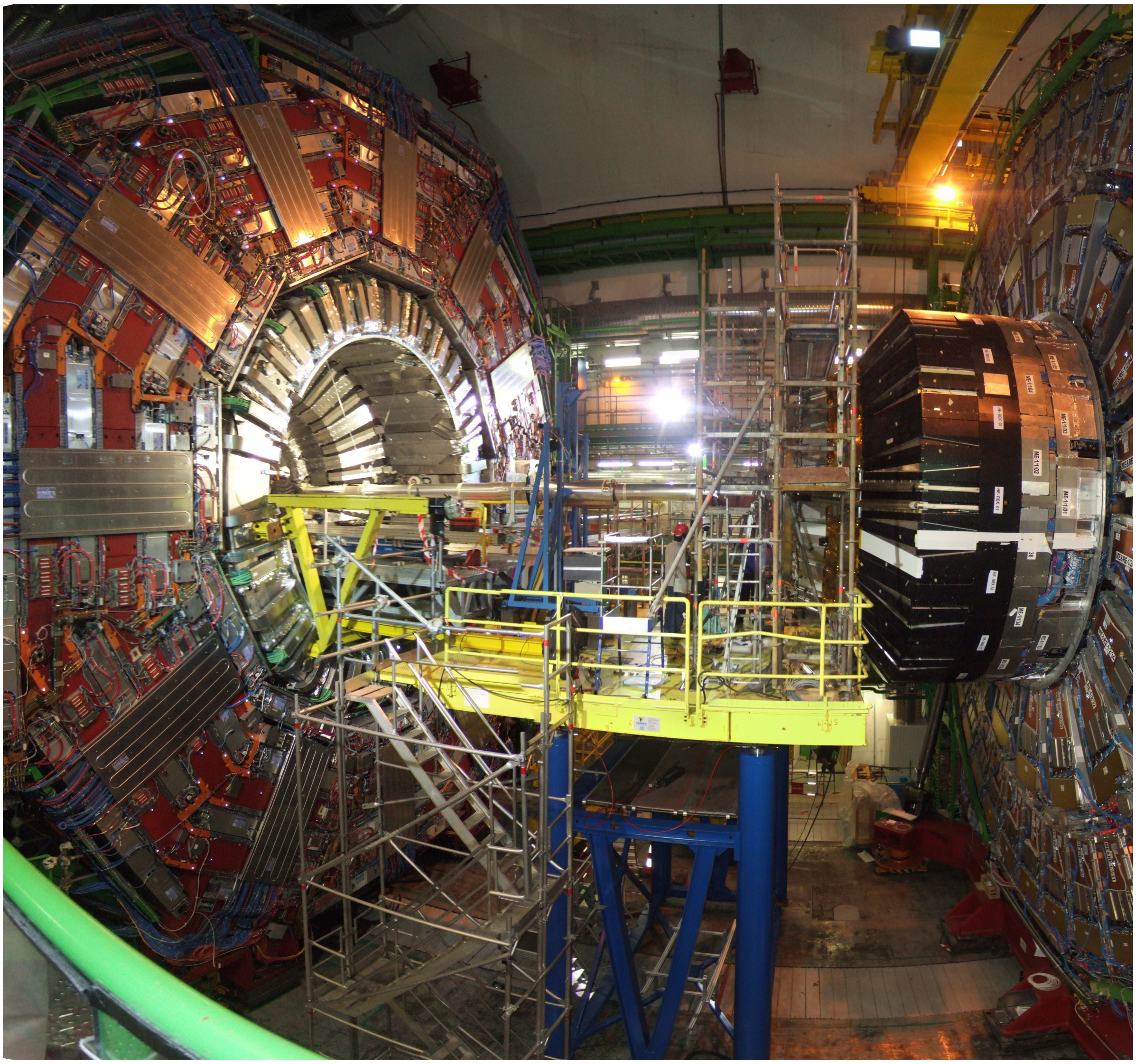


Figure 4.1: CMS opened for maintainance during 2009. The endcap (right) has been moved away from the barrel (left) to allow access to the centre, with the beam-pipe suspended in between. The total radius is 7 metres, and the radius inside the solenoid 3 metres.

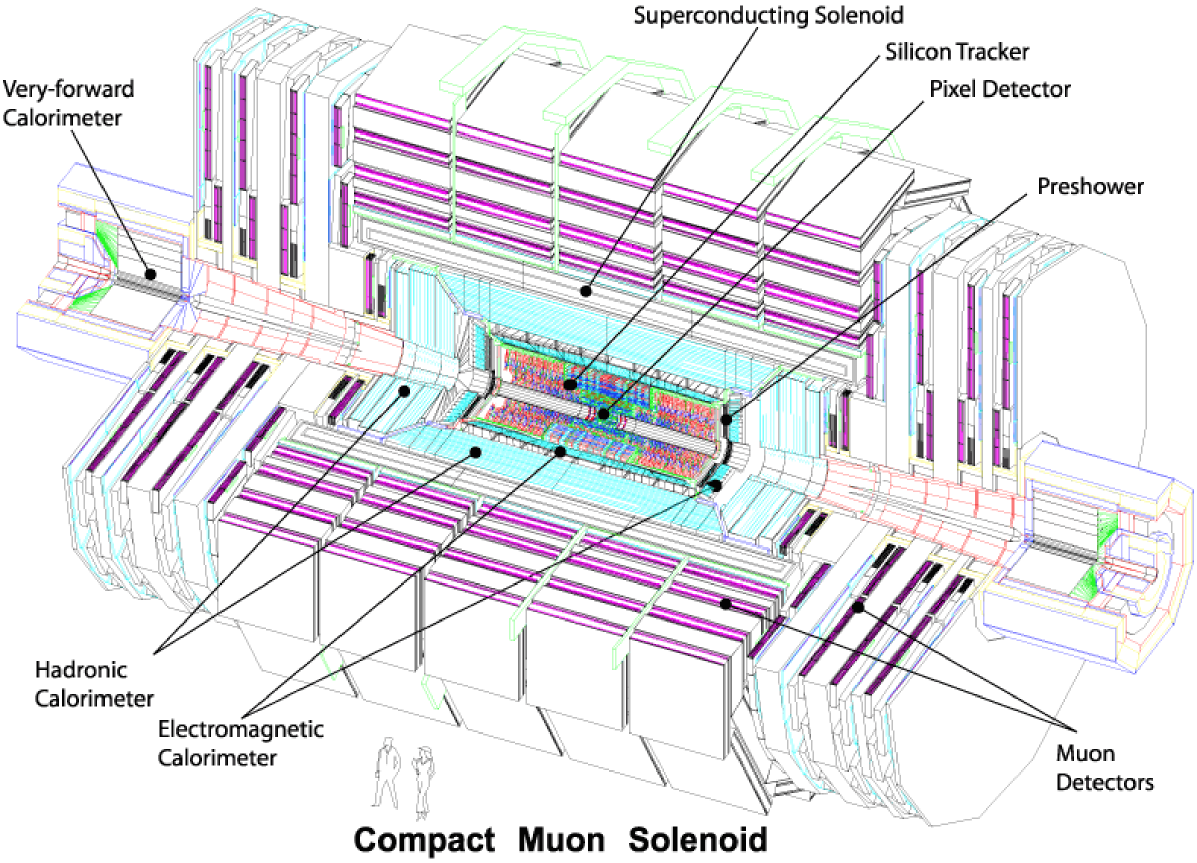


Figure 4.2: Cutaway diagram of the CMS detector, with human outlines for scale. This shows the “onion-skin” layering of the detector components from the pixel tracker around the interaction point to the muon detector at the outside. [34].

The resulting detector is shown in Figure 4.2. CMS is a conventional "4π" hermetic design, with full coverage within $|\eta| < 2.5$ and some coverage to $|\eta| < 5$. The detector is 21.6 m long, 14.6 m in diameter and has a completed mass of $12,500$ tonnes.

The design was constrained by the need to construct it wholly on the surface, with limited underground assembly time possible before nominal LHC start-up since the Point 5 cavern had to be enlarged during the same era as CMS construction.

4.2.1 Coordinate System

CMS uses a right-handed coordinate system, with z along the beamline, pointing (anti-clockwise from above) towards ALICE, x in the horizontal plane pointing towards the centre of the LHC ring and y in the vertical plane pointing towards the surface.

The azimuthal angle ϕ is measured in the $x - y$ plane starting from the $+x$ axis, and the polar angle θ is measured from the z axis. The pseudorapidity η is given by $-\ln(\tan\frac{\theta}{2})$.

4.3 Detectors

The CMS detectors are built around a 13 m long, 6 m diameter superconducting solenoid with a magnetic field of up to 4 T. The high field allows the compact design of the rest of the detector, with all subdetectors except for parts of the hadronic calorimeter and the muon detector built inside the magnet volume. The large length to radius ratio of the magnet provides a uniform magnetic field in the main η region, removing the need for additional magnet systems at high η to provide charge identification. The magnet current is 19.5 kA at 3.8 T[2], with a total stored energy of 2.5 GJ.

4.3.1 Tracking

At full luminosity, approximately 20 inelastic collisions are expected per bunch crossing, with an average yield of 1000 charged particles. The tracker needs to be able to efficiently reconstruct tracks at this expected occupancy, in order that both the primary vertices of

[2] The magnet was originally designed for 4 T, but the lower field strength was deemed more prudent for prolonged operation.

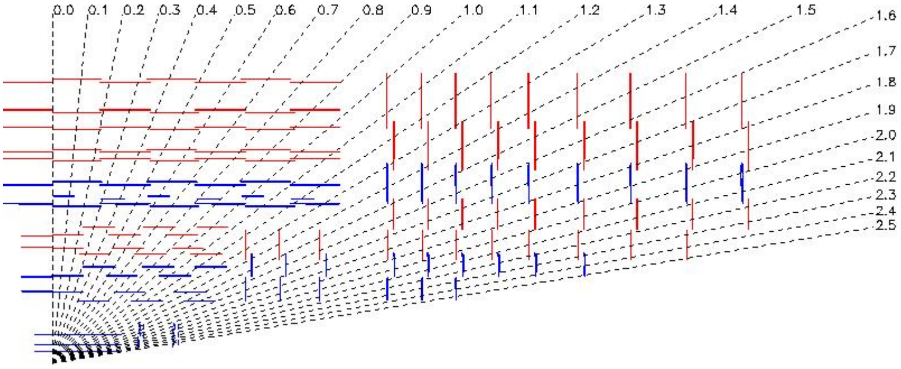


Figure 4.3: Quarter-view of the CMS inner tracking system, showing the positions of the pixel (inner three radial/two longitudinal) and strip layers, as a function of η. [34].

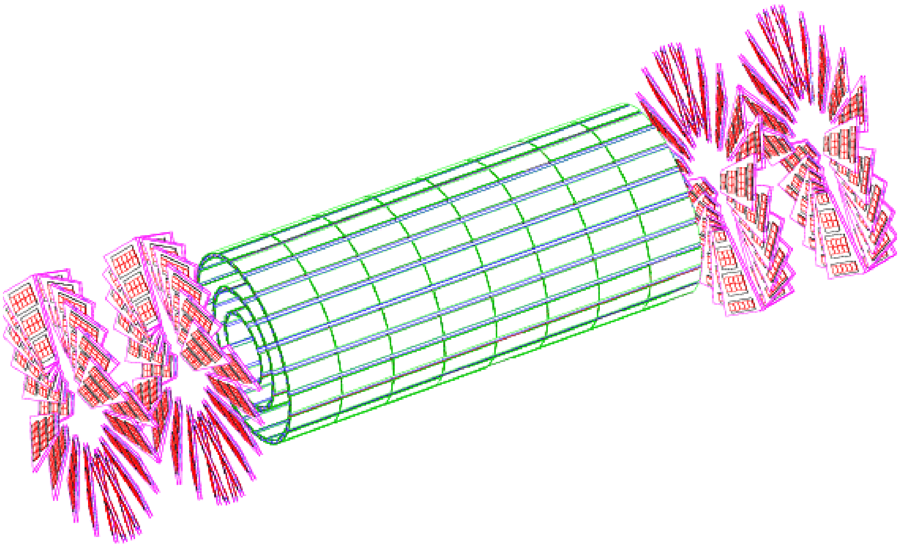


Figure 4.4: 3D render of the CMS pixel detector, showing the three inner barrel layers ($r < 102$ mm) and the geometry used for the two pixel endcap wheels. [34].

tracks can be identified to distinguish superimposed events, and that secondary vertices (if any) can be located. This requires a design sufficiently segmented that occupancy remains low ($< 3\%$), and hit position resolution of < 50 µm to unambiguously identify the charge of charged particles with $p_T = 1$ TeV.

The tracker (full description in [35]) is a fully-silicon design, consisting of layers between 320 and 500 µm thick mounted on carbon-fibre support structures, instrumenting the region $|\eta| < 2.5$. The high magnetic field ensures that most low p_T tracks will spiral in the traverse plane, and therefore that occupancies in the outer layers decrease faster than $\frac{1}{r^2}$, which reduces the segmentation required. Two different major geometries are used, approximately square pixels in the inner layers and rectangular strips in the outer layers.

The pixel detector (see Figure 4.4) consists of three inner layers, at radii of 44, 73 and 102 mm from the interaction point. There are also two endcap wheels in each direction, located at $|z| = 345$ and 465 mm to extend pixel coverage to $|\eta| < 2.2$. The pixels are 100×150 µm, giving a total of 66 million channels, with an average occupancy of 0.01% in the full luminosity scenario. The total active area of the pixel detector is 1 m^2. Being very close to the interaction point, the pixel detector exists in a harsh radiation environment and is expected to have a useful lifetime of approximately six years. As the semiconductor is damaged, charge leakage increases and charge yield from hits decreases, reducing signal-to-noise ratios and requiring larger voltages (and hence more heat) to operate.

The strip tracker occupies the remaining tracker volume (see Figure 4.3) out to a radius of 108 cm and $|z| = 280$ cm. There are eleven strip layers in the barrel, with the first four ($r < 55$ cm) using 10 cm × 80 µm strips and the remaining seven using 25 cm × 180 µm. Of these, the first and second, and fifth and sixth layers are arranged with a stereo angle of 100 mrad to provide a measurement in both the $r - \phi$ and $r - z$ directions. The ends of the inner strip layers are filled by three rings (with $|z| < 120$ cm) and the remainder of the endcap with a further nine rings, using the coarser strip geometry of the outer barrel layers. The strip layers in total encompass 9.6 million channels, with a surface area of 200 m^2.

At nominal luminosity, for tracks with 10 GeV $< p_T <$ 100 GeV and $|\eta| < 1.5$, the tracker achieves approximately 95% reconstruction efficiency. Momentum resolution $\frac{\sigma_p}{p}$ is 1.5% and radial and longitudinal vertex resolutions are 20 µm and 100 µm respectively.

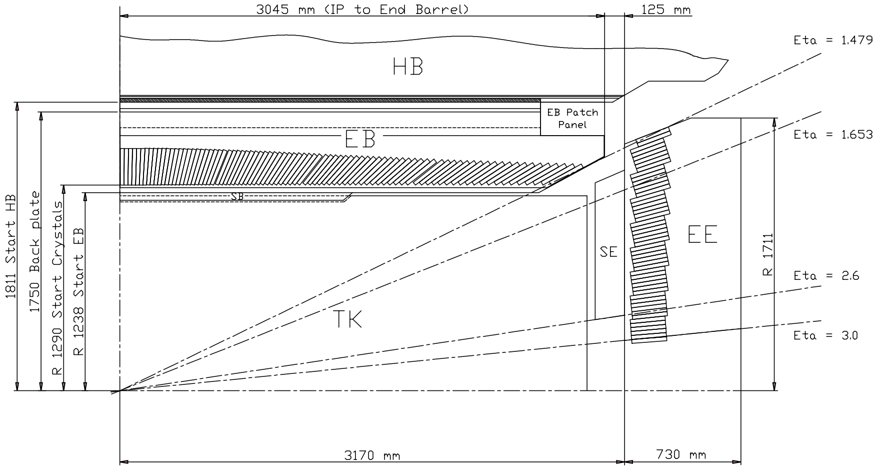


Figure 4.5: Quarter-view of the CMS ECAL geometry, showing the coverage of the barrel (EB) and endcap (EE) as a function of η. Note the crossover region through which tracker cooling and electronics are routed at $1.479 < |\eta| < 1.653$. [36].

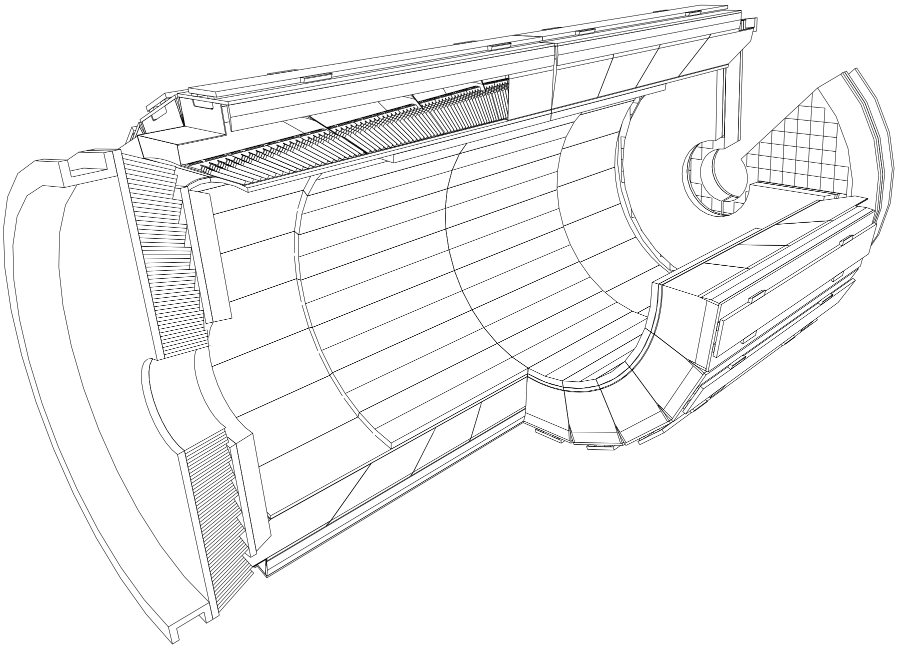


Figure 4.6: 3D render of the geometry of the CMS ECAL, showing the segmentation of the barrel into ECAL modules (divisions in η) and supermodules (divisions in ϕ). The internal radius is 1.29 m and length 6.38 m. [36].

4.3.2 Electromagnetic Calorimetry

With the relatively large tracker volume (dictated by the requirement for a measurable sagitta of high-momentum charged tracks[3], the electromagnetic calorimeter[36] is

[3] $s \approx 200$ µm for a 1 TeV charged particle within the tracker.

necessarily a compact design to fit inside the magnet volume. The primary physics consideration was the optimal reconstruction of approximately 50 GeV photons (such as those produced in pairs by light Higgs decay). This requires both sub-percent energy resolution and a fine-grained structure to allow multiple showers to be distinguished. As close to hermetic coverage as possible is required to allow accurate missing transverse energy measurements to be made.

The calorimeter is built from lead tungstate crystals[37]. This material has a density $\rho = 8280$ kg/m^3 and atomic numbers $Z = 74$ and 82, which result in a short radiation length $\chi_0 = 0.89$ cm and high radiation hardness. Scintillation occurs quickly, with 80% yield within a 25 ns crossing, reducing the risk of interference between events (although measurements are integrated over a 250 ns window). The light yield is 30 gamma/MeV (with $\lambda = 430$ nm), requiring amplification with an avalanche photodiode or vacuum phototriode, in the barrel and endcap respectively. The gain of the amplifiers is highly temperature-dependent, requiring the temperature to be maintained within 0.1 K to maintain resolution performance[4].

The electromagnetic calorimeter (ECAL) geometry (see Figure 4.5, Figure 4.6) consists of a barrel with radius $r = 129$ cm extending to $|z| = 314$ cm (corresponding to $|\eta| < 1.479$) and a pair of endcaps covering $1.479 < |\eta| < 3.0$. The barrel crystals have a face 22×22 mm (corresponding to $0.0174 rad$ in η, ϕ), and are 230 mm deep (corresponding to $25.8\chi_0$[5]). The endcap crystals have a slightly larger face (28.6×28.6 mm) and are slightly shallower (220 mm, $24.7\chi_0$). The crystals are separated by a carbon fibre matrix, and are projected slightly away from the origin in both the η, ϕ directions (by 3°) so that photons from the interaction point cannot pass wholly through this low density material. Between supermodule assemblies, and between the barrel and endcap are detectable gaps which must be considered during reconstruction (usually by blacklisting tracks pointing into these regions). There are a total of 61200 crystals in the barrel and 7324 in each endcap.

The energy resolution $\frac{\sigma_E}{E}$ of the ECAL on 50 GeV photons is 0.6%, decreasing to 0.4% for $E > 200$ GeV and increasing to 1.5% for $E < 10$ GeV.

[4] This is complicated by approximately 20 kW of electrical heating from the (wholly enclosed) inner tracking system.

[5] The tracker provides up to $1.4\chi_0$ extra radiation depth, depending on η.

4.3.3 Hadronic Calorimetry

The design of the hadronic calorimeter (HCAL) was not driven by any one physics process, but rather by the need to provide a hermetic missing transverse energy measurement and the need to distinguish leptons produced in the primary event from those produced by heavy flavour jets. The design maximises the interaction depth within the solenoid, and supplements it with additional calorimetry volumes in the central and very forward regions.

The HCAL[38] consists of machined brass plates, interleaved with plastic scintillators, a technology also used by UA1 and CDF. The barrel consists of fifteen layers, each consisting of a 50 mm brass absorber and a 3.7 mm scintillator tile. This is supplemented by an additional 9 mm scintillator on the inside and steel structural elements on the outside, for a total interaction depth $\lambda_0 > 5.8$. Readout is achieved with wavelength-shifting fibres, read out by hybrid photodiodes. The barrel is segmented into towers with η, $\phi = 0.087$ ($\frac{1}{25}$th of ECAL granularity), covering the region $|\eta| < 1.4$.

The HCAL barrel occupies the volume between $1.81 < r < 2.95$ m, and the endcaps $3.90 < |z| < 5.68$ m. Thus, the HCAL constitutes the majority of the volume within the solenoid.

For sufficient resolution, an interaction depth of $10\lambda_0$ is desirable, so the central barrel region ($|\eta| < 1.26$) is supplemented by an additional layer outside of the solenoid[6]. This is termed a “tail-catcher” and consists of one or two scintillator layers, separated by an 18 cm iron absorber (part of the solenoid return yoke). This provides at least $10\lambda_0$ over the instrumented region.

The endcaps cover the region $1.3 < |\eta| < 3.0$. At lower η the granularity in η, ϕ is the same as the barrel, nearer to the beamline towers are merged into larger ones. Compared to the barrel, there are more independent readout channels, motivated by the need to alter corrections as the scintillators become radiation damaged.

To provide coverage in the region $3 < |\eta| < 5$, the forward hadronic calorimeter is placed 11.2 m either side of the interaction point. This region experiences the highest hadronic densities, and consists of a 1.65 m deep steel absorber, containing quartz fibres which collect Cherenkov radiation and channel it to photomultipliers. The forward calorimeter is divided into towers with granularity in η, $\phi = 0.175$.

[6] The barrel at high η has sufficient thickness from incident angle.

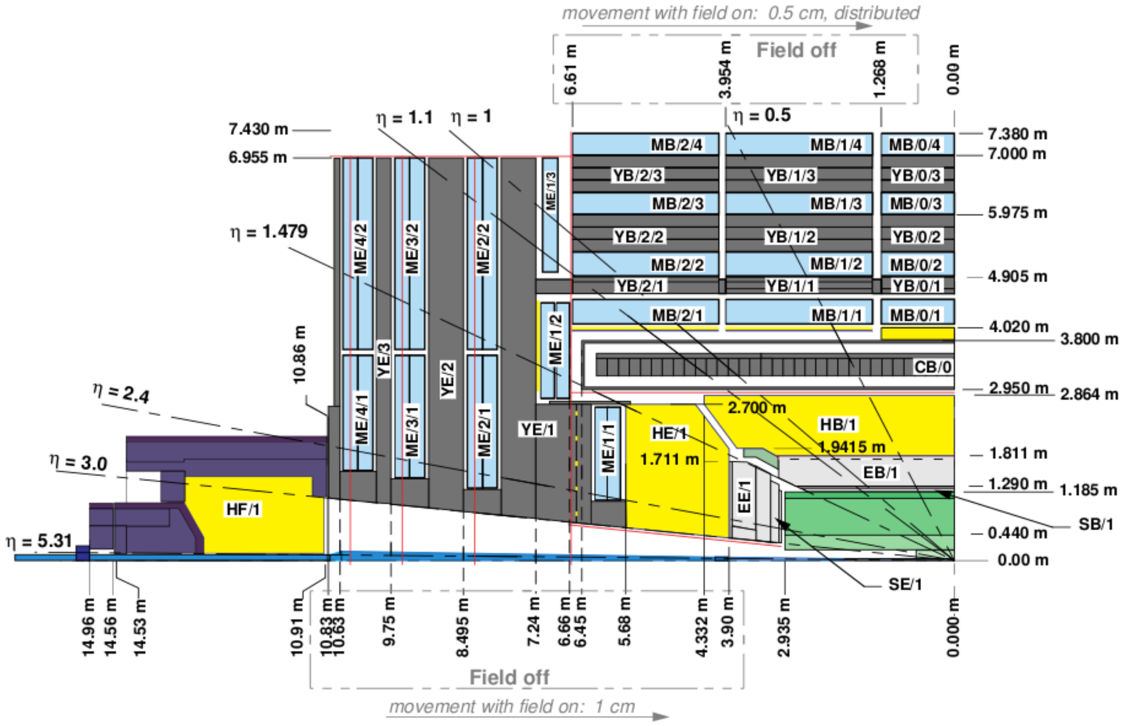


Figure 4.7: Quarter-view of the outer layers of CMS, showing the barrel and endcap components of the Muon detector (blue) and HCAL (yellow). Adapted from [39].

The resolution of the hadronic (barrel) calorimeter is 20% for a 50 GeV charged pion.

4.3.4 Muon Detection

The majority of the CMS volume is dedicated to detecting muons (as implied by the name of the experiment). Although the best momentum measurement of muons is typically made by the inner tracking system (before the muon has passed through the dense ECAL, HCAL and solenoid), the outer muon detectors allow unambiguous detection of muons for the purposes of identification, triggering, and refining the momentum measurement. The muon detectors[39] are interleaved with the return yoke structure (see Figure 4.7) for the solenoid.

Three types of gaseous detector are used; resistive plate chambers (RPCs), drift tubes (DTs) and cathode strip chambers (CSCs). The choice of detector is influenced by the local magnetic field and the amount of neutron-capture induced radioactivity of the surrounding bulk.

RPCs are used over the whole region, since they provide good time resolution (to determine to which crossing a muon belongs) and fast response (for triggering) but have poor position resolution. They consist of two, 2 mm gas gaps between bakelite plates with a central aluminium readout strip, and provide a time resolution of 2 ns and readout within a 25 ns crossing.

In the barrel $|\eta| < 1.2$, with low magnetic field and radiation, there are four superlayers of twelve drift tubes layers each, associated with one or two RPC layers. The drift tubes are divided into $r - \phi$ measurement and z measuring layers, so that each superlayer both measures a position and a vector. Individual tubes are 42 mm × $13mm^2$, with a central anode wire and maximum drift time of 400 ns. Superlayer boundaries are offset to ensure each muon should traverse at least three of them. There are 250 chambers in total providing a resolution of 100 µm in ϕ and 1 mrad in direction.

In the endcaps, $1.2 > |\eta| > 2.4$, the higher magnetic field and radiation background necessitated the use of CSCs, which are trapeziodal with cathodes on the inner surfaces and anode wires inside. The position of a hit within a CSC is determined by the charge share between the walls. Position resolution is slightly worse than the barrel at 200 µm and direction much worse at 10 mrad.

Muons are detected with greater than 95% efficiency. The momentum resolution (using the muon system only) is approximately 9% for 100 GeV muons. For muons with $p_T < 100$ GeV, a sub-percent measurement can be obtained with the inner tracker and the muon system measurement is dominated by scattering before reaching the outer layers. For higher momentum muons the position resolution of the muon chambers becomes the limiting factor, and the combined resolution for 1 TeV muons is approximately 4%.

Unlike the other detectors, it was possible to extensively calibrate and test the muon systems in-situ before collisions by measuring muons from cosmic rays. The resolution was measured as 8% for 500 GeV muons[40] with data taken during the CRAFT[7] exercise in late 2008.

[7] **C**osmic **R**un **a**t **F**our **T**esla

4.3.5 Triggering

Under ideal conditions, the $40MHz$ collision rate multiplied by the approximately 1.5 MiB[8] detector readout volume would grossly exceed the capacity of any conceivable storage or transmission system likely to be available in the near future. The trigger identifies events of interest and selects the O(100) Hz[9] which can be permanently stored.

The trigger is divided into the "Level 1 Trigger" (L1T), which runs on custom, dedicated hardware and reduces the event rate to approximately 50 kHz, and the "High Level Trigger" (HLT) which runs on ordinary computer hardware and performs software reconstruction and selection of events to produce the final output rate.

Each detector component includes buffers as part of the readout system, which store the complete detector readout for each event in a pipeline. The pipelines each hold 128 events (corresponding to 3.2 µs), during which time a level 1 decision has to be made before the event is overwritten. The speed of light (7.5 m per 25 ns crossing) requires that the level 1 trigger is very near the detector to allow sufficient processing time (1 µs).

At level 1, only the calorimeters and muon system are accessible; the computation time to unpack or reconstruct any of the tracker is prohibitive. The ECAL and HCAL are summed into coarse "towers", consisting of 5 × 5 ECAL crystals and a single HCAL segment. The Regional Calorimeter Trigger (RCT) identifies electron/photon candidates ("primitives") and the overall energy sums in each of 18 regions. The Global Calorimeter Trigger (GCT) performs jet finding over larger regions (12 × 12 towers), and calculates the overall missing E_T direction and magnitude.

The level 1 Global Trigger receives the missing E_T and the four highest energy electrons, central and forward jets from the GCT, and the highest energy four muon candidates from the Global Muon Trigger (GMT). Up to 128 trigger bits can be set by boolean and counting logic on the primitives. Some of these bits are used for other purposes, such as the presence of any particles in either forward region (unidirectional indicating a beam-gas collision).

Events passing level 1 selection are transferred to readout buffers (causing a short detector dead time) and processed for transmission (zero-suppression, data compression, etc). The events are then transmitted to the surface and stored in a front-end buffer.

[8] Binary prefixes are defined as multiples of 2^{10} instead of 1000, ie 1 MiB = 2^{20} B = 1048576 B (and equivalently for KiB, GiB etc.), to avoid ambiguities when using SI prefixes.

[9] This proves a somewhat flexible number, with 2010 rates reaching 300 Hz and 1 kHz under consideration.

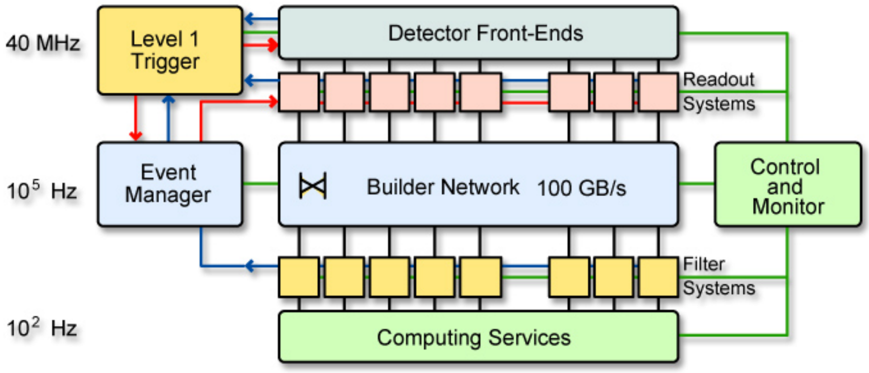


Figure 4.8: CMS data acquisition architecture, showing the Level 1 and High Level triggers. [34]

The HLT itself is implemented within the CMS Software Framework (CMSSW), with around 2000 instances running. Each event is processed by a single instance. The average HLT decision must be made in 40 ms to avoid event dropping. The architecture of the HLT is shown in Figure 4.8.

Within the HLT, all possible trigger paths are executed regardless of the L1 bits set (although many check for the presence of an appropriate L1 bit as a first step). The dependency graph of all utilised steps is calculated and those that appear in multiple paths are only executed once. Trigger paths have to reject the majority of events quickly to maintain the average runtime. This generally means that paths start with calorimetry (including missing E_T), before moving onto progressively more expensive operations on the muon chambers, pixel detector and finally the full tracker.

Events passing HLT selection are split into primary datasets[10], depending on which triggers they have passed. Datasets are packed into bulk ROOT files and transmitted from Point 5 to the main CERN site for custodial storage and distribution to other computing sites.

[10] $O(50)$ datasets were anticipated in the long term, but in early data running this has been restricted to $O(10)$. An event may be placed in multiple primary datasets.

4.4 Computing

4.4.1 CMSSW

All CMS “physics” computing requirements[41], including Monte-Carlo generation, the HLT, event reconstruction and analysis are performed using the CMS Software Framework (CMSSW), a predominantly C++ framework that runs in a Linux[42, 43] environment.

Any process run under CMSSW is controlled by a python configuration file, evaluated by an embedded interpreter, and defines:

- A data source, either one or more ROOT files or an event generator.
- Modules, which perform simple steps, abstracted as “Producers”; which create new collections of objects, “Filters”; which reject events based on some criteria, and “Analyzers”; which produce output based on the event content. Configuration parameters such as cut values can be specified here rather than compiled in.
- Services, which perform ancillary functions such as message logging, store output, provide conditions data or allow the geometry to be queried.
- Paths, which are sequences of modules that will be run on events. All paths are run for each event, but modules in common will only be run once according to the calculated dependency graph.

A visualisation for the CMSSW configuration is discussed in Appendix A.1.

All modules and services are C++ classes inheriting from framework base classes. To avoid individual installations requiring gigabytes of source code and recompilations for each change, a user’s software area contains only code under active modification, with the default versions for all other libraries used. A Perl-based build tool (`scram`) is used to handle the merging between local modifications and globally available versions.

Performance typically varies from 10 Hz to 1 kHz, depending on the complexity of the operations and the CPU, memory and disk/network bandwidth of the host. While event processing is an intrinsically parallelisable task (since each event is essentially an independent processing task), CMSSW is not itself multithreaded[11]. Instead, it is

[11] Some work has been done on multithreading within events, by parallelising independent parts of the reconstruction. However, in practice this provided a limited benefit as the tracker reconstruction forms a single-threaded bottleneck meaning the overall reconstruction time is not significantly reduced for the additional scheduling difficulties this approach creates.

intended that the input is sliced into a large number of smaller tasks, each of which runs as a separate instance, after which the user is responsible for appropriately merging the results.

4.4.2 The Event Data Model

Data storage for CMS requires a form that:

- Is relatively compact per-event, to reduce the overall storage requirements.
- Supports efficient sparse access, so analyses considering only a small subset of the available data per-event do not require each entire event be read, particularly over networked file systems.
- Allows the serialisation and deserialisation of the C++ classes which represent reconstructed particles, with provision for these structures to change with time, and without resort to raw packed binary data requiring custom streaming to be implemented for every class.
- Stores the full provenance information of data, to ensure subsequent reproducibility.
- Contains both the events themselves and any necessary metadata in the same volume.

The ROOT[44] file format satisfies these requirements[12]. Events containing the raw detector output and reconstructed objects are typically about 2 MiB, and are packed into files containing $O(1000)$ events[13].

Types of data within a file are split into "tiers". These are:

- RAW - Raw binary readout from the detector.
- DIGI - Raw data converted into classes representing individual detector components, eg tracker hits and calorimeter crystals.

[12] With caveats; true serialisation of C++ is not possible due to the lack of reflection support within the language, so XML dictionaries of the appropriate formats are also required (part of the standard CMSSW distribution), and significant modifications to some header structures was required to avoid the need for continuous rewinding of the file pointer and thus inefficient read-ahead behaviour when performing sparse reads.

[13] This is a tradeoff between larger files, which are more efficient for storage and retrieval on tape-based mass-storage systems, and smaller files, which are more portable for local use and reduce the losses associated with any single corruption event.

- HLT - Reconstructed objects created by HLT paths, and the HLT decision.
- RECO - Particles and related data found during complete offline reconstruction of the event.
- AOD - Subset of RECO containing only high level physics objects.
- GEN - Output of the Monte-Carlo generator.
- SIM - Information related to the Monte-Carlo simulation of the generated particles passing through the detector.

All data taken is promptly reconstructed, and a custodial copy containing RAW and RECO data stored at CERN and at least one Tier-1 computing site. Approximately four times per year the complete dataset is re-reconstructed using a more recent version of CMSSW (and hence more recent object reconstruction, calibration constants, etc). AOD data is currently sparsely used but as the size of the main dataset grows (and hence the computing time required to analyse it), it is intended that the majority of analysis will be done with AOD.

4.4.3 The Grid

The computational requirements for CMS (and the other LHC experiments) far exceed the practical storage, bandwidth and CPU capacity of the CERN computing facilities, necessitating an infrastructure for distributed computing.

The LHC Computing Grid (LCG) project[45] provides the underlying architecture and coordination for the grid project, while the actual hardware is provided by participating universities and similar facilities, supporting those experiments in which they have an interest. As of early 2011, the Grid encompasses approximately 2×10^5 CPUs[46], although each experiment only has access to a subset of that capacity.

CMS divides accessible computing sites (Figure 4.9) into tiers[47]:

- Single Tier-0 site at CERN, initially receives data, performs prompt reconstruction and stores a tape copy
- Eight Tier-1 sites, national-level computing sites with mass-storage capability. Additional copies of each primary dataset are stored at one or more Tier-1 sites, and

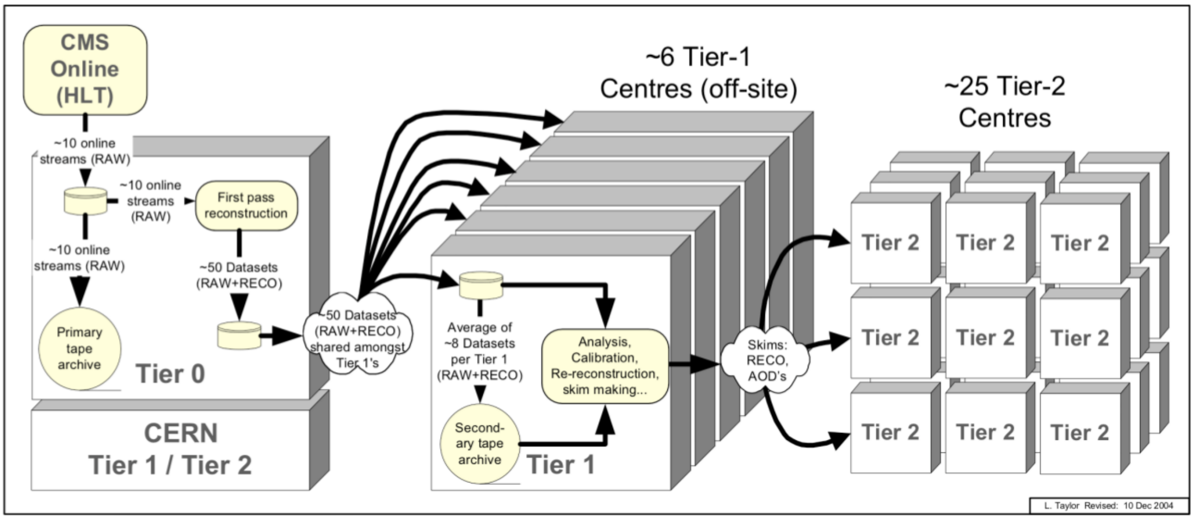


Figure 4.9: Diagram showing the composition of the CMS computing grid (although the number of Tier 1 and Tier 2 sites has grown), and the seperation of work between prompt reconstruction at Tier 0 and later reconstruction passes at Tier 1 sites. [47].

their CPU capacity is used for data reprocessing, official skimming and prioritised analysis, but is inaccessible to normal users.

- $O(50)$ Tier-2 sites, support local users and one or more analysis groups, with CPU capacity used for analysis and Monte-Carlo production, provide centrally managed space for analysis groups and host datasets fetched from Tier-1 sites as required by their users.
- $O(50)$ Tier-3 sites, similar to Tier-2 (although usually smaller), maintained on a best-effort basis and not forming part of capacity calculations and pledges.

A Grid site consists of one or more Compute Elements (CEs) and one or more Storage Elements (SEs), as well as ancillary services such as database caches. There exist a wide variety of underlying technologies that can be used for each of these (SGE, Condor, etc for CE and Storm, Lustre, HDFS, DCache, etc for SE), but the grid middleware ensures an approximately consistent interface is presented to running jobs.

Chapter 5

Reconstruction

While quick reconstruction of physics objects is performed during the HLT, full reconstruction occurs offline due to the considerable CPU time required, primarily to completely unpack and reconstruct the tracker, and also to perform more accurate reconstruction of the other detectors. Where multiple methods for reconstructing any individual particle exist, all are run and the results of each stored[1].

5.1 Muons

Muons are reconstructed using both the inner silicon tracker (Section 4.3.1) and the outer muon chambers (Section 4.3.4). Tracks in each of these sectors are reconstructed separately, then muons can be constructed[34, 40, 48] by matching and refitting compatible pairs of tracks.

In the HLT, muon reconstruction consists of finding muons in the outer chambers and then using the extrapolated track to seed track-finding in a limited region of the inner tracker in which to find a compatible track. In the offline reconstruction, the two detectors are reconstructed independently and muons are then created by finding compatible track pairs.

[1] The multiplicity of different reconstruction algorithms for common physics objects, and the quantity of metadata required for each one has resulted in a significantly larger volume of RECO-level information than originally anticipated by CMS; from $O(250)KiB$ in the TDR[47] to $O(1)MiB$ per event.

5.1.1 Muon Chamber Reconstruction

Due to the large size of the muon detector system ($r = 7.3$ m, $\frac{r}{c} = 24$ ns), and the significant drift times in the DT and CSC detector elements (up to 400 ns, Section 4.3.4), the first part of muon reconstruction consists of using the fast RPCs to assign signals to the appropriate bunch crossing.

In the barrel DT stations, the only information available is the drift time (and the drift speed is not linear). This does not uniquely identify a point within the tube and reconstruction is by a three-step process. An estimate of the muon direction and position is used as a seed, which is then refined by reconstructing the adjacent drift tube layers into an $r - \phi$ or $r - z$ measurement and finally reconstructing the whole superlayer into a position and momentum measurement.

Reconstruction of the $r - \phi$ and $r - z$ layers is done by finding the most separated hits in the layers (producing a track angle compatible with the interaction region), and then fitting and disambiguating the solution using the remaining layers. For cases of multiple solutions, the largest number of hits and best χ^2[49] value is used to determine the optimal solution.

In the endcap CSC stations, the charge on both the wire anodes and orthogonal strip cathodes is stored. The charge distribution across several strips or wires is fitted (after correction) to find the average position in each direction, from which a two-dimensional hit position can be calculated.

All six layers of each CSC station are then combined. The first and last hits used to form an axis, which is then progressively refined by finding matching intermediate hits. At least four hits out of six are required to accept the track segment, with the position given by the two-dimensional hit nearest to the calculated track axis.

The RPCs are reconstructed by clustering all strips that have been fired and taking the centre of the overall cluster as a position measurement.

To construct overall tracks in the muon chamber, the position and momentum states on the innermost barrel and endcap layers are used as seeds. The indicated track from the inner layer is propagated outwards to the next layer, using both radiation and magnetic field simulation. If no hit can be found the track is propagated outwards to the next-again layer (to allow for gaps in muon stations). Tracks are refined using the Kalman-filter fitter[50]. In the barrel both the position and momentum states of subsequent layers are used as inputs, whereas in the endcaps the greater magnetic field complexity makes using the outer momentum states difficult, so after the innermost layer only the position information is used. In both cases, the beamspot is used as an additional position in the track fitting.

5.1.2 Tracker Reconstruction

Reconstruction of the tracker[51] consists of clustering individual strip or pixel hits into calculated hit positions, finding track seeds, and finally fitting and refining full tracks based on the seeds. Reconstruction of the full tracker is by far the most computationally intensive part of reconstruction.

Clustering requires a database containing the noise and gain measured for each channel, so the measured charges for each channel can be corrected (since $\approx 20\%$ gain variation is expected in the strips). In the strip tracker, clusters are seeded by a single strip with $\frac{S}{N} > 3$[2], and then expanded to include adjacent strips with $\frac{S}{N} > 2$, and requiring $\sum S > 5 \times \sqrt{\sum N^2}$. The hit position is determined from the weighted average of the charge distribution.

Clustering in the pixel detector is similar, but requiring $\frac{S}{N} > 6$ for the original seed, which is then expanded recursively with any pixels at least diagonally adjacent and with $\frac{S}{N} > 5$. The overall cluster must satisfy $\frac{S}{N} > 10$. The hit position is estimated in η and ϕ independently, using the charge distribution and predicted track angle (based on a straight line to the interaction point). The position estimate is refined once the track parameters are better determined.

Track seeds are found in the pixel layers by locating pairs of hits in any combination of the three layers, requiring that they extrapolate to within the interaction region.

The track seeds are then extrapolated outwards into the strip layers. First, the possible hit regions of the tracker are determined from the seed direction, and the available clusters found. The track is propagated to each potential layer in turn, using a Kalman-filter based fitter updated with each new position measurement (as with the muon chambers). Where there are multiple candidate hits within the calculated error cone, a separate track candidate is created for each. In addition, a track candidate is created at each layer with no hit in that layer, to account for the possibility of a missing

[2] **S**ignal, **N**oise.

hit. To avoid exponential growth of track candidates, a maximum child count (5) from each seed is maintained and the worst-fitted candidates discarded at each level.

After the track candidates are propagated to the outermost layer, candidates sharing more than half of their hits are identified and the best fitted (by number of hits and χ^2) is chosen, and the others discarded. This is done for all the candidates from a single seed, and also on the entire track candidate collection once all seeds have been propagated.

Fitted tracks are refined using a combined Kalman-filter and smoothing pass, starting from the innermost hit and working outwards. At each layer, the hit position is refined using the track angle, and the track parameters updated appropriately. This is repeated until the outer layer is reached, at which point a third, smoothing pass is made starting from the outside.

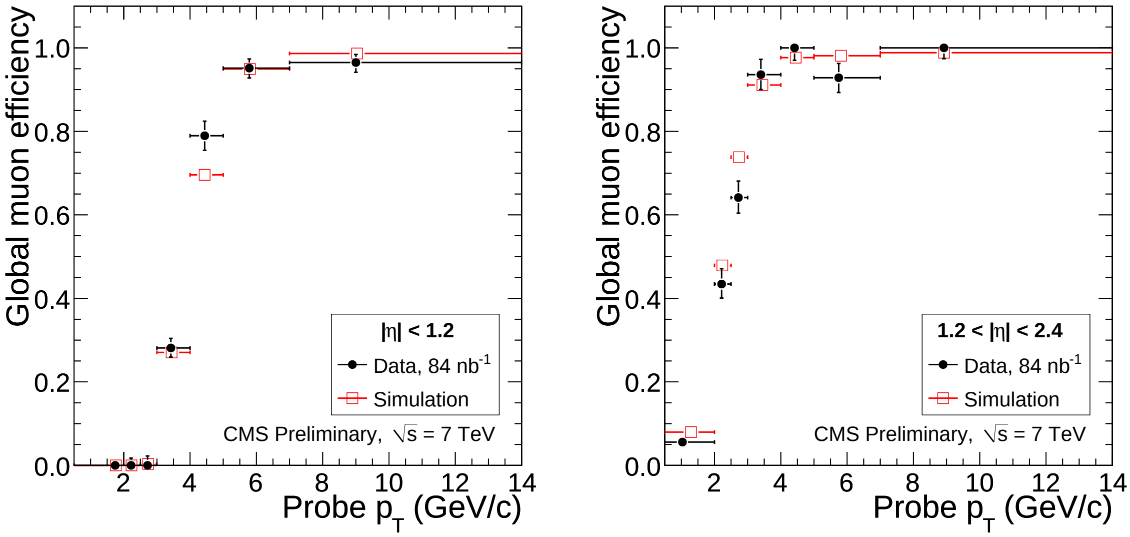
The fully fitted and refined track is then used to calculate the momentum and transverse impact parameter, and assign a charge based on the curvature direction.

5.1.3 Muon Combination

Once the tracker and muon chambers are reconstructed, several classes of muons can be found.

- *Global* muons are seeded from a muon chamber track, propagated to the inner tracker and a compatible track found. A single track is then fitted combining all the hits in these two tracks. For muons with $p_T < 200$ GeV, the inner tracker provides better momentum resolution than the muon chambers.
- *Tracker* muons are seeded from the inner tracker (using any good-quality track with $p_T > 2.5$ GeV), propagated outwards to the muon chamber and any matching muon track or DT/CSC stub found. This improves efficiency for muons with $p_T < 5$ GeV, which frequently do not fire multiple muon layers.
- *Standalone* muons are tracks found in the muon chambers for which there exists no corresponding inner track. This accounts for $< 1\%$ of muons from collisions, with $> 99.9\%$ of standalone muons originating from cosmic rays.

All of these muon categories are represented by the same reconstructed object class, and stored in the same collection, with flags set to indicate their origin.



(a) Barrel

(b) Endcap

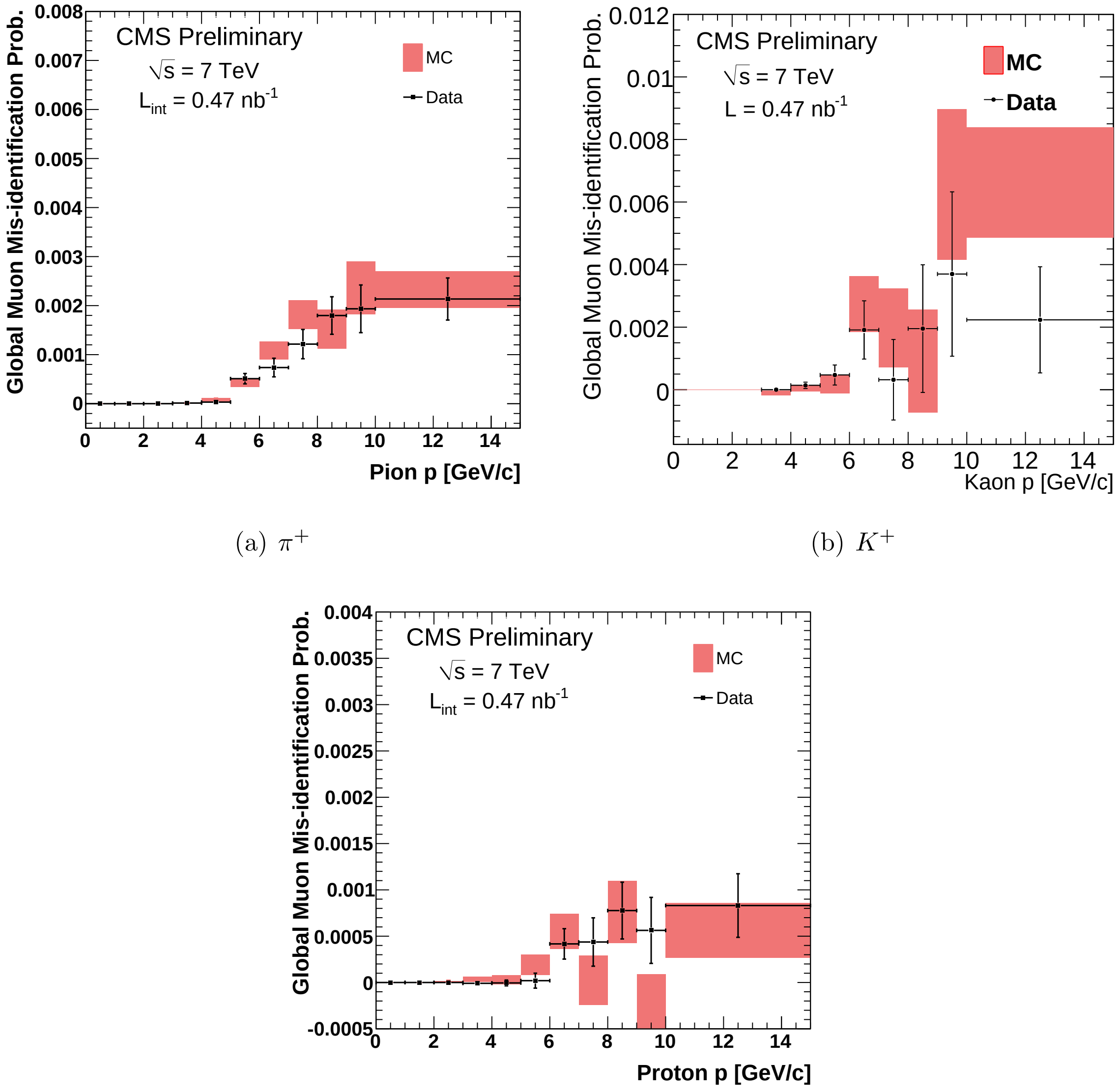
Figure 5.1: Identification efficiencies for global muons in the barrel and endcap regions, measured in early data. [48].

The high purity of global muons ($\approx$ 98%[48]) means that relatively little further identification is required. The main background contributions are pions and kaons in the central region where the HCAL depth is at a minimum. The efficiencies and fake rates for global muons are shown in Figure 5.1 and 5.2.

Further identification ("tight muon") requiring global track $\chi^2 < 10$, >= 10 hits in the tracker and impact parameter $|d_{xy}| < 2$ mm to the primary vertex increases global muon purity to 99.5%, for which the efficiency (for muon $p_T > 10$ GeV) is 95%.

5.2 Electrons

At Level 1, electrons and photons are identical objects consisting of a confined energy deposit in the ECAL (Section 4.3.2). HLT and offline reconstruction[52] is very similar, performing a more detailed reconstruction of the energy deposits in the ECAL before searching for a compatible track, although the latter uses a more detailed (and time-expensive) track reconstruction method.



(a) π^+

(b) K^+

(c) p

Figure 5.2: Probability of pions, kaons and protons to fake a global muon, as measured in early data. [48].

5.2.1 Supercluster Reconstruction

Because of the significant radiation depth ($2\chi_0$ at $|\eta| = 1.5$) of the tracker, there is high probability[3] of electrons undergoing bremsstrahlung in one or more tracker layers before reaching the ECAL surface. To account for this, ECAL reconstruction searches not only for the primary electron energy deposit but multiple separated deposits at the same η.

Seed crystals (with $E_T > 1$ GeV) in both barrel and endcap are identified and used as the centres of clustering regions. In the barrel 5×1 (in η) crystal "dominoes" are added from up to 17 crystals either side of the seed in ϕ (corresponding to ± 0.3 rad), providing the domino has $E_T > 100$ MeV. In the endcaps 5×5 regions are used instead of "dominos", collected if their centres fall within $\Delta\phi < 0.3$, $\Delta\eta < 0.07$ of the seed crystal.

Superclusters for electron reconstruction are required to have $E_T > 4$ GeV. In addition, a hadronic veto of $\frac{E_T^{HCAL}}{E_T^{supercluster}} < 0.15$ is applied, where the HCAL energy is the sum of towers with $\Delta R < 0.15$ from the energy-weighted centre of the supercluster.

5.2.2 Track Matching

Based on the supercluster E_T, an approximate track curvature is calculated and two track hypotheses (corresponding to either electron or positron) made. The hypotheses are compared to the available pixel track seeds (produced as per Section 5.1.2), and compatible seeds within a loose region identified ($\Delta\phi < 0.14$ rad, $\Delta z < 5$ cm).

Electron track reconstruction is performed using a procedure similar to that in Section 5.1.2, except that the Kalman filtering algorithm is modified to include a model of bremsstrahlung energy loss at each layer. The track is refined using a Gaussian Sum Filter (GSF)[53] method which better models the non-gaussian effects that bremsstrahlung in thin layers has on the track parameters.

The GSF track refinement improves the positional accuracy of the track propagated to the calorimeter surface compared to the general Kalman filter technique for electrons with $p_T < 30$ GeV, but does not significantly alter the momentum resolution.

[3] Approximately 50% of electrons have radiated > 50% of their initial energy before reaching the calorimeter surface.

To match the GSF track and supercluster, the distance between the energy-weighted supercluster centre and extrapolated track incidence on the ECAL surface is required to be $\Delta\eta < 0.02$, $\Delta\phi < 0.15$.

5.2.3 Electron Identification

There are several categories of "fake" electrons, covering both those cases where an electron is reconstructed where none existed, and where the reconstructed electron is real but did not originate from the primary event.

There are two main cases involving non-electrons faking an electron. The first is a single charged hadron (typically a charged pion) that interacts in the ECAL and produces a sufficient energy deposit to seed an electron. The second case arises from overlap between a charged hadron and one or more almost-collinear neutral hadrons decaying to photons, such that the ECAL deposit and track appear to be connected. In both of these cases the $\frac{E_{supercluster}}{p_{track}}$ ratio will be small, in the former case because the ECAL is calibrated for photons and hadronic energy deposits will be underestimated, and in the latter case because the p_T spectrum of π^0 falls off rapidly.

Real but non-original electrons originate from several different sources. The majority result from in-flight semileptonic meson decays, which can be identified by looking for the rest of the hadronic jet near to the electron axis. Longer-lived parent particles ($\mathbf{b}$, τ) will often produce electrons with a large impact parameter relative to the primary vertex.

Another important source is electron-positron pairs produced by photon conversions ($\approx 34\%$[52]), where the photons radiate from the original event or are produced by neutral hadron decays. Electrons from conversions can be identified by an opposite-charged track with small relative impact parameter and usually are missing one or more pixel hits (before the photon converted).

The variables used for electron ID are:

- $\frac{E_T^{HCAL}}{E_T^{supercluster}}$ (as per Section 5.2.1)
- $\Delta\phi(track, supercluster)$, $\Delta\eta(track, supercluster)$ (as per Section 5.2.2)
- $\sigma_{i\eta i\eta}$, a variable for the shape of the ECAL shower, defined as the root mean square of the shower width in η of the 5×5 crystals centred on the seed crystal

	WP80		WP85		WP95	
	Barrel	Endcap	Barrel	Endcap	Barrel	Endcap
$H/E <$	0.04	0.025	0.04	0.025	0.15	0.07
$\Delta\phi <$	0.06	0.03	0.06	0.04	0.8	0.7
$\Delta\eta <$	0.004	0.007	0.006	0.007	0.007	0.01
$\sigma_{i\eta i\eta} <$	0.01	0.03	0.01	0.03	0.01	0.03

Table 5.1: Cut variables for the WP80, WP85 and WP95 (nominal efficiency) electron identification working points.

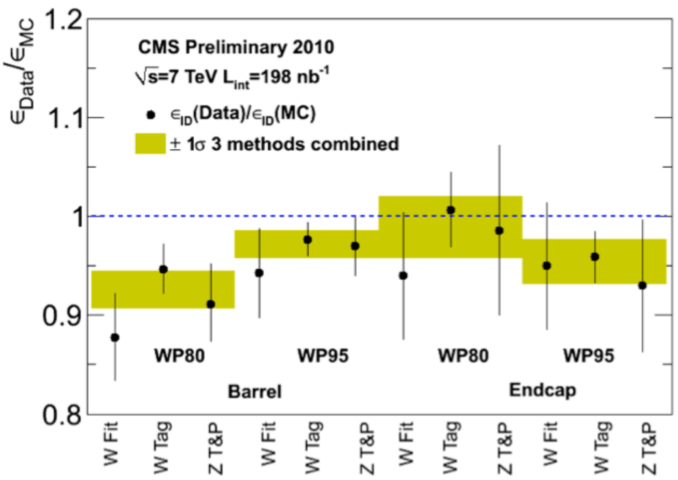


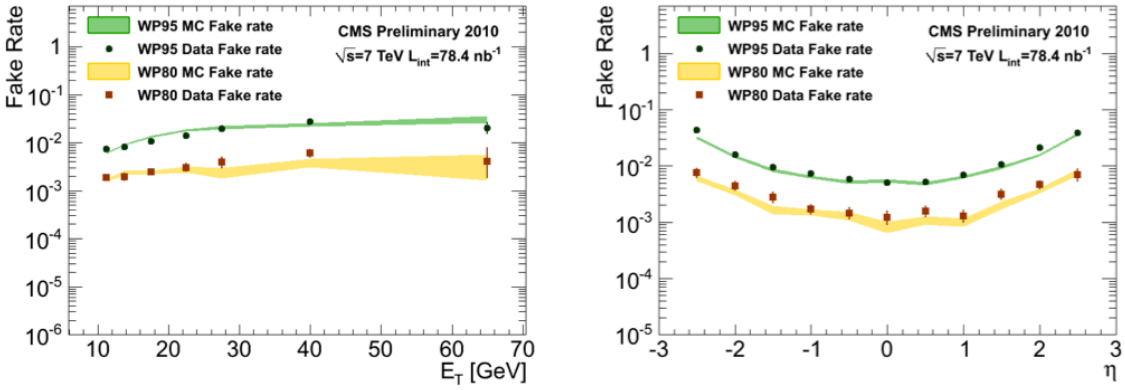
Figure 5.3: Actual efficiency of the “WP80” and “WP95” working points, as a function of their nominal efficiency, measured in early data using W and Z events. [52].

From these variables, a series of working points with decreasing efficiency and increasing purity are defined, denoted WP x where x% is the nominal electron efficiency. The working points used in Chapter 6 are given in Table 5.1.

The actual efficiencies (from data) of the “WP80” and “WP95” working points, compared to their nominal efficiencies are shown in Figure 5.3. The measured fake rates for these working points are shown in Figure 5.4.

To reject electrons from conversions, the electron is first required to have a hit on the innermost pixel layer. The collection of general, Kalman-filter tracks is then searched to find possible (opposite-charge) conversion partner tracks by calculating the quantities:

- $\Delta R(e, track)$, the opening angle between the tracks.



(a) p_T

(b) η

Figure 5.4: Fake rate (fraction of fake identified electrons) for the "WP80" and "WP95" working points, measured in early data. [52].

- $|cot(\theta_e) - cot(\theta_{track})|$, the cotangent difference of the track polar angles.
- $|d_{xy}|$, the distance at closest approach between the tracks in the transverse plane.

The electron is rejected as a probable conversion if there is an opposite sign track satisfying $\Delta R < 0.3$, $|\Delta cot(\theta)| < 0.02$ and $|d_{xy}| < 200$ µm. This rejects approximately 90% of electrons from conversions.

5.3 Taus

The short tau lifetime ($\tau = 290$ fs, $c\tau = 89$ µm[24]) prevents direct observation in CMS, hence by "tau" we mean a jet of light mesons from a hadronic tau decay (tau decays to electrons or muons are reconstructed by the standard methods in Section 5.1 and Section 5.2). The distinctive features of tau jets compared to QCD jets are:

- The secondary vertex is sufficiently displaced that the tau jet is usually colour-isolated from the underlying event, which tends to produce a more tightly confined and isolated jet.
- Tau jets consist of a small number of unique decay modes, which we can attempt to reconstruct and identify.

Hadronic tau jets suffer from a number of backgrounds, and while it is possible to select a sample with relatively high tau purity, this is usually at the cost of very low efficiency (and possibly biassed selection).

- QCD jets are the largest background, and due to the very large production cross section cannot entirely be eliminated. Tau jets are more isolated and generally have a lower track multiplicity, as well as a slightly displaced secondary vertex, but QCD jets can (at low probabilities) fake all of these properties.
- Electrons appear as the extreme case of a tau with a single charged hadron. These can be reduced by looking at the different ECAL/HCAL signatures of electrons and hadrons, and the presence of bremsstrahlung photons.
- Muons can sometimes fake tau jets, in the case where some energy is deposited in the HCAL, they appear (like electrons) as a jet with a single charged hadron. This can be reduced by comparing the HCAL deposit and track momentum, and looking for the presence of the muon chamber track.

5.3.1 Particle Flow

The jets used for all the tau algorithms are derived from the particle flow algorithm. The aim of the particle flow (PF) algorithm, described in detail in [54], is to reconstruct all final-state particles at once, using information from all the detectors.

Tracks are reconstructed using an iterative method, where track-finding initially takes place with very harsh quality cuts applied, after which the selected tracks and associated hits are removed and tracking run again with looser cuts (but a much smaller pool of unselected hits), until no further tracks can be found. The removal of good tracks allows the subsequent, looser passes to run with a minimal fake rate due to the limited combinations possible.

Calorimetric clusters are formed separately in each calorimeter system (ECAL and HCAL, barrel and endcaps). Local maxima in each system are used as seeds and clusters grown outwards as long as the deposited energy is 2σ above expected noise levels.

The identified tracks and energy clusters are linked into *blocks* based on compatibility within calculated measurement errors and blocks further classified as *charged hadrons*, *neutral hadrons* or *photons*. Dedicated reconstruction for *electrons* and *muons* (based on

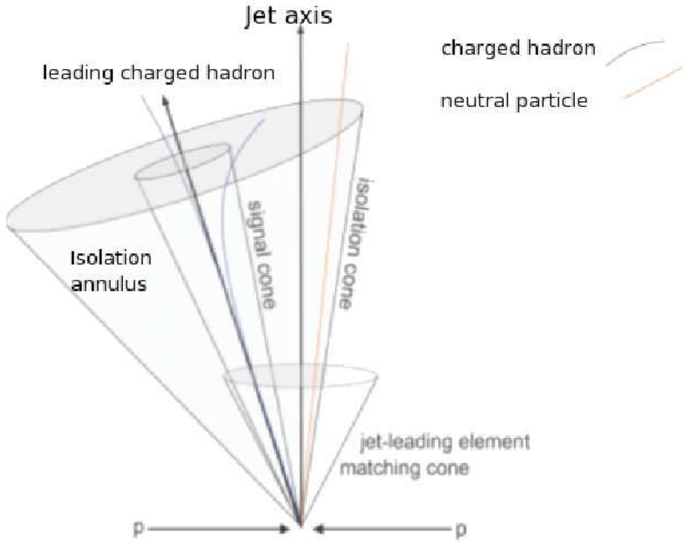


Figure 5.5: Diagram illustrating the construction of a tau candidate, with the original jet axis, identified leading track, signal cone round the leading track and isolation cone. [56]

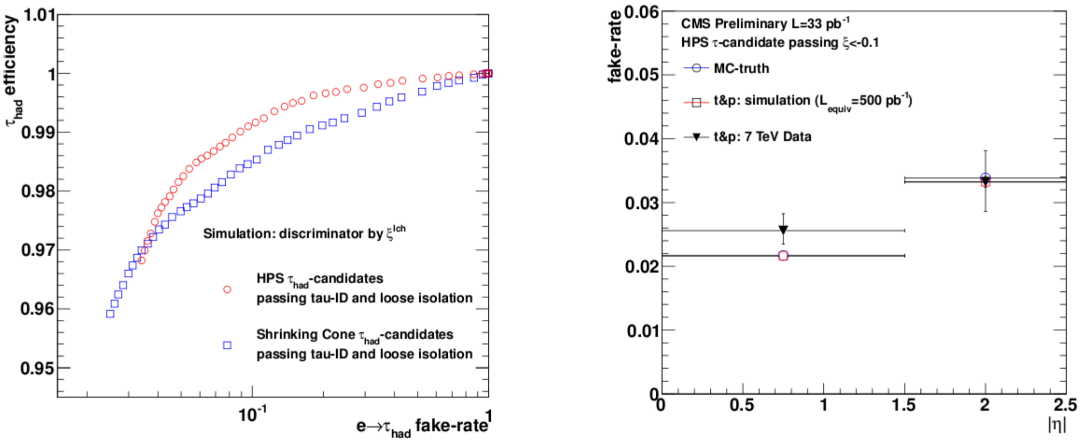
similar algorithms as in Section 5.1 and Section 5.2) is performed, and the appropriate candidates and energy deposits subtracted.

Particle flow jets are reconstructed using an iterative-cone algorithm[55], built from all particle flow types (including electrons and muons, if overlapping). The simplest forms of tau jets are particle flow jets including only particles in a "signal cone" centred around the highest p_T charged hadron (Figure 5.5), with any particles outside the signal cone subsequently used to calculate isolation. Two forms exist:

- *Fixed-cone*, where the signal cone is a fixed size around the leading hadron, usually $\Delta R < 0.1$.
- *Shrinking-cone*, where the signal cone is a function of the jet E_T, usually $\Delta R < \frac{5\ \text{GeV}}{E_T}$, with the cone size constrained between 0.07 and 0.15.

The basic (PF)Tau[56] collections consist of taus built from all possible particle flow jets, without any further identification applied. A large number of discrimination algorithms can then be run to distinguish true tau decays from fakes.

- *Leading track p_T* - due to the small particle multiplicity of tau jets, requiring a 5 GeV leading track eliminates a lot of QCD contribution.



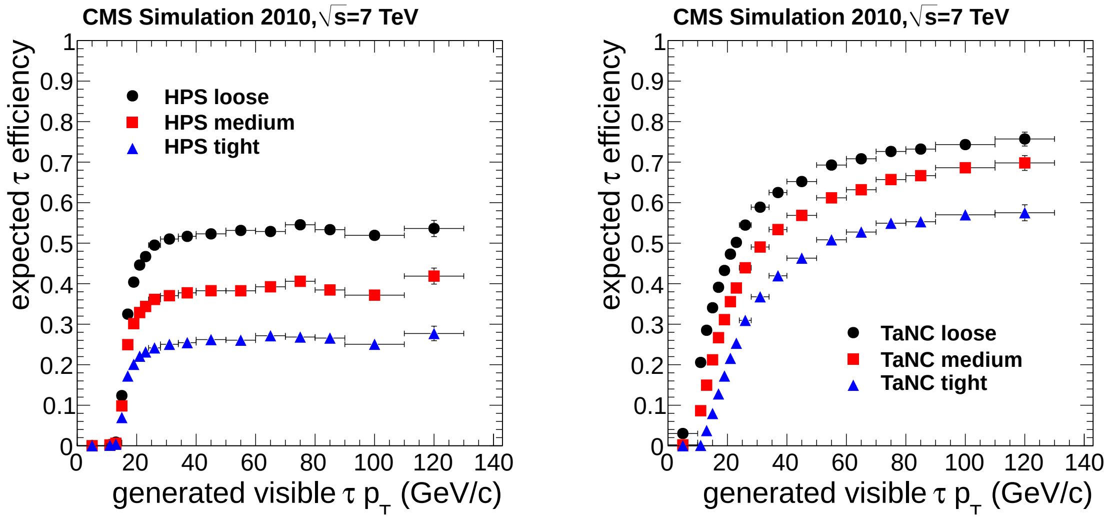
(a) Efficiency vs Fake Rate

(b) Fake Rate vs η

Figure 5.6: $e \to \tau_{had}$ fake rate for the HPS and shrinking cone tau algorithms, as a function of the electron MVA cut (with the used value, $\zeta > -0.1$ at the left) (Figure 5.6(a)) and the fake rate as a function of η for the HPS algorithm, showing the increased fake rate in the endcap region (Figure 5.6(b)). [57].

- *ECAL isolation* - require no particle flow neutral hadrons or photons (outwith the signal cone) with $E_T > 1$ GeV within $\Delta R < 0.5$.
- *Tracker isolation* - require no particle flow charged hadrons (outwith the signal cone) with $p_T > 1.5$ GeV within $\Delta R < 0.5$.
- *Charged track multiplicity* - require the jet contains either one or three charged hadrons (and has unit charge).
- *Against muon* - require that the leading track in the signal cone does not match a global muon track. Although simple, this provides 99% muon rejection for < 1% reduction in tau efficiency.
- *Against electron* - the leading track is tested using the particle flow electron pre-identification algorithm[56], a MVA for distinguishing electron candidates from charged hadrons with approximately 90% electron efficiency and 5% pion efficiency. To improve the electron rejection efficiency, the candidate is then cut based on the E/P[4] and $H_{3\times3}/P$ (sum of 3×3 HCAL towers) fractions, with cuts of $E/P < 0.8$ and $H_{3\times3}/P > 0.15$ respectively for a candidate failing electron pre-identification and 0.95 and 0.05 respectively if passed. This is found[57] to have an efficiency of 97% for taus and 2.4% for electrons, as shown in Figure 5.6.

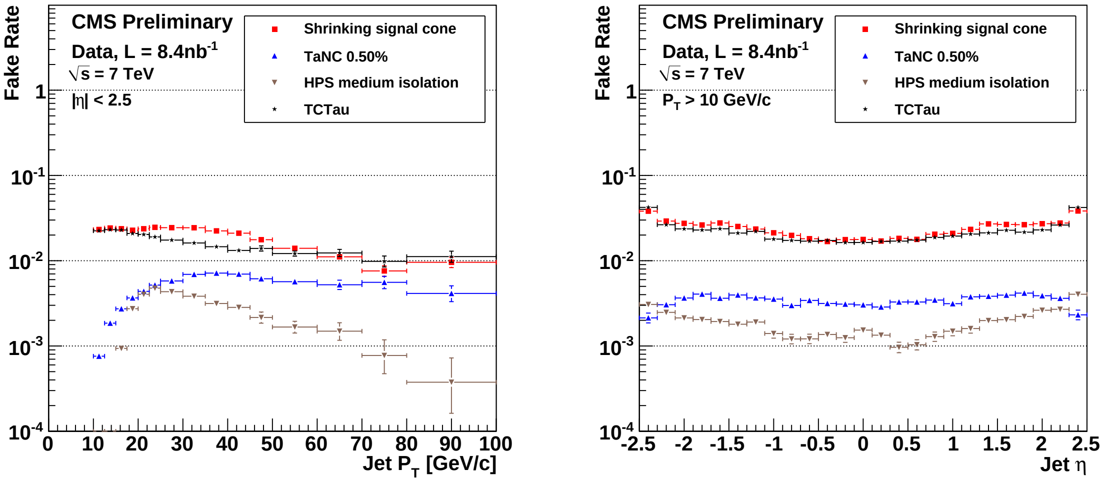
[4] $E \to$ ECAL energy, $P \to$ total track momentum.



(a) HPS

(b) TaNC

Figure 5.7: Tau identification efficiency plots for the HPS and TaNC algorithms, as a function of p_T. [58].



(a) p_T

(b) η

Figure 5.8: Fake rates from light jets for the shrinking cone, HPS (with medium isolation) and TaNC (0.50% working point) algorithms, showing the fake rate as a function of p_T and η in early data (note that the HPS points correspond to isolation as well as HPS reconstruction having been applied. [59].

5.3.2 Hadron Plus Strips

The Hadron Plus Strips (HPS) algorithm[60] is an alternative method for reconstructing taus from particle flow jets, based on examining the individual charged and neutral hadrons (as classified by the particle-flow algorithm) from which the jet is composed, to determine the specific tau decay mode. The efficiency as a function of p_T of this algorithm is shown in Figure 5.7(a), and fake rate in Figure 5.8.

Unlike the circular cones used in the particle flow algorithm, the HPS algorithm takes account of photon conversions (of photons from neutral pion decays), the electrons and positrons from which cause spreading in the ϕ direction of the energy deposit.

The strip finding algorithm starts from the highest energy electromagnetic object in the jet cone, then searches for additional EM objects in a strip with $\Delta\eta = 0.05$ and $\Delta\phi = 0.20$. For each object found, the Lorentz vectors are summed and the strip re-centered on the aggregate centre, until no more can be found. Additional strips are created for any objects in the jet not associated with the first strip, and iterated as above. Finally, those strips with aggregate $p_T > 1$ GeV are retained, and the charged hadrons and strip centres required to fit in an shrinking cone with $\Delta R < \frac{2.8}{p_T}$ (using the p_T of the summed hadrons and strips), constrained between 0.05 and 0.1, and the ΔR between the aggregate direction and the jet axis must be less than 0.1. The selected hadrons and strips are then compared against possible decay modes;

- Hadron only - reconstructs hadron only (11.6%[24]), and possibly hadron plus a single neutral hadron (25.9%), if the neutral hadron is not well reconstructed.
- Hadron + strip - reconstructs hadron plus a single neutral hadron.
- Hadron + strip + strip - reconstructs hadron plus single neutral hadron (decay photons well separated) or hadron plus two neutral hadrons (9.5%).
- Three hadrons - reconstructs three prong tau decays (9.8%), and possibly three pronged with a single neutral hadron (4.8%). The overall charge is required to be $|q| = 1$ and the tracks must converge to a common secondary vertex.

In each case, the sum of all the four-vectors must have a visible mass close to that expected for a tau decay (the exact cut varies per decay mode). For the two-strip mode, the sum of the strip four-vectors must be compatible with the mass of a pion. If there are multiple valid modes, the mode that consumes the maximum fraction of the jet energy is preferred.

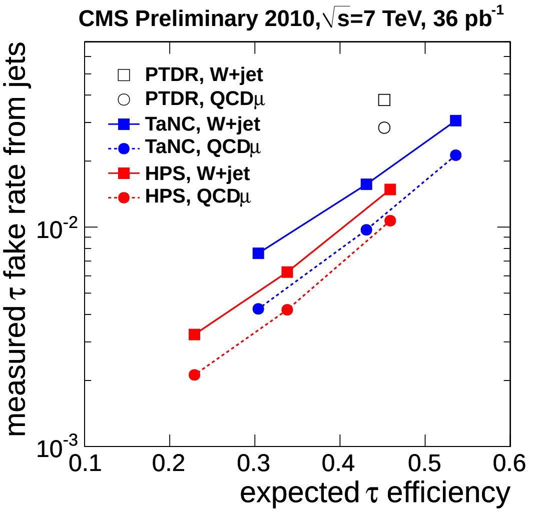


Figure 5.9: Fake rates (from $W + jet$ and QCD events) for the HPS and TaNC algorithms, derived from W and Z events[58]. The points correspond (left-to-right) to "tight", "medium" and "loose" working points. The open points correspond to the expected tau fake rates/efficiencies from the TDR[34], using a shrinking cone algorithm. This shows an approximately three-fold decrease in fake rate for equivalent tau efficiency.

	PF h^+/GeV	PF γ/GeV	Efficiency	Fake Rate
Loose	1.0	1.5	53.6%	1.0%
Medium	0.8	0.8	43.1%	0.4%
Tight	0.5	0.5	30.4%	0.2%

Table 5.2: HPS tau isolation working points, showing the veto thresholds for PF objects in the isolation cone, efficiency for $p_T^\tau > 15$ GeV and fake rate from QCD jets. [58].

Three levels of isolation are defined for use with HPS taus, each applying to an isolation cone $\Delta R < 0.5$. The efficiencies and fake rates for each are shown in Table 5.2 and Figure 5.9.

5.3.3 Tau Neural Classifier

The Tau Neural Classifier (TaNC)[61] is not a tau reconstruction algorithm but rather an identification algorithm which can be applied to taus built either by the cone or HPS tau algorithms. The efficiency is shown in Figure 5.7(b).

The collection of particle flow photons within the tau cone is analysed, and all possible invariant mass pairs constructed. Pairs with $M < 200$ MeV are tagged as π^0 candidates, and those closest to the π^0 mass progressively selected, removing less well reconstructed candidates. Any unpaired photons (carrying at least 10% of the jet E_T) remaining are considered also as neutral pion candidates, in the case that both photons impacted the ECAL too close together to be resolved, or the decay was sufficiently asymmetric that one photon was not resolved.

The set of neutral and charged hadrons are then fed into a neural network, one for each of the five leading decay modes, and for fake-rate working points of 1%, 0.5% and 0.25%. These are trained on Monte-Carlo using $Z \rightarrow \tau\tau$ events as signal and QCD as backgrounds, with samples divided into a training fraction and an independent validation fraction. Events within the samples are weighted to have equal probability distributions in p_T and η, so that these variables can be used as inputs but will not result in training to a a specific (ie, originating from Z) tau spectrum. The neural network uses as inputs a number of topological and kinematic properties of the tau, such as the invariant mass of the signal tracks, p_T of the leading track and the angle and p_T of the Nth charged hadron or π^0 constituent.

At this time, the TaNC algorithm does not perform as well as the HPS algorithm, but is included here as a promising future approach.

5.4 Missing Transverse Energy

The presence of neutrinos (or new physics with equivalent properties) can only be inferred by apparent non-conservation of energy in the transverse plane. Since the actual momentum of the interacting partons in the beam direction is not known, the inference can only be made perpendicular to the beamline, and neither is it possible to determine whether there are one or more weakly-interacting particles involved (except by analysing the visible products).

Several E_T^{miss} algorithms are used by CMS:

- “CaloMET”, the negative vector sum of the ECAL and HCAL towers, with energy scale corrections for jets (type-I) and low-p_T unclustered particles (type-II).

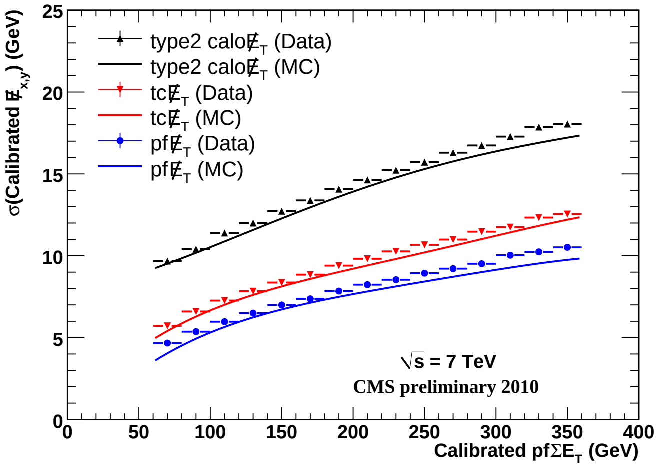


Figure 5.10: Resolution of the three main E_T^{miss} methods, as a function of the scalar sum of E_T in the event, showing the best performance from the PFMET method. [62].

- "TCMET", derived from CaloMET, with the track momentum of identified charged hadrons substituted for the calorimeter deposits, taking advantage of better resolution in the tracker than the HCAL.

- "PFMET", the negative vector sum of all particle flow candidates, reconstructed as per Section 5.3.1.

Resolutions for the three methods are shown in Figure 5.10. For the subsequent analysis we use the PFMET algorithm, which generally has the best resolution, both by scale and peak width.

Chapter 6

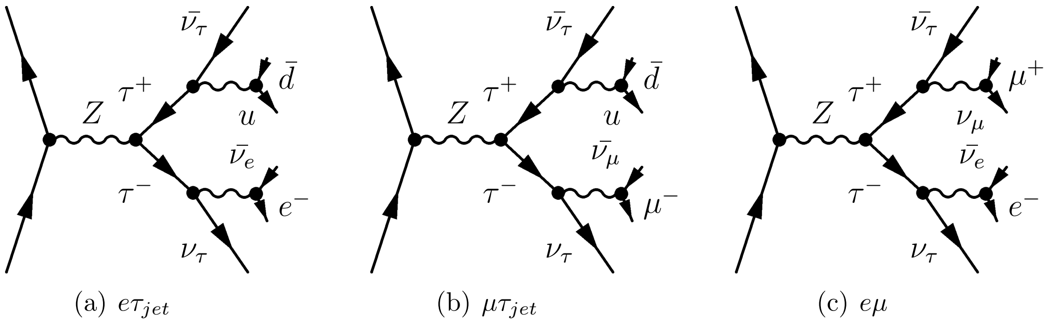
Measuring the $Z \to \tau\tau$ production cross section

6.1 Introduction

The decay modes of tau leptons have been extensively studied at LEP[63], and the Z production cross section measured at CMS using the $Z \to ee$ and $Z \to \mu\mu$ channels[64] with better accuracy than is possible in the $Z \to \tau\tau$ channel, due to "smearing" of the visible mass distribution by between two and four neutrinos in the final state. The value of studying the $Z \to \tau\tau$ channel is as a "standard candle" against which to study tau reconstruction, and as a precursor for light $\Phi \to \tau\tau$ searches.

Of the six possible final states from $Z \to \tau\tau$, we will focus on the $e\tau_{jet}$, $\mu\tau_{jet}$ and $e\mu$ modes (Figure 6.1). The ee and $\mu\mu$ modes suffer both from the smallest branching ratios (3.1% each) and are very difficult to distinguish from $Z \to ee$ and $Z \to \mu\mu$ direct production. The $\tau_{jet}\tau_{jet}$ mode has the largest branching ratio (41.6%), but was not possible to collect during the 2010 running period as Level 1 tau triggers were not enabled, and the p_T thresholds of the available generic jet triggers eliminated most of the tau spectrum. This channel would also suffer doubly from tau identification uncertainty.

The $e\tau_{jet}$ and $\mu\tau_{jet}$ channels lend themselves to analysis in parallel, as they share common kinematics and (mostly) common backgrounds. The $e\mu$ channel is considered alongside them since, although it requires slightly different analysis, it presents an almost unambiguous final state.



(a) $e\tau_{jet}$ (b) $\mu\tau_{jet}$ (c) $e\mu$

Figure 6.1: The $e\tau_{jet}$, $\mu\tau_{jet}$ and $e\mu$ production and decay modes.

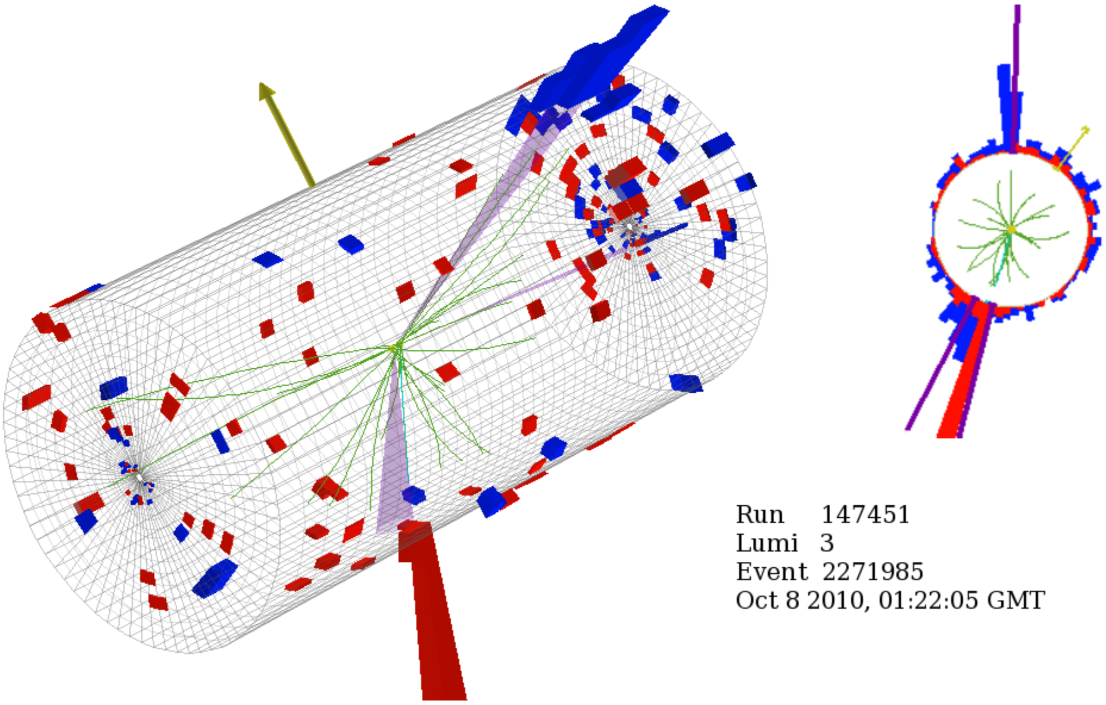


Figure 6.2: $e\tau_{jet}$ candidate event from run 147451. The event contains a 23.3 GeV electron (teal) and 29.6 GeV tau jet (purple), with visible invariant mass 64.9 GeV.

For the three channels combined, 820 candidate events were found in the 2010 dataset, of which around 470 are expected to be actual $Z \rightarrow \tau\tau$ events. Event displays of candidates for each channel are shown in Figure 6.2, 6.3 and 6.4.

While the size of the 2010 dataset (36.0 pb^{-1}) means that this analysis is statistically limited, the p_T spectrum of $Z \rightarrow \tau\tau$ decay products (Figure 6.5) is concentrated at low p_T and so the low-threshold triggers available in early running result in a high acceptance. Higher thresholds would result in significantly lower acceptance and doubling the current thresholds will reduce $Z \rightarrow \tau\tau$ acceptance by approximately a factor of ten.

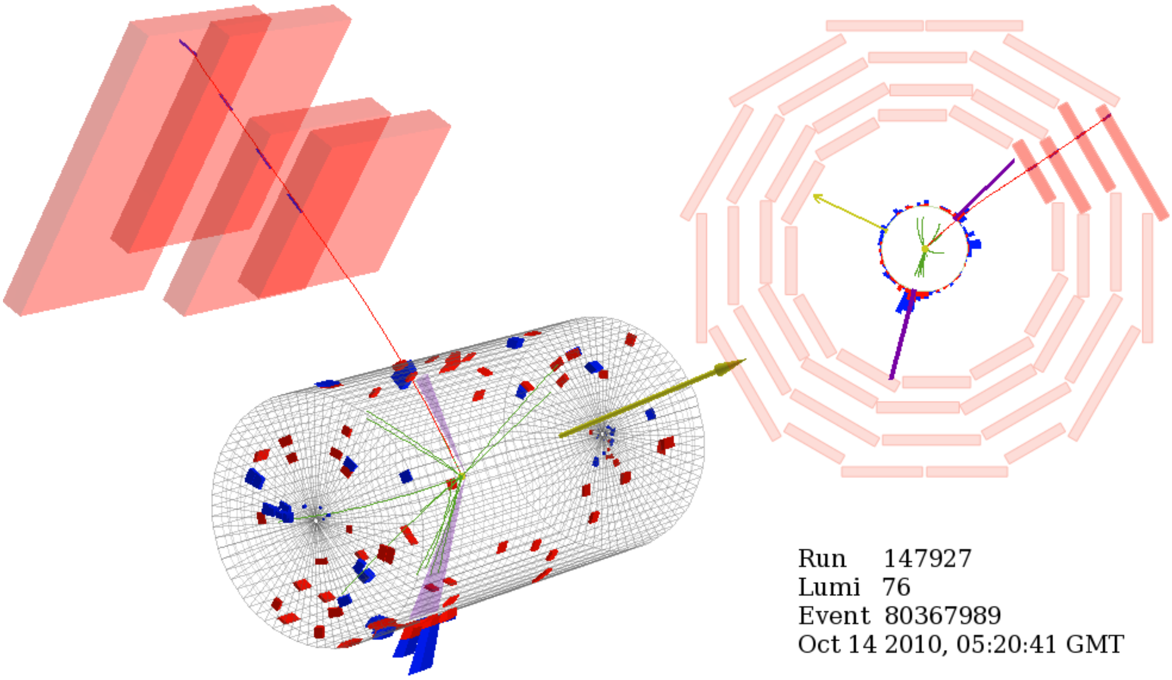


Figure 6.3: $\mu\tau_{jet}$ candidate event from run 147927. The event contains a 19.6 GeV muon (red) and 21.7 GeV tau jet (purple), with visible invariant mass 48.5 GeV. This event contains a relatively large (18.9 GeV) E_T^{miss} vector for a $Z \to \tau\tau$ event.

6.1.1 Simulated and Real Datasets

The Monte Carlo (MC) datasets covering the signal and expected background processes are listed in Table 6.1. All of the MC samples listed were generated using PYTHIA6.4[65] with Tune Z2[66] and originate from the *FALL10* MC production campaign[1]. Datasets including tau decays are additionally processed with the *tauola*[67] package to properly handle tau polarisation and decay. Pile-up of additional minimum bias events is added to events according to the observed distribution[2] in early data.

The simulated traversal of the generator-level particles through the CMS detector was performed using the *GEANT4*[69] package. Triggering and reconstruction of the simulated detector output was then performed using *CMSSW* 3.8.5 (Section 4.4).

For the analysis, the MC has been scaled to an integrated luminosity of the data samples (36.0 pb^{-1}). All MC datasets, except for the lower p_T bins of QCD and $\gamma + jets$ samples simulate a greater integrated luminosity than this.

[1] Although some are reprocessings of existing generator-level samples from *SPRING10* and *SUMMER10*.
[2] On average, 2.8 extra minimum-bias vertices per event [68].

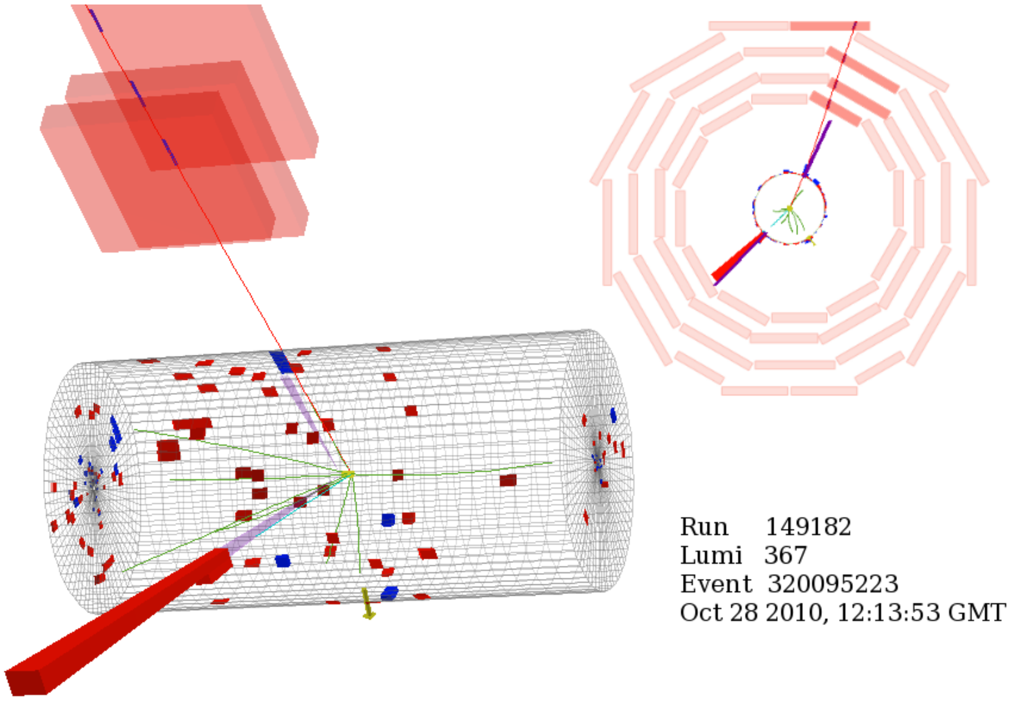
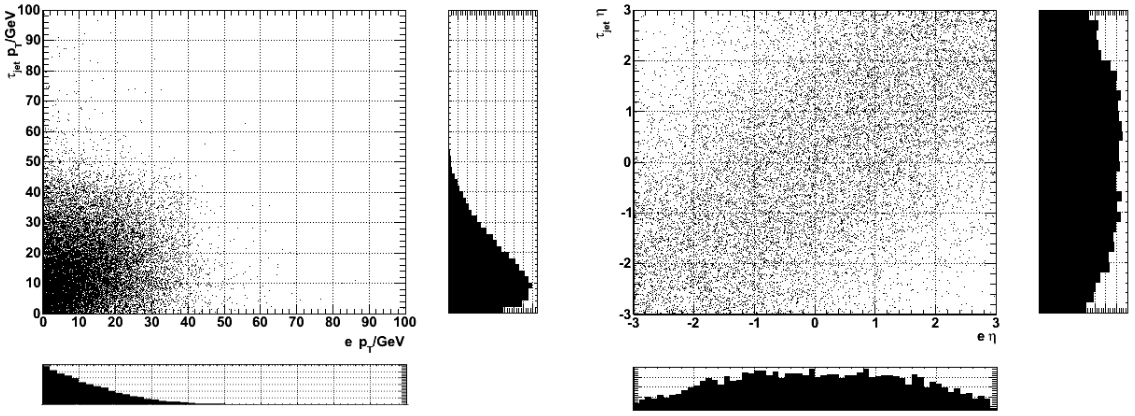


Figure 6.4: $e\mu$ candidate event from run 149182. The event contains a 37.6 GeV electron (teal) and 32.0 GeV muon (red), with visible invariant mass 68.4 GeV. Both leptons have been reconstructed as tau candidates, but neither tau passes lepton rejection.



(a) $e\tau_{jet}$ p_T

(b) $e\tau_{jet}$ η

Figure 6.5: Generator level p_T, η distributions for the $Z \to \tau\tau \to e\tau_{jet}$ channel, generated with PYTHIA6. Subplots are projections of the two-dimensional histograms. The spectrum for the $\mu\tau_{jet}$ channel is very similar.

The enriched QCD samples are subsets of general QCD dijet production, with a filter applied at the generator-level. The muon-enriched sample only requires a generator-level muon with $p_T > 15$ GeV and $|\eta| < 2.5$. The EM-enriched samples require a generator-level electron or photon (divided by decay from light or heavy flavour) with $|\eta| < 2.4$, which is then required to be loosely isolated. The approximate ECAL and tracker isolation values are calculated (from the generator-level particles only, not the radiation simulation), requiring that $\sum p_T < 5$ GeV for charged particles and $\sum E_T < 10$ GeV for neutral ones, in a cone of $\Delta R < 0.2$. In addition, $\frac{E_{had}}{E_{EM}} < 0.5$ is required within the same cone.

Real data for this analysis comes from the Run2010A and Run2010B eras, and are tabulated in Table 6.2. The data is from the November 2010 reprocessing of the complete dataset with *CMSSW* 3.8.6. The *EG*[3] and *Electron* datasets are used for the $e\tau_{jet}$ channel, and the *Mu* dataset for the $e\mu$ and $\mu\tau_{jet}$ channels.

6.1.2 Triggering

The "online" trigger table was changed repeatedly during the Run2010A and Run2010B eras, in order to maintain manageable HLT rates as the instantaneous luminosity increased[4]. The lowest p_T unprescaled single-lepton trigger path was used for each channel, although for the $e\tau_{jet}$ and $\mu\tau_{jet}$ channels lepton-tau cross triggers became available later on, which allowed a lower p_T threshold than the equivalent single lepton trigger. The triggers used are summarised in Table 6.3.

Comparison to Monte-Carlo is complicated by the fact that the MC trigger table corresponds to a snapshot of the data-taking table early in Run2010B, and hence the MC and overall data trigger efficiencies are different, due to the changing p_T thresholds and the changing ID and isolation requirements of the electron trigger.

[3] **E**lectron, **G**amma; these were merged for Run2010A and separated into *Electron* and *Photon* for Run2010B.

[4] From $3\mu b^{-1}s^{-1}$ at the beginning of Run2010A to $200\mu b^{-1}s^{-1}$ at the end of Run2010B[31].

Name	Subset	Events/10^6	σ/pb	Lumi/pb^{-1}	Channels
$Z/\gamma* \to ee$	$M_{Z/\gamma*} > 20$	1.9	1666	1130	$e\tau_{jet}$, $e\mu$
$Z/\gamma* \to \mu\mu$	$M_{Z/\gamma*} > 20$	2.2	1666	1340	$e\mu$, $\mu\tau_{jet}$
$Z/\gamma* \to \tau\tau$	$M_{Z/\gamma*} > 20$	2.0	1666	1200	all
$W \to e\nu_e$		3.7	10438	350	$e\tau_{jet}$, $e\mu$
$W \to \mu\nu_\mu$		5.2	10438	500	$e\mu$, $\mu\tau_{jet}$
$W \to \tau\nu_\tau$		4.3	10438	410	$e\tau_{jet}$, $e\mu$
WW		2.1	43.0	48000	all
WZ		2.2	18.2	1.21×10^5	all
ZZ		4.2	5.90	7.16×10^5	all
$\gamma + jet$	$0 < p_T < 15$	1.0	8.40×10^7	0.012	$e\tau_{jet}$, $e\mu$
	$15 < p_T < 30$	1.0	1.72×10^5	5.97	$e\tau_{jet}$, $e\mu$
	$30 < p_T < 50$	1.0	16700	61.4	$e\tau_{jet}$, $e\mu$
	$50 < p_T < 80$	1.0	2720	376	$e\tau_{jet}$, $e\mu$
	$80 < p_T < 120$	1.0	447	2340	$e\tau_{jet}$, $e\mu$
	$120 < p_T < 170$	1.0	84.2	12200	$e\tau_{jet}$, $e\mu$
	$170 < p_T < 300$	1.1	22.6	48500	$e\tau_{jet}$, $e\mu$
QCD Muon-enriched		12	84700	142	$e\mu$, $\mu\tau_{jet}$
QCD EM-enriched (bc)	$20 < p_T < 30$	2.2	1.32×10^5	16.8	$e\tau_{jet}$, $e\mu$
	$30 < p_T < 80$	2.0	1.37×10^5	14.5	$e\tau_{jet}$, $e\mu$
	$80 < p_T < 170$	1.0	9360	110	$e\tau_{jet}$, $e\mu$
QCD EM-enriched (uds)	$20 < p_T < 30$	37	2.45×10^6	15.0	$e\tau_{jet}$, $e\mu$
	$30 < p_T < 80$	42	3.87×10^6	10.8	$e\tau_{jet}$, $e\mu$
	$80 < p_T < 170$	7.3	1.40×10^5	52.1	$e\tau_{jet}$, $e\mu$
$t\bar{t}$		1.0	158	6230	all
Total		140			

Table 6.1: Monte-Carlo datasets used in the analysis. Cross-sections given include generator-level selection efficiencies (eg, the QCD enriching filters), where relevant. Cross sections are NLO except for the QCD samples, which are LO. The p_T values given for the $\gamma + jet$ and QCD datasets correspond to the photon and leading jet, respectively.

Name	Era	Events/10^6	$\int \ell/\mathrm{pb}^{-1}$	Channels
EG	2010A	53	3.2	$e\tau_{jet}$
Electron	2010B	33	3.2	$e\tau_{jet}$
Mu	2010A	52	32.8	$e\mu$, $\mu\tau_{jet}$
Mu	2010B	33	32.8	$e\mu$, $\mu\tau_{jet}$
Total		171	36.0	

Table 6.2: Real datasets used in the analyses. Each consists of a logical OR of any single-lepton, di-lepton or cross-channel triggers for the appropriate lepton available during the run era.

Trigger	Channel
`HLT_Mu11`	$e\mu$, $\mu\tau_{jet}$ (MC)
`HLT_Mu15`	$e\mu$, $\mu\tau_{jet}$
`HLT_Mu11_PFTau15`	$\mu\tau_{jet}$
`HLT_Ele10_LW_EleId_L1R`	$e\tau_{jet}$
`HLT_Ele10_SW_EleId_L1R`	$e\tau_{jet}$
`HLT_Ele12_SW_TightEleIdIsol_L1R`	$e\tau_{jet}$ (MC)
`HLT_Ele12_SW_TighterEleIdIsol_L1R`	$e\tau_{jet}$
`HLT_IsoEle12_PFTau15`	$e\tau_{jet}$

Table 6.3: Electron and muon triggers used to select events for the analysis, and the MC trigger used. “L1R” in the electron trigger names denotes the Level 1 Regional Calorimetric Trigger, and “LW” and “SW” large and small windows respectively used for pixel track reconstruction.

The muon triggers were all derived from a 7 GeV L1 muon seed[5], and the electron triggers from a 8 GeV L1 electromagnetic seed[6] (except for `HLT_Ele10_LW_EleId_L1R`, which used a 5 GeV[7] seed).

6.1.3 Software

To allow for repeated, fast analysis passes across the whole dataset, the original CMS RECO-level data[8] was skimmed using CMSSW 3.8.7 into a compact form[9]. The skimming converted the deeply nested and cross-linked CMS data format (serialised C++ classes) into a flattened structure of POD[10] types, consisting of (possibly multidimensional) floating point or integer arrays per event, stored in a ROOT tree structure.

Events were selected from runs in which the detector was fully functional[11], they passed a logical OR of all the leading triggers (as per Table 6.3), contained a reconstructed vertex and contained either an electron or muon. The skimming cuts for electrons, muons and taus are given in Table 6.4. Supplemental information such as event and run numbers, trigger results, E_T^{miss} and generator-level particles[12] were also stored in the skimmed format.

The resulting flat tuples were then analysed with a python selection and plotting framework, described in Appendix A.2. The CPU time required for a complete analysis pass was reduced to $O(hour)$ instead of $O(year)$.

6.1.4 Backgrounds

The principal backgrounds to the analysis are:

- QCD multijet events containing a real electron or muon fake $e\tau_{jet}$ or $\mu\tau_{jet}$ if a jet is mistagged as a hadronic tau. This background has very low efficiency but

[5] `L1_SingleMu7`

[6] `L1_SingleEG8`

[7] `L1_SingleEG5`

[8] $400 KiB/event, \;\; 1.1 PiB$ total

[9] $1.5 KiB/event, \;\; 120 GiB$ total

[10] **P**lain **O**ld **D**ata, ie basic C data types.

[11] Other analyses have been able to recover somewhat larger fraction of the ≈ 45 pb^{-1} 2010 data by only requiring subdetectors of interest to them to be active, but the analysis presented here requires all of the major detector components.

[12] Regrettably, this information was missing from some samples.

Particle	Type	Cuts
Electron	GSF (Section 5.2)	$p_T > 10$ GeV $\lvert\eta\rvert < 2.4$ Not pointing at any ECAL gap Passes WP95 Electron ID
Muon	Global (Section 5.1)	$p_T > 10$ GeV $\lvert\eta\rvert < 2.4$ Track $\chi^2 < 10$ and $n_{hits} > 10$
Tau	HPS (Section 5.3)	$p_T > 15$ GeV $\lvert\eta\rvert < 2.4$ Leading track $p_T > 5$ GeV

Table 6.4: Selection cuts used for the data/MC skim.

contributes due to a cross section ≈ 1000 times in excess of $Z \to \tau\tau$. It is rejected by requiring isolated leptons, since leptons from in-flight meson decays are usually within or near their parent jet, and requiring that lepton and tau not overlap, since the lepton often becomes the leading track of a reconstructed tau jet along with the rest of the parent jet.

- $W \to l(= e, \mu, \tau)\nu + jet$ events fake $e\tau_{jet}$ or $\mu\tau_{jet}$ where the recoil jet is mistagged as a hadronic tau. The single neutrino produced by the W decay gives these events a characteristic transverse mass spectrum which can be used for rejection.

- $Z \to ll(= e, \mu) + jet$ events fake $e\tau_{jet}$ or $\mu\tau_{jet}$ where either the second lepton or the recoil jet is mistagged as a hadronic tau. These are rejected by applying electron and muon discrimination methods to the hadronic tau, and checking for the presence of a second lepton if the fake tau arises from the recoil jet. They can also be distinguished by a much narrower visible mass distribution, since there are no neutrinos in the final state.

- $\gamma + jet$ events fake $e\tau_{jet}$ if the conversion pair is skewed to one hard electron (and the partner is not reconstructed or lost entirely), and the recoil jet is mistagged as a hadronic tau. These events are detected using electron conversion rejection methods to detect the displaced vertex and partner track.

- $t\bar{t}$ events fake any of the channels since the final states can include multiple prompt leptons and multiple mistaggable jets, eg $t\bar{t} \to e\nu_e\mu\nu_\mu b\bar{b}$, which could fake any of the channels. This is a relatively small background, which if necessary can be rejected by tagging one or more b-jets.
- Di-boson (WW, WZ, ZZ) events have a very low cross section (and as such are largely insignificant compared to other backgrounds for $e\tau_{jet}$, $\mu\tau_{jet}$) but are a concern for the $e\mu$ channel, eg $WZ \to ee\mu\nu_\mu$. Depending on the vector bosons involved and resulting topology, these may be rejected by the methods for single W or Z events, and are in any case a very small background.

6.2 Selections

The selections for the three channels are the result of a somewhat convoluted evolutionary process, attempting to maximise the $S/\sqrt{B}$ ratio while (where possible) maintaining commonality between the channels to reduce the number of special cases required, and commonality with other analyses to avoid the need to remeasure already studied constants and systematic errors.

All events at this stage have already implicitly passed the skimming cuts in Table 6.4. The Monte-Carlo predicted event yields are shown in Table 6.5.

6.2.1 $e\tau_{jet}$ and $\mu\tau_{jet}$

The $e\tau_{jet}$ and $\mu\tau_{jet}$ channels have common kinematics and may, in the abstract, be treated similarly. While the details of the lepton identification differ, the selections are otherwise similar.

The basic kinematic requirements are that the event contains at least one lepton of the appropriate flavour, with $p_T > 15$ GeV and inside the tracker, ECAL and muon system acceptances ($|\eta| < 2.4$), and at least one hadronic tau candidate (HPS tau) with $p_T > 18$ GeV and $|\eta| < 2.4$. Taus are additionally required to fall outside the ECAL barrel-endcap gap, $1.442 < |\eta| > 1.566$[13]. These p_T thresholds are the lowest possible

[13] This cut is made on electrons at the reconstruction level, and is irrelevant for muons.

	$e\tau_{jet}$	$\mu\tau_{jet}$	$e\mu$
Observed	378	352	91
$Z \to \tau\tau$	146.5 ± 2.1	232.5 ± 2.6	89.7 ± 1.6
$Z \to ll$	42.6 ± 1.2	15.1 ± 0.6	2.6 ± 0.3
$W \to l\nu_l$	36.8 ± 2.0	57.6 ± 2.3	2.4 ± 0.4
QCD	100.2 ± 16.9	40.2 ± 3.2	1.8 ± 0.7
$\gamma + jet$	34.0 ± 10.8	–	–
$t\bar{t}$	1.8 ± 0.1	2.6 ± 0.1	5.9 ± 0.2
Di-boson	0.7 ± 0.0	1.0 ± 0.0	2.3 ± 0.0
Total background	216.0 ± 20.1	116.6 ± 4.5	15.0 ± 1.9
Total MC	362.6 ± 20.3	349.1 ± 4.7	104.7 ± 1.9
Expected S/B	40.4%	66.6%	85.7%
Expected $S/\sqrt{B}$	10.0	21.5	23.2

Table 6.5: Event yields for each channel, and the expected composition from Monte-Carlo (scaled for data luminosity 36.0 pb^{-1}), along with the MC statistical errors.

with the available triggers, and also represent the lower limits for which the electron and tau identification algorithms provide useful efficiency (Section 5.1 and 5.2).

Electrons are required to pass "WP80" identification and conversion rejection cuts (Section 5.2.3). No additional identification requirement is placed on muons (which have already been required to be *global* and pass track quality cuts at the skimming stage, sufficient for $> 99\%$ purity).

Tau identification consists of requiring that a valid decay mode was found by the HPS algorithm (Section 5.3.2). Discrimination is performed against electrons and muons, as per Section 5.3.1. An additional lepton-rejection cut is made, requiring $\frac{E_{HCAL}}{p_T^{lead}} > 0.1$, which rejects both electrons and muons and is applied more tightly than the HPS tau discriminator.

Lepton isolation is calculated using particle flow (Section 5.3.1) charged hadron, neutral hadron and photon candidates, rather than calorimeter deposits. The relative isolation variable is calculated using particle flow candidates in $\Delta R < 0.4$[14] (not matching

[14] Measured against the inner momentum state of charged hadron candidates.

the source lepton) as

$$I^{PF}_{rel} = \frac{\sum_{h^+}^{p_T > 1.5\text{ GeV}} p_T + \sum_{h^0}^{E_T > 1\text{ GeV}} E_T + \sum_{\gamma}^{E_T > 1\text{ GeV}} E_T}{p_T^{lepton}} < 0.1$$

where h^+ refers to charged hadrons and h^0 to neutral hadrons. Tau isolation is also particle flow based and relative, but is precalculated as part of the reconstruction. Taus are required to pass the "loose" HPS isolation discriminator (Section 5.3.2).

All possible opposite-charge lepton-tau pairs (with $\Delta R(l, \tau) > 0.5$) are then constructed, and a cut placed on the transverse mass[15] between the lepton and the E_T^{miss}[16], $M_T(l, E_T^{miss}) < 40$ GeV. We require that there can be only one.

The final mass distribution still includes a significant $Z \to ll$ contribution[17]. To reduce this, a "loose" lepton collection is created, requiring in each case $p_T > 10$ GeV, $|\eta| < 2.4$ and in the electron case, passing the "WP95" identification. Events containing two loose leptons are rejected.

To add further Z rejection, if there is only one lepton and no additional loose lepton is found, all possible lepton-track pairs are built (from the general Kalman-filter track collection, with $p_T > 5$ GeV and not matching any reconstructed lepton), and the event rejected if any $|M_{l,trk} - M_Z| < 5$ GeV. The small mass window exploits the narrow $Z \to ll$ mass shape compared to that of $Z \to \tau\tau$. This serves to catch the small tail of electrons that fail WP95 identification, and electrons pointing into ECAL gaps which are not reconstructed. Combined with the loose lepton veto, $Z \to ll$ contribution is reduced by a factor of three in the $\mu\tau_{jet}$ channel and a factor of two in the $e\tau_{jet}$ channel[18], while the efficiency for signal events to pass the Z rejection cuts is in each case $\approx 97\%$.

The visible mass distributions obtained with these selections are shown in Figure 6.6.

6.2.2 $e\mu$

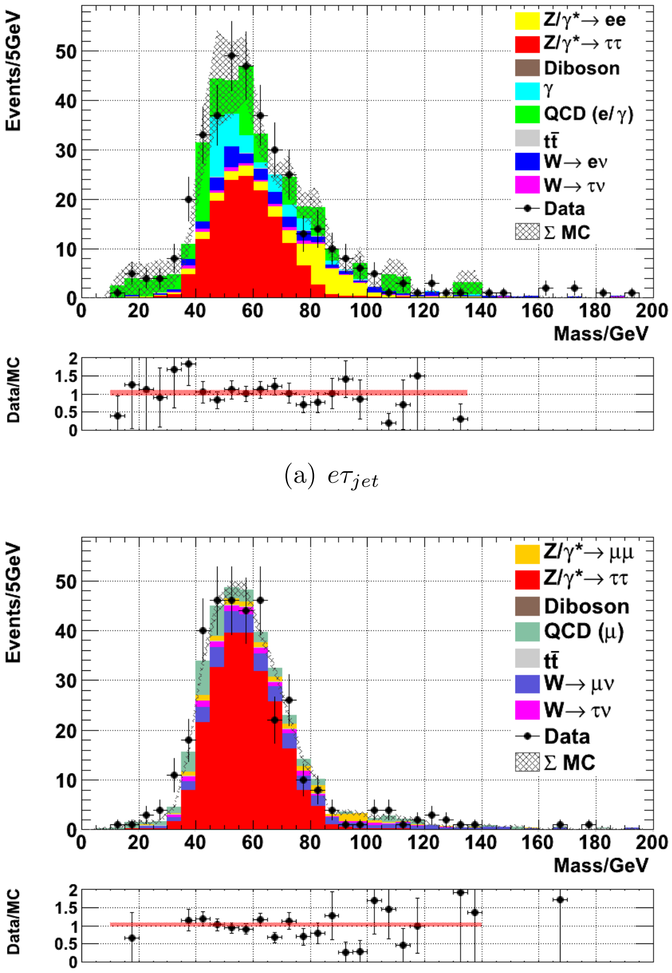
The $e\mu$ channel benefits from a final state with less ambiguity than the hadronic channels, which reduces the number of rejection steps necessary to select a relatively pure sample.

[15] $M_T = \sqrt{2 \cdot p_T(l) E_T^{miss} (1 - \cos\Delta\phi(l, E_T^{miss}))}$

[16] Using PFMET (Section 5.4).

[17] 12.8% $Z \to ee$ contribution for $e\tau_{jet}$, 12.1% $Z \to \mu\mu$ for $\mu\tau_{jet}$.

[18] 6.7% $Z \to ee$ in $e\tau_{jet}$, 4.1% $Z \to \mu\mu$ in $\mu\tau_{jet}$, proportion of events remaining after $Z \to ll$ rejection cuts.



(a) $e\tau_{jet}$

(b) $\mu\tau_{jet}$

Figure 6.6: Visible mass distributions for the $e\tau_{jet}$ and $\mu\tau_{jet}$ channels, showing good data agreement with Monte-Carlo. The light gray overlay shows the combined total and errors for the MC samples, which are dominated by the QCD and $\gamma + jet$ samples (having smaller simulated integrated luminosity than the data). The $e\tau_{jet}$ channel selects 378 data events, and the $\mu\tau_{jet}$ channel 352. These sample colouring conventions and showing the MC sum are also used hereafter.

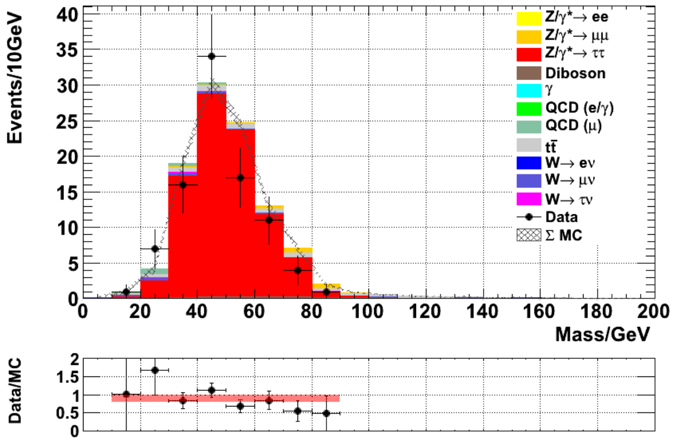


Figure 6.7: Visible mass distribution for the $e\mu$ channel, showing data and Monte-Carlo. The $e\mu$ channel selects 91 data events.

The channel is triggered using the muon leg (which offers less-frequent trigger changes, and better trigger efficiency and purity than the electron leg), and the muon selection is identical to that used for the $\mu\tau_{jet}$ channel (Section 6.2.1). The limited backgrounds to the $e\mu$ channel however allow us to loosen the requirements on the electron leg (and hence increase the acceptance). Since the electron is not required to have passed a trigger threshold it is possible to reduce the p_T cut without risking trigger turn-on effects. The electron leg is required to have $p_T > 10$ GeV, pass the looser "WP85" identification and conversion rejection[19], and looser relative isolation of $I^{PF}_{rel} < 0.3$.

Opposite sign electron-muon pairs with $\Delta R(e, \mu) > 0.5$ are built, and transverse mass requirements $M_T(e, E^{miss}_T) < 50$ GeV and $M_T(\mu, E^{miss}_T) < 50$ GeV imposed. The M_T threshold is raised compared to the lepton-tau channels since we have two well-reconstructed leptons to measure this quantity with, and in any case the intrinsic W background is much reduced compared to that for lepton-tau ($\approx 4\%$ of events prior to the M_T cuts). The event is finally required to only have one pair passing these requirements.

It was found unnecessary to apply the further loose lepton and lepton-track Z rejection cuts, since the background from this process is minimal from the existing requirements ($\approx 2\%$).

The visible mass distributions obtained with this selection are shown in Figure 6.7.

[19] The WP85 conversion rejection part is identical to WP80.

6.3 Background Extraction

Since we cannot hope to select directly a pure sample of $Z \to \tau\tau$ events, we must inevitably take an approach of estimating the signal and background fraction in the selected events. To first order, we can estimate the signal fraction from MC, but this is unsatisfactory due to uncertainties in the PDF, process model, radiation simulation and detector response and, for the QCD and photon backgrounds, statistical limits in the available MC.

For the $e\mu$ channel the total predicted background is ≈ 15 events, of which the largest individual background is ≈ 6 events. The small number of $e\mu$ events mean that template-based methods would be completely impossible and the statistical error on counting methods considerable, and since the QCD contribution is small we expect to model the remaining electroweak backgrounds with reasonable fidelity. Hence, the backgrounds to the $e\mu$ channel (except for $t\bar{t}$) are estimated directly from MC.

Backgrounds for the $e\tau_{jet}$ and $\mu\tau_{jet}$ channels are calculated using a cascade[20] of control regions. The method is applied equally to both channels (except for the addition of a further control region for $\gamma + jet$ in the $e\tau_{jet}$ channel), using efficiencies measured separately for each channel. The small $t\bar{t}$ and di-boson contributions are obtained from MC, having a total of < 2 events per channel.

Except where otherwise noted, the *efficiencies* quoted are calculated using Monte Carlo samples. The efficiency is given by $\epsilon = \frac{N^{final}_{process}}{N^{control}_{all}}$, where $N^{control}_{all}$ is the number of expected Monte Carlo events passing the control region selection (for all processes), and $N^{final}_{process}$ is the number of expected Monte Carlo events in the signal region belonging to the target process.

6.3.1 Z Background

As mentioned in Section 6.1.4, there are two main pathologies for fakes arising from $Z \to ll(= e, \mu)$; in the first, the fake tau arises from a recoil jet, and in the second, from the other Drell-Yan lepton.

For the $ll + jet$ mechanism, we define a control region (Figure 6.8) as:

- Select a lepton-tau pair as per the normal selection (Section 6.2.1).

[20] Control regions that themselves contribute to background removal in other control regions.

- Require a second loose lepton[21] in the event, non-collinear with either leg of the lepton-tau pair.
- No cut on lepton-track mass near M_Z.

This mechanism is expected to contribute equally to the opposite charge and same charge final states (since the fake tau charge is uncorrelated to the lepton charge). The values and efficiencies (derived from $Z \to ll$ MC) are shown in Table 6.6. This is the smallest control region used, and a large statistical error from the number of events selected is unfortunately inevitable. The number of events in the signal and same-sign control region are given by

$$N_{Zllj} = N_{Zllj}^{control} \times \epsilon_{Zllj}$$

		$e\tau_{jet}$	$\mu\tau_{jet}$
Events in control region	$N_{Zllj}^{control}$	16	30
Efficiency	ϵ_{Zllj}	0.599 ± 0.038	0.177 ± 0.014
Events in signal and SS regions	N_{Zllj}	9.6 ± 0.6	5.3 ± 0.4

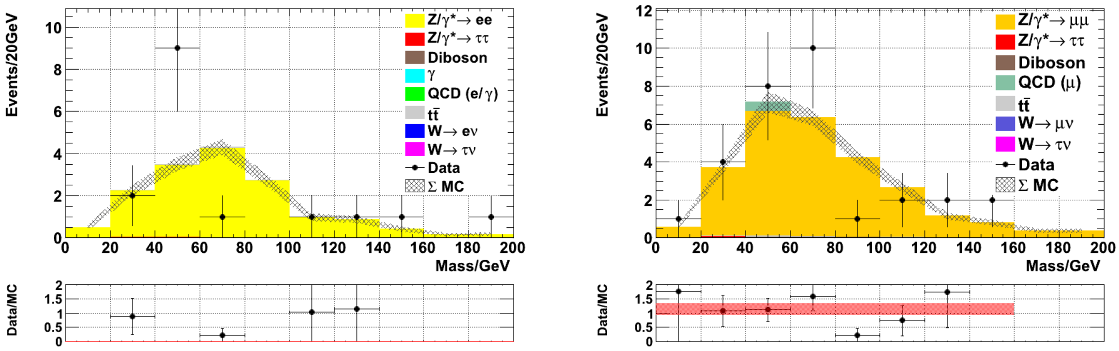
Table 6.6: Observed values for the $Z \to ll + jet$ fake mechanism. Errors shown arise only from MC statistics, errors on the observed number of events and systematic errors on the efficiency are factored in during the fitting procedure. This also applies to the subsequent control region tables.

The control region for the lepton-fakes-tau case (Figure 6.9) is defined by inverting the lepton rejection cuts on the hadronic tau leg, and removing the general $Z \to ll$ rejection cuts, to select a high-purity ($> 99\%$) sample of $Z \to ll$ events:

- Electron rejection MVA (Section 5.3.1) inverted ($\zeta > -0.1$), or muon rejection inverted (tau leading track must match a global muon track), for $e\tau_{jet}$ and $\mu\tau_{jet}$ respectively.
- Tau hadronic fraction cut inverted ($\frac{E_{HCAL}}{p_T^{lead}} < 0.1$).
- No cut on a second loose lepton or lepton-track mass near M_Z.

Since this region contains, with very high purity, actual $Z \to ll$ pairs, the same charge rate is negligible and hence this mechanism only contributes to the opposite-sign region.

[21] Loose lepton as defined in Section 6.2.1.



(a) $e\tau_{jet}$ (b) $\mu\tau_{jet}$

Figure 6.8: Visible mass distributions for the $Z + jet \to ll + jet \to l(l)\tau_{fake}$ control regions, showing the limited statistics available.

The values are shown in Table 6.7. The number of events in the signal region is given by

$$N_{Zl\tau} = N_{Zl\tau}^{control} \times \epsilon_{Zl\tau}$$

		$e\tau_{jet}$	$\mu\tau_{jet}$
Events in control region	$N_{Zl\tau}^{control}$	7977	13411
Efficiency	$\epsilon_{Zl\tau}$	$3.9 \times 10^{-3} \pm 1.4 \times 10^{-4}$	$7.8 \times 10^{-4} \pm 4.8 \times 10^{-5}$
Events in signal region	$N_{Zl\tau}$	31.2 ± 1.1	10.4 ± 0.6

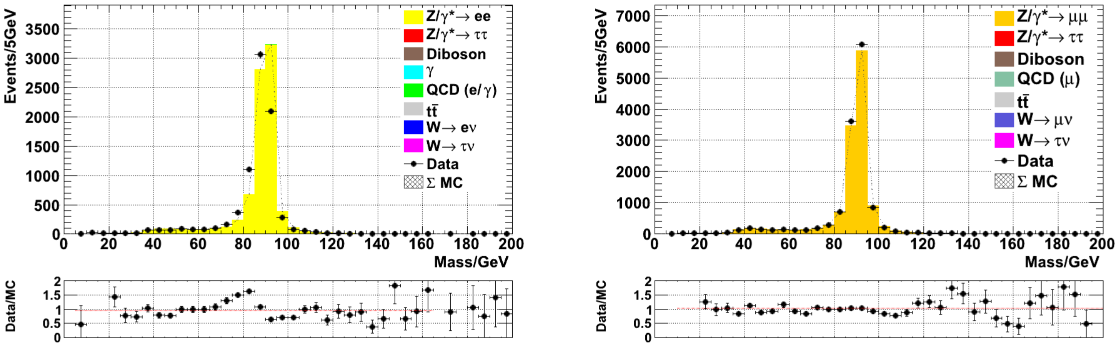
Table 6.7: Observed values for the $Z \to l\tau$ fake mechanism.

6.3.2 W Background

To extract the W background, we exploit the large $M_T(l, E_T^{miss})$ values (mean $M_T^{Wl\nu} \approx$ 70 GeV) resulting from the two-body $W \to l\nu_l$ decay (distribution shown in Figure 6.10). By selecting events with $M_T(l,\tau) > 60$ GeV we obtain a high purity sample of W events[22]. The selected W events are divided by lepton-tau charge into same-charge and opposite-charge regions.

In the opposite-charge region, there is a detectable contribution from $t\bar{t}$ events ($\approx 4\%$), which is subtracted using the estimated contribution from MC. The events in the control

[22] 92% W for $e\tau_{jet}$, 93% W for $\mu\tau_{jet}$, for opposite-charge events.



(a) $e\tau_{jet}$ (b) $\mu\tau_{jet}$

Figure 6.9: Visible mass distributions for the $Z \to ll \to l\tau_{fake}$ control regions. The detectable data-MC shift in the $e\tau_{jet}$ distribution appears to be caused by a mismatch in energy-scale for $e \to \tau$ fakes, not corrected at the time this data was reconstructed. However, in this case the distribution is only being used for counting rather than shape, so the shift is immaterial.

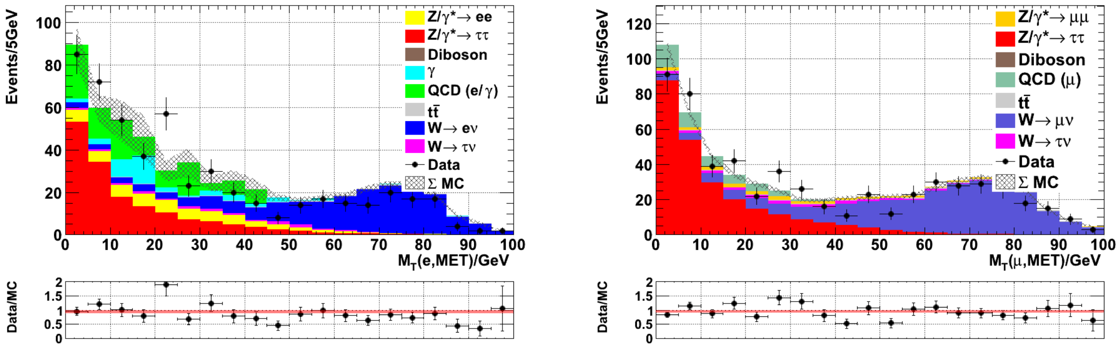
region are used to estimate the sum contribution in the signal region from $W \to l\nu_l$ and $W \to \tau\nu_\tau$, since the latter lacks a distinctive M_T distribution, due to the second neutrino arising from the tau decay vertex, but the relative rate $\frac{W \to \tau\nu_\tau \to l\nu_l\nu_\tau}{W \to l\nu_l}$ is known, and the efficiency can therefore be calculated for both $W \to l\nu_l$ and $W \to \tau\nu_\tau$ contributions to the final state. The contents of the control regions are shown in Figure 6.11 and 6.12, and the values in Table 6.8, including efficiencies calculated from $W \to l\nu_l$ and $W \to \tau\nu_\tau$ MC. The number of events in the signal and same-sign control regions is given by

$$N^{OS}_{Wl\nu} = (N^{control}_{OS,Wl\nu} - N^{control}_{t\bar{t}}(MC)) \times \epsilon_{OS,Wl\nu}$$

$$N^{SS}_{Wl\nu} = N^{control}_{SS,Wl\nu} \times \epsilon_{SS,Wl\nu}$$

6.3.3 γ Background

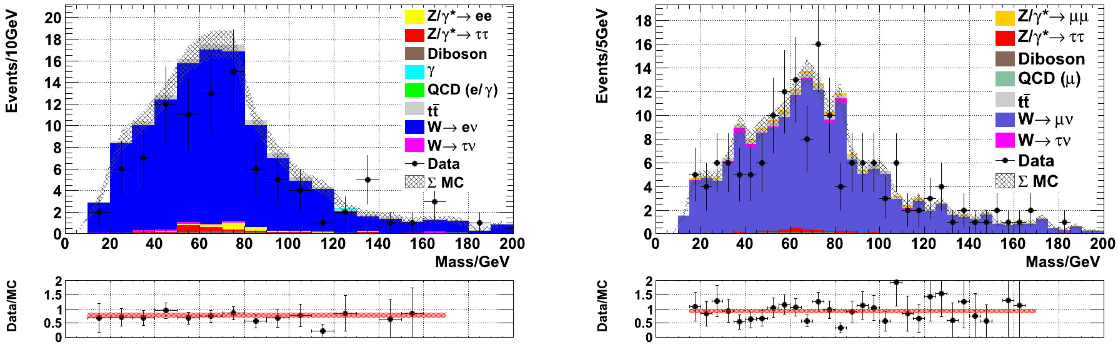
The $\gamma + jet$ background is a large source of uncertainty for the $e\tau_{jet}$ channel (but irrelevant to all the others), due in part to very limited MC statistics. Normally this process would be considered as part of the QCD background, but it is desirable if possible to deal with it separately, since the method used for the QCD estimation uses anti-isolated regions containing almost no $\gamma + jet$ contribution, and thus it is somewhat unreasonable to



(a) $e\tau_{jet}$

(b) $\mu\tau_{jet}$

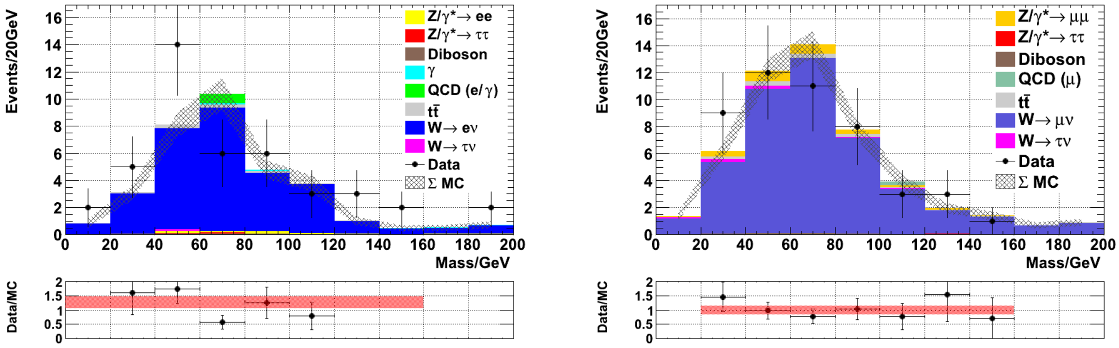
Figure 6.10: $M_T(l, E_T^{miss})$ distributions for $e\tau_{jet}$ and $\mu\tau_{jet}$, showing the high purity $W \to l\nu_l$ region where $M_T(l, E_T^{miss}) > 60$ GeV. (The cut on signal events is $M_T(l, E_T^{miss}) < 40$ GeV).



(a) $e\tau_{jet}$

(b) $\mu\tau_{jet}$

Figure 6.11: Visible mass distributions for the opposite-sign $W \to l\nu_l$ control region.



(a) $e\tau_{jet}$

(b) $\mu\tau_{jet}$

Figure 6.12: Visible mass distributions for the same-sign $W \to l\nu_l$ control region.

		$e\tau_{jet}$	$\mu\tau_{jet}$
Events in OS $W \to l\nu_l$ control region	$N^{control}_{OS,Wl\nu}$	95	160
MC $t\bar{t}$ events	$N^{control}_{OS,t\bar{t}}$	1.75	6.45
Efficiency	$\epsilon_{OS,Wl\nu}$	0.322 ± 0.017	0.345 ± 0.012
Events in signal region	$N^{OS}_{Wl\nu}$	30.0 ± 1.6	53.0 ± 1.8
Events in SS $W \to l\nu_l$ control region	$N^{control}_{SS,Wl\nu}$	43	47
Efficiency	$\epsilon_{SS,Wl\nu}$	0.360 ± 0.033	0.357 ± 0.024
Events in SS control region	$N^{SS}_{Wl\nu}$	15.5 ± 1.4	16.8 ± 1.1

Table 6.8: Observed values for the $W \to l\nu_l$ control regions.

extrapolate from anti-isolated pure QCD to the $QCD + \gamma$ combination in the isolated regions.

We can obtain a moderately pure ($\approx 75\%$) sample of $\gamma + jet$ events (in the case of one hard conversion electron) by inverting the conversion rejection cuts on the electron, and requiring a near back-to-back electron-tau topology (to reject QCD multijet events). The control region is defined by:

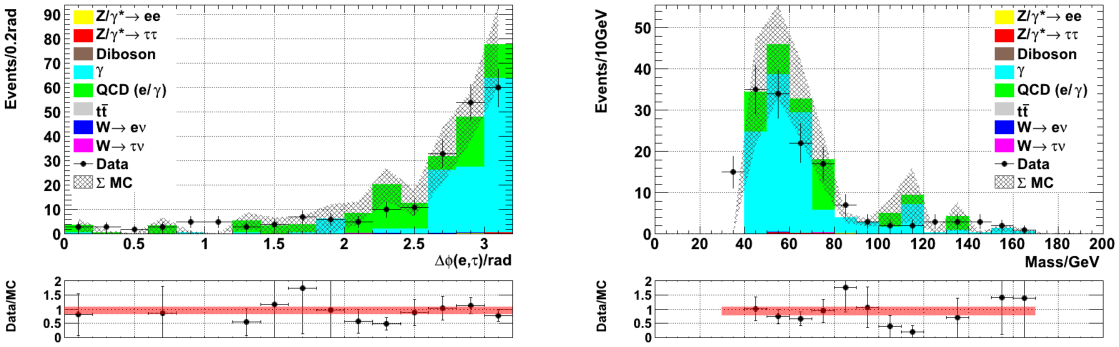
- Electron leaves no hit in the innermost pixel layer
- Electron has a companion track with $|cot(\theta_e) - cot(\theta_{track})| < 0.05$
- $\Delta\phi(e, \tau_{jet)} > 2.5$
- No cut on $q_e + q_\tau$ or $M_T(e, E_T^{miss})$
- No cut on a second loose electron or electron-track mass near M_Z

No cut on pair charge is applied[23], we expect the charge distribution of the fake tau to be random[24], and the size of the control region becomes unfeasibly small selecting only OS or SS events. Hence the overall region is used to estimate the $\gamma + jet$ contribution to both the OS and SS signal regions, with a separate MC-measured efficiency for each. The number of events in the signal and same-sign control regions is given by

$$N^{OS}_{\gamma} = N^{control}_{\gamma} \times \epsilon_{OS,\gamma}$$

[23] Compare with the W control regions, which we separate by pair charge; there we have a subtractable background only contributing to the OS region, whereas here the QCD contamination should also be randomly distributed by charge.

[24] Except for the rare case where the tau is the conversion partner, rather than a jet.



(a) $\Delta\phi(e,\tau)$

(b) Visible Mass $\Delta\phi(e,\tau) > 2.5$

Figure 6.13: (a) Distribution of $\Delta\phi(e,\tau)$ showing the enrichment of $\frac{\gamma}{QCD}$ possible by requiring a near back-to-back pair, and (b) Visible mass distribution of the resulting control region.

$$N_\gamma^{SS} = N_\gamma^{control} \times \epsilon_{SS,\gamma}$$

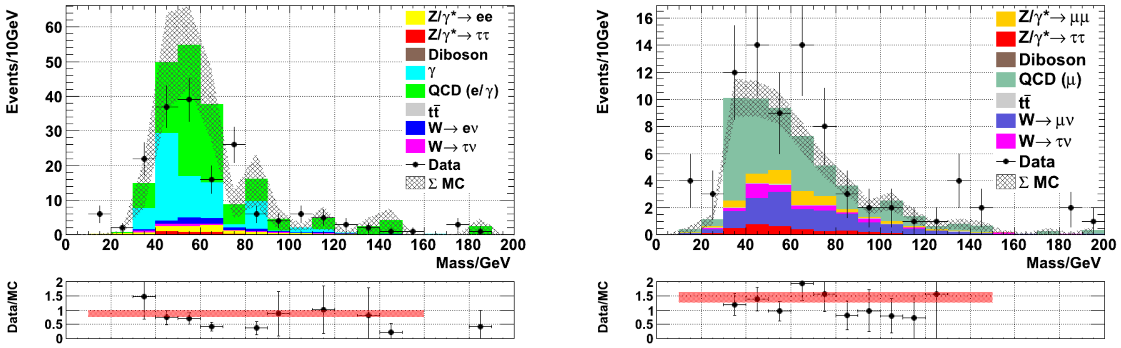
The distributions are shown in Figure 6.13, and the observed values and MC-derived efficiencies are shown in Table 6.9.

		$e\tau_{jet}$
Events in control region	$N_\gamma^{control}$	149
OS efficiency	$\epsilon_{OS,\gamma}$	0.211 ± 0.028
SS efficiency	$\epsilon_{SS,\gamma}$	0.408 ± 0.049
Events in signal region	N_γ^{OS}	31.4 ± 4.2
Events in SS control region	N_γ^{SS}	60.8 ± 7.3

Table 6.9: Observed values for the $\gamma + jet$ control region.

6.3.4 QCD Background

QCD is the largest background for either of the lepton-tau channels, and the most difficult to estimate accurately. By applying the full lepton-tau selection but requiring the lepton and tau to have the same charge, we can select a region enriched in QCD. While this region is largely free of $Z \to \tau\tau$ and $Z \to ll$ contributions, there is still significant contamination from W processes, and $\gamma + jet$ in the $e\tau_{jet}$ channel. The same-charge



(a) $e\tau_{jet}$ (b) $\mu\tau_{jet}$

Figure 6.14: Distributions for the isolated, same charge control regions.

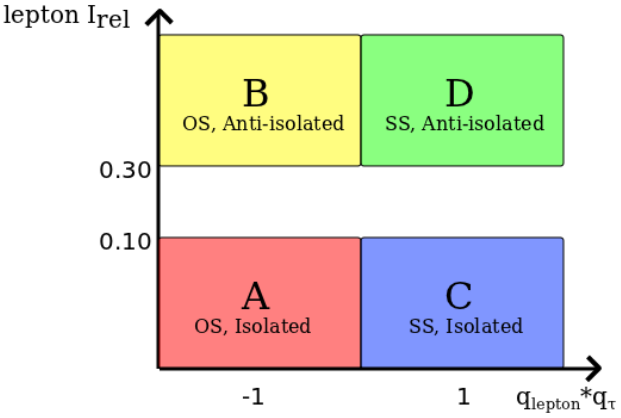


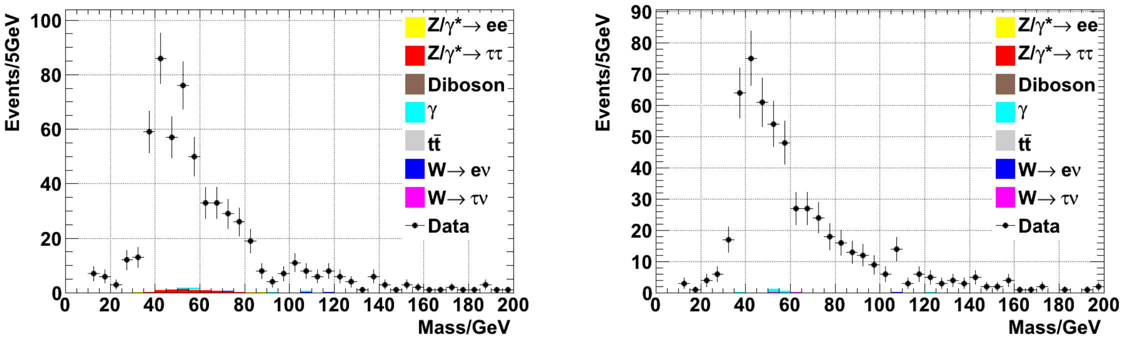
Figure 6.15: Diagram illustrating the control regions used for the ABCD method.

regions are shown in Figure 6.14. All of these contributions can be estimated, however to infer the number of QCD events in the signal region we also require the ratio between opposite-charge and same-charge QCD events, which we measure by inverting the lepton isolation cut to create high-purity opposite-sign and same-sign regions.

To calculate the QCD contribution to the signal region, we use an "ABCD" method, illustrated in Figure 6.15. The number of events in each of the regions is counted, and then providing the variables (charge, lepton isolation) are not correlated[25] the QCD contribution to the signal region can be calculated as

$$N_{QCD}^{OS,iso} = \frac{N_{QCD}^{SS,iso} \cdot N_{QCD}^{OS,anti-iso}}{N_{QCD}^{SS,anti-iso}}$$

[25] Correlation $r_{QCD}^{q,iso} = 0.006 \pm 0.010$ in MC.



(a) OS

(b) SS

Figure 6.16: Distributions for the anti-isolated QCD control regions for the $e\tau_{jet}$ channel. QCD MC is not shown, since it includes generator-level isolation and is not expected to be consistent. The estimated contribution from non-QCD sources are 9.7 events (1.6%) and 2.5 events (0.5%) respectively. The data contribution is only from triggers without an isolation requirement, with $\int L = 5\ \mathrm{pb}^{-1}$.

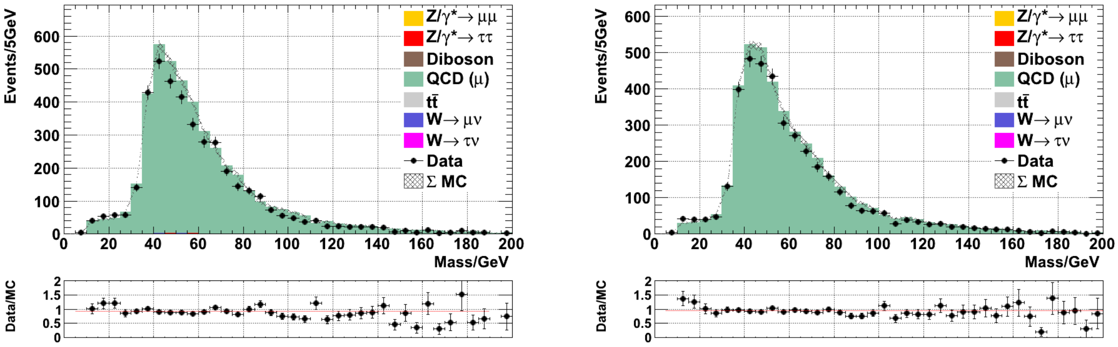
This method is relatively straightforward for the $\mu\tau_{jet}$ channel, but the $e\tau_{jet}$ channel is complicated by available MC and triggers. The EM-enriched MC QCD sample includes a generator-level isolation requirement[26], resulting in a significant discrepancy between MC and anti-isolated data. Compounding this, for the majority of Run2010B the leading electron trigger[27] included an isolation requirement, resulting in an useable isolation-neutral sample of only 5 pb^{-1}. The control regions are selected by using the full selection except for the lepton isolation and pair charge requirements used to construct the ABCD regions.

Directly inverting the isolation cut ($I_{rel} < 0.1$, Section 6.2.1) catches a tail of signal and other electroweak events in the anti-isolated region, so the cut for the anti-isolated region is increased to $I_{rel} > 0.3$, which results in anti-isolated, opposite-charge regions expected to be at least 98.5% and 99.6% pure for the $e\tau_{jet}$ and $\mu\tau_{jet}$ channels respectively, shown in Figure 6.16 and 6.17.

The isolated-lepton, same-charge control region (Figure 6.14) includes contamination from $Z \to ll\ jet \to \tau_{fake}$ (Section 6.3.1), $W \to l\nu_l$ and $W \to \tau\nu_\tau$ (Section 6.3.2) and (in the $e\tau_{jet}$ case) $\gamma + jet$ (Section 6.3.3). The event yields from each of these, calculated as per the previous sections, are subtracted, and finally the signal QCD contribution

[26] As a matter of practicality; the selection efficiency for such events means the $\approx 75 Mevent$ sample reduces to $O(100)$ events, and removing the isolation requirement would result in approximately five times as many events required.

[27] eg, `HLT_Ele12_SW_TightEleIdIsol_L1R`



(a) OS

(b) SS

Figure 6.17: Distributions for the anti-isolated QCD control regions for the $\mu\tau_{jet}$ channel. The estimated contribution from non-QCD sources are 19.3 events (0.4%) and 4.9 events (0.1%) respectively.

calculated using the charge ratio extracted from anti-isolated events. The results are shown in Table 6.10. The QCD contribution is given by

$$N^{OS}_{QCD} = \frac{N^{OS,anti-iso}_{QCD}}{N^{SS,anti-iso}_{QCD}} \times (N^{control}_{SS} - N_{Zllj} - N^{SS}_{Wl\nu}[-N^{SS}_{\gamma}])$$

Although the MC statistics available in each case are limited, there appears to be a significant data excess in the $\mu\tau_{jet}$ channel, with approximately a 50% excess observed in data compared to Monte Carlo after background subtraction[28]. The QCD sample does not appear to be seriously flawed given the good match observed for the overall opposite-sign distribution (Figure 6.6) and the anti-isolated control regions, so the origin of this effect is not clear. A possible explanation is the bottom cut-off of the muon and jet p_T in the QCD sample being equal to the lower bounds used in the analysis, and hence leaving a void in some cases of badly reconstructed taus or leptons.

6.3.5 $t\bar{t}$ Background

The $t\bar{t}$ background to the lepton-tau channels is very small (≈ 2 events/channel) and is estimated using MC. However, for the $e\mu$ channel, this is the leading background and it is possible to extract using a similar sideband to Section 6.3.2. We select the region where both $M_T(\mu, E_T^{miss}) > 60$ GeV and $M_T(e, E_T^{miss}) > 60$ GeV, which in the $e\mu$ final state

[28] A 2.9σ excess.

		$e\tau_{jet}$	$\mu\tau_{jet}$
Events in OS, anti-isolated region	$N_{OS,anti-iso}$	603	4119
Events in SS, anti-isolated region	$N_{SS,anti-iso}$	546	3931
Ratio	$R_{OS,SS}$	1.099 ± 0.064	1.049 ± 0.023
Events in SS, isolated region	$N_{SS}^{control}$	180	82
Z contribution	N_{Zllj}	9.6 ± 0.6	5.3 ± 0.4
W contribution	$N_{Wl\nu}^{SS}$	15.5 ± 1.4	16.8 ± 1.1
γ contribution	N_{γ}^{SS}	60.8 ± 7.3	–
QCD events in SS, isolated region	N_{QCD}^{SS}	94.1 ± 7.5	59.9 ± 1.2
QCD events in signal region	N_{QCD}^{OS}	103.4 ± 10.2	62.8 ± 1.9

Table 6.10: Observed values for the QCD control regions.

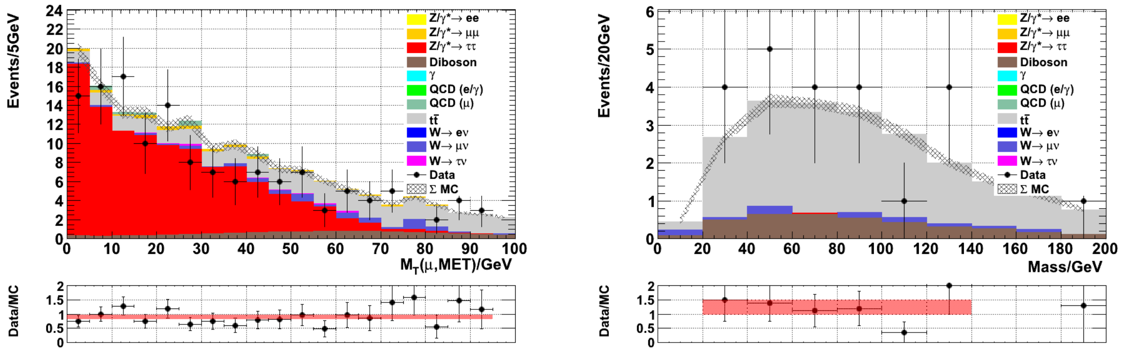
consists of 79% $t\bar{t}$, with the remainder predominantly from di-boson processes (which are subtracted from the control region using MC). The control region is shown in Figure 6.18 and results in Table 6.11.

As with the $Z \rightarrow ll + jet$ control region, the number of events in this control region is limited and hence a relatively large statistical error is introduced where the apparent MC error (from MC statistics and associated systematic errors) was much smaller. However, it would seem preferable to at least select some background for the $e\mu$ channel in a data-driven way rather than being wholly dependent on MC. The number of events in the signal region is given by

$$N_{t\bar{t}}^{OS} = (N_{t\bar{t}}^{control} - N_{VV}^{control}(MC)) \times \epsilon_{t\bar{t}}$$

		$e\mu$
Events in control region	$N_{t\bar{t}}^{control}$	23
MC di-boson events	$N_{VV}^{control}$	3.8
Efficiency	$\epsilon_{t\bar{t}}$	0.335 ± 0.012
Events in signal region	$N_{t\bar{t}}^{signal}$	6.4 ± 0.2

Table 6.11: Observed values for the $t\bar{t}$ control region.



(a) $M_T(\mu, E_T^{miss})$ distribution

(b) $t\bar{t}$ control region

Figure 6.18: Distribution of $M_T(\mu, E_T^{miss})$ for the $e\mu$ channel, and the control region based on this and the similar $M_T(e, E_T^{miss})$ distribution.

6.3.6 Signal

To obtain the number of $Z \to \tau\tau$ events, we subtract all of the extracted backgrounds described in the previous sections. For the $e\tau_{jet}$ and $\mu\tau_{jet}$ channel the $t\bar{t}$ and di-boson contributions are taken directly from MC, and for $e\mu$ all backgrounds except for $t\bar{t}$. The resulting event yields are shown in Table 6.12. Systematic errors are not included in the stated errors at this point, but are included in the fitting procedure. The number of signal events for each channel is given by

$$N_{Z\to\tau\tau}^{e\tau_{jet}} = N_{OS} - N_{Ze\tau} - N_{Zeej} - N_{We\nu}^{OS} - N_{\gamma}^{OS} - N_{QCD}^{OS} - N_{t\bar{t}}(MC) - N_{VV}(MC)$$

$$N_{Z\to\tau\tau}^{\mu\tau_{jet}} = N_{OS} - N_{Z\mu\tau} - N_{Z\mu\mu j} - N_{W\mu\nu}^{OS} - N_{QCD}^{OS} - N_{t\bar{t}}(MC) - N_{VV}(MC)$$

$$N_{Z\to\tau\tau}^{e\mu} = N_{OS} - N_{Zll}(MC) - N_{Wl\nu}^{OS}(MC) - N_{QCD}^{OS}(MC) - N_{t\bar{t}} - N_{VV}(MC)$$

In general, good agreement is shown between both the overall expected yield, and the extracted backgrounds compared to MC composition.

The majority of the differences can be explained by the poisson error on the number of events observed, but we see notable discrepancies in the number of QCD events predicted for the $\mu\tau_{jet}$ channel (arising from the observed difference in the same-charge control

		$e\tau_{jet}$	$\mu\tau_{jet}$	$e\mu$
Events in signal region	N_{OS}	374	352	91
Z contribution	$N_{Zl\tau}$	31.2 ± 1.1	10.4 ± 0.6	2.6 ± 0.3
	N_{Zllj}	9.6 ± 0.6 (42.6)	5.3 ± 0.4 (15.1)	–
W contribution	$N^{OS}_{Wl\nu}$	30.0 ± 1.6 (36.8)	53.0 ± 1.8 (57.6)	2.4 ± 0.4
γ contribution	N^{OS}_{γ}	31.4 ± 4.2 (34.0)	–	–
QCD contribution	N^{OS}_{QCD}	103.4 ± 10.2 (100.2)	62.8 ± 1.9 (40.2)	1.8 ± 0.7
$t\bar{t}$ contribution	$N_{t\bar{t}}$	1.8 ± 0.1	2.6 ± 0.1	6.4 ± 0.2 (5.9)
Di-boson contribution	N_{VV}	0.7 ± 0.0	1.0 ± 0.0	2.3 ± 0.0
$Z \to \tau\tau$ yield	$N_{Z\to\tau\tau}$	165.9 ± 11.2 (146.5)	216.9 ± 2.7 (232.5)	75.5 ± 0.9 (89.7)

Table 6.12: Observed values for the signal regions. Only statistical errors arising from MC efficiencies are shown. MC values are given in parathenses.

region). There are also fewer than expected $e\mu$ signal events[29], although this is less significant given the limited number of events for this channel.

6.4 Cross Section Extraction

To convert the event yields described in Section 6.3.6 into a cross section, we calculate

$$\sigma(pp \to Z \to \tau\tau) = \frac{N}{A \cdot \epsilon \cdot Br \cdot \ell (= 36.0 \text{ pb}^{-1})}$$

Where N is the number of signal events extracted, A is the acceptance of decay products within the selection η and p_T cuts, ϵ is the efficiency for events in the acceptance, Br is the branching ratio $\tau\tau \to (e\tau_{jet}, \mu\tau_{jet}, e\mu)$ and ℓ is the integrated luminosity of the analysed data.

6.4.1 Acceptance and Efficiency

The cross section of $Z \to \tau\tau$ events is usually given as the cross section of $Z/\gamma* \to \tau\tau$ events for which $60 < M_{Z/\gamma*} < 120$ GeV, whereas the signal sample up until now, has

[29] 1.6σ shortfall.

(dictated by necessity), been a general Drell-Yan sample with $M_{Z/\gamma*} > 20$ GeV. The selections described in Section 6.2.1 in practice select almost entirely events within the former mass range ($> 98\%$ for all channels), but we cannot use this sample to determine the acceptance of the desired events.

To obtain the acceptance, 10^6 $Z \to \tau\tau$ events were generated with $60 < M_{Z/\gamma*} < 120$ GeV using PYTHIA6 (Tune Z2) and otherwise standard CMS configuration, and the acceptance of each of the three channels determined. To obtain the efficiency, the main MC sample was used, with corrections applied for the fraction of events outside the $[60, 120]$ window after the acceptance and in the final selected events. The acceptance and efficiency values are shown in Table 6.13.

		$e\tau_{jet}$	$\mu\tau_{jet}$	$e\mu$
Branching ratio	Br	0.230	0.224	0.062
Acceptance	A	0.111	0.119	0.111
Efficiency	ϵ	0.163	0.248	0.372
Product	$BrA\epsilon$	4.16×10^{-3}	6.61×10^{-3}	2.56×10^{-3}

Table 6.13: Acceptance and efficiency values from MC.

6.4.2 Systematic Errors

The following major sources of systematic error contribute:

- Trigger efficiency, particularly for the frequently-changing electron triggers.
- Lepton identification efficiency. These are measured in both cases with tag-and-probe methods in the $Z \to ee$ and $Z \to \mu\mu$ channels[64].
- Hadronic tau identification efficiency, which is the largest source of error for the lepton-tau channels. This was measured in [58] using $Z \to \tau\tau \to \mu\tau_{jet}$ events, by selecting (without tau identification) a tau-enriched sample, and then finding the efficiency of tau identification.
- Lepton energy scale uncertainty, affecting the efficiency of acceptance cuts. Based on the momentum resolution of the muon system and energy resolution of the ECAL, this is expected to be approximately 1% in each case. To obtain the effect this has

on the acceptance, the analysis was run on events with shifted particle energies and the relative efficiencies measured.

- Hadronic tau energy scale uncertainty. The magnitude was measured in [58] using the observed mass distribution of the $Z \to \tau\tau \to \mu\tau_{jet}$ system (taking advantage of better constrained muon energy scale) and found to be 3%. The effect on the efficiency was measured, as with the lepton energy scale, by mutating the particle energies and re-running the analysis.
- E_T^{miss} uncertainty, affecting the efficiency of the $M_T(l, E_T^{miss})$ cuts. This was measured in [70] using a tau-embedding technique, where high-purity $Z \to \mu\mu$ data was modified at the particle flow level by subtracting the muons, simulating tau decays with the same kinematics and injecting them back into the events, then recalculating the E_T^{miss}.
- Luminosity uncertainty, affecting overall normalisation. This was measured using Van der Meer scans during several dedicated beam fills, further discussed in [71].
- PDF uncertainty, affecting both the acceptance/efficiency values and the correct scaling of MC samples, where directly subtracted from data. The uncertainty has been estimated elsewhere[72] by comparing the values of different PDF sets, as 2%.

The values of the systematic errors used are given in Table 6.14. The largest sources of uncertainty are tau identification and luminosity normalisation.

6.4.3 Fitting

The cross-section fitting was performed with 10^5 Monte-Carlo toy experiments. This "brute-force" approach was chosen because of the large number of variables involved in the three-channel fit, with 15 measurements and 28 nuisance parameters required, and the complexity of modelling the internal correlation analytically. For each iteration, a single vector of systematic error values and event yields was generated and the signal extraction calculation (as described in Section 6.3.6) performed.

- Systematic errors are modelled as gaussian distributions with $\mu = 1$ and σ as per Table 6.14.

	$e\tau_{jet}$	$\mu\tau_{jet}$	$e\mu$
Trigger Efficiency	1.0%	0.2%	0.2%
Electron ID	1.3%	–	1.3%
Muon ID	–	0.9%	0.9%
Tau ID	23%	23%	–
Electron Energy Scale	1.1%	–	1.1%
Muon Energy Scale	–	1.1%	1.1%
Tau Energy Scale	3.2%	3.2%	–
M_T Scale		2%	
Luminosity		4%	
PDF		2%	

Table 6.14: Values of systematic errors, by channel.

- Event counts from data are modelled as a poisson distribution with $\lambda = N$. There are 7 counts for $e\tau_{jet}$, 6 for $\mu\tau_{jet}$ and 2 for $e\mu$, and each is then scaled according to the relevant systematics.
- Background estimation efficiencies are varied as gaussian distributions with σ given by the statistical errors of the MC they were calculated from.
- MC values used for subtraction are varied as gaussian distributions with σ given by their statistical error and PDF uncertainity.
- The acceptance was scaled according to the PDF uncertainity.

For each channel, the value is extracted as the maximum of the calculated cross section distribution, and the one-sigma errors on this value are extracted by integration of the distribution. The distribution of cross sections obtained from toy experiments is shown in Figure 6.19, and with repeated experiments the separate errors due to statistical, systematic, luminosity or tau ID sources are separately calculated (by allowing the appropriate error to vary and fixing all others). The extracted errors are listed in Table 6.15 and results and comparison plotted in Figure 6.20.

The extracted cross sections are $(1100 \pm 180_{stat} \pm 110_{syst} \pm 45_{lumi} \pm 170_{\tau ID})$pb for $e\tau_{jet}$, $(880 \pm 95_{stat} \pm 40_{syst} \pm 40_{lumi} \pm 180_{\tau ID})$pb for $\mu\tau_{jet}$ and $(895 \pm 110_{stat} \pm 40_{syst} \pm 35_{lumi})$pb for $e\mu$.

	$e\tau_{jet}$	$\mu\tau_{jet}$	$e\mu$
$\sigma(pp \to Z \to \tau\tau)$	1100	880	895
Statistical	180	95	110
Systematic	110	40	40
Luminosity	45	40	35
Tau ID	170	180	–
Overall uncertainity	265	250	130

Table 6.15: Extracted cross section for each channel, with a breakdown of error sources. Values in pb.

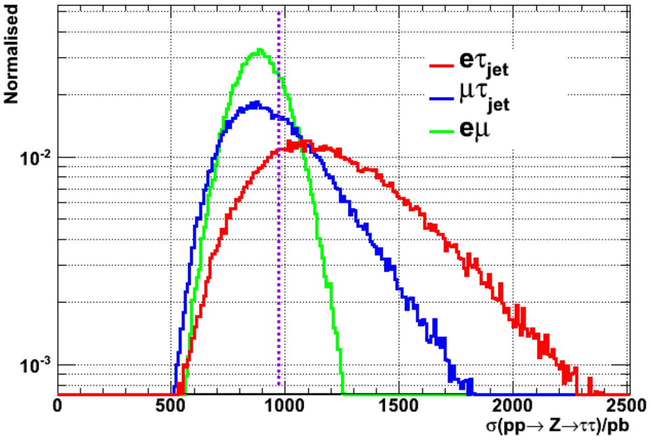


Figure 6.19: Distribution of $Z \to \tau\tau$ cross section, calculated by MC toy experiments, with all sources of uncertainty included. The purple line shows the NNLO value of 972 pb.

The results are mutually compatible, and compatible with both the NNLO prediction[73], (972 ± 40) pb, and measurements of the $Z \to ee$ and $Z \to \mu\mu$ cross sections made by CMS[64], $(992 \pm 11_{stat} \pm 18_{syst} \pm 40_{lumi})$pb and $(968 \pm 8_{stat} \pm 7_{syst} \pm 39_{lumi})$pb respectively.

The $e\mu$ channel is already statistically limited, and the $e\tau_{jet}$ and $\mu\tau_{jet}$ would also be but for a tighter constraint on the tau identification efficiency.

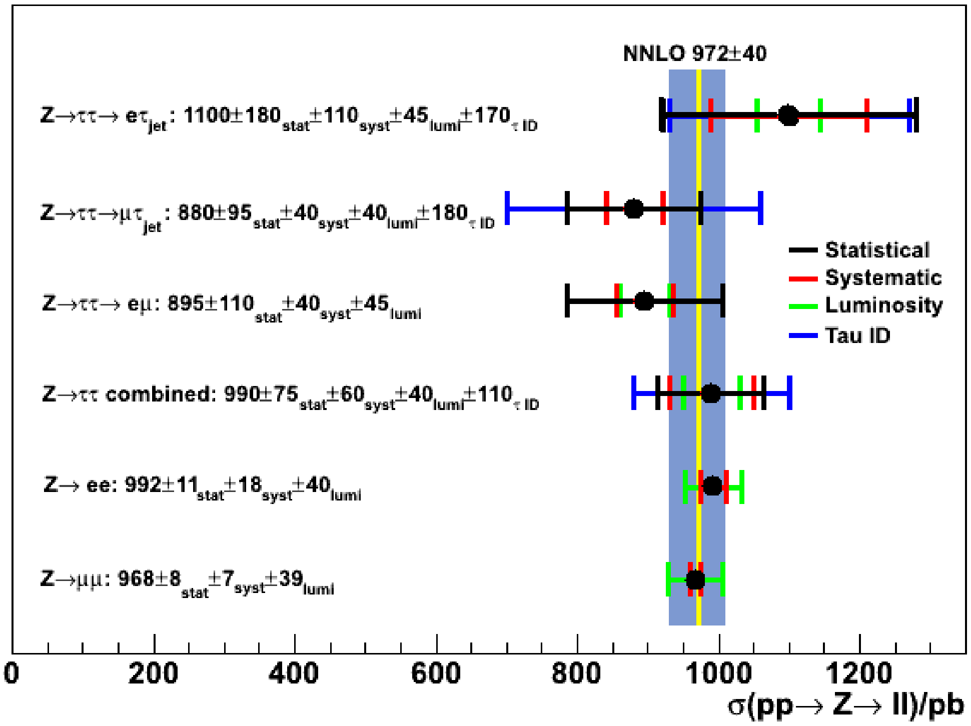


Figure 6.20: Extracted cross sections and their errors, compared with the NNLO prediction and values measured in the $Z \to ee$ and $Z \to \mu\mu$ channels[64].

Chapter 7

Limits for MSSM $\Phi \to \tau\tau$

7.1 Introduction

The lower mass limit of the light, neutral, minimally-supersymmetric Higgs boson ($\Phi = h, H, A$) decaying to tau pairs is extremely similar in signature to the Z boson considered in Chapter 6, and the same selections and event samples can be used to find these events. The tau channel is the second highest by branching ratio[1] for the MSSM Higgs, after $b\bar{b}$ [2].

The scope of this analysis is limited by the available data, with very few events expected[3], except in the very low mass and high $\tan\beta$ scenario. The focus is therefore on setting limits on the possible cross section (and hence the m_A, $\tan\beta$ plane), rather than discovery.

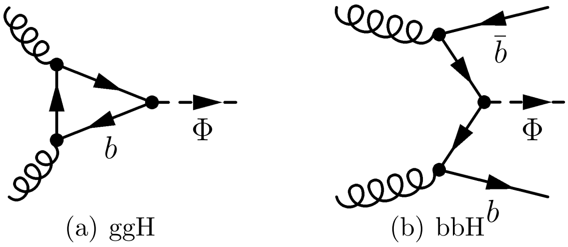
Two production modes are considered; gluon fusion via a bottom quark loop, and $b\bar{b}$ annihiliation (denoted as ggH and bbH). The former has a very similar signature to $Z \to \tau\tau$ events, the latter contains two additional heavy flavour jets from the remaining two bottom quarks. These processes are shown in Figure 7.1.

Ten values of m_A between 90 and 300 GeV are considered for this analysis, although both the production cross section and branching ratio to taus falls off at high masses ($m_A > 200$ GeV). The mass points and associated cross sections (for $\tan\beta = 30$) are listed in Table 7.1.

[1] $Br(\Phi \to \tau\tau) \in [8, 16]\%$ for the m_A, $\tan\beta$ range considered.

[2] For $m_A < 2m_W$.

[3] In the scenario $m_A = 120$ GeV and $\tan\beta = 30$, using the acceptance and efficiency values calculated for $Z \to \tau\tau$ (Section 6.4.1), the expected yield between the three tau decay channels would be ≈ 20 events.



(a) ggH (b) bbH

Figure 7.1: Leading-order diagrams for the ggH and bbH production modes.

The MSSM Higgs masses and branching ratios were calculated by the LHC Higgs Working Group[74] using *FeynHiggs* 2.7.4[75] and the production cross sections calculated with *bbh@nnlo*[76] for the quark-quark process and *ggh@nnlo*[77] and *HIGLU*[78] for the gluon-gluon process. The m_H^{max} scenario[79] is used, with the following parameters:

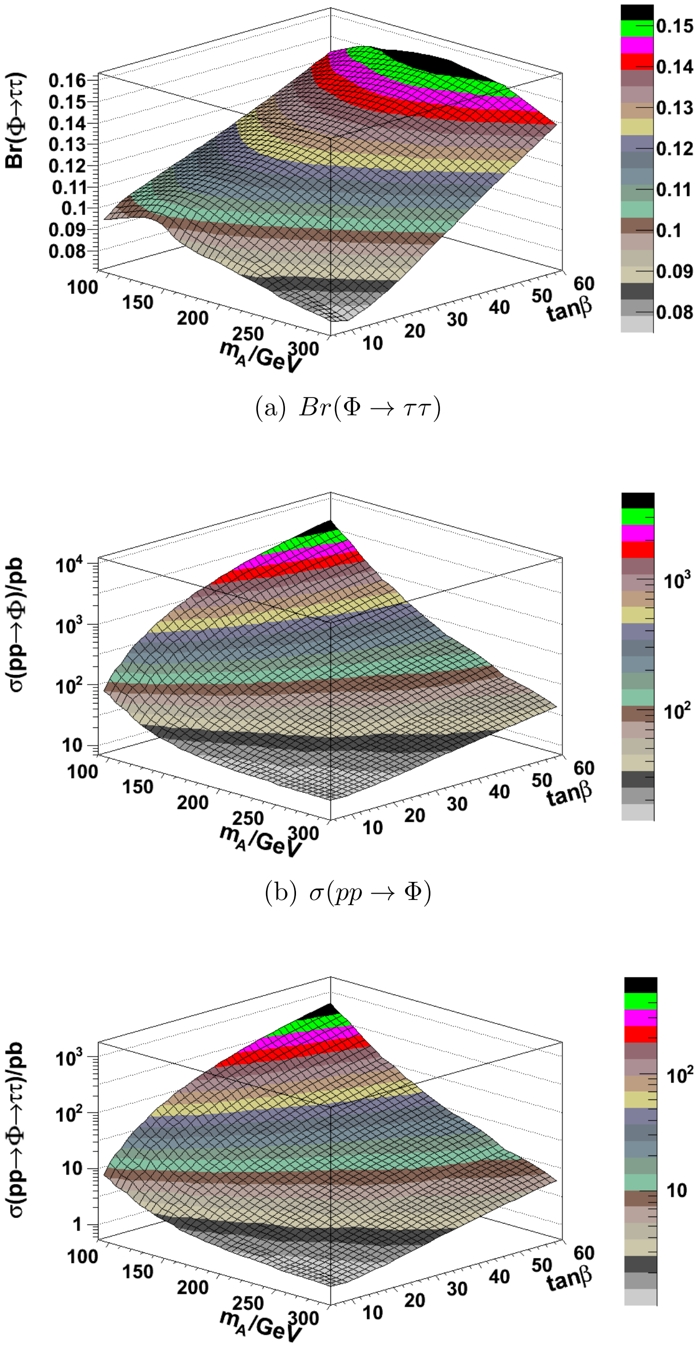
- Top quark mass $M_{top} = 172.5$ GeV
- Soft SUSY breaking squark mass $M_{SUSY} = 1$ TeV
- Higgsino mass parameter $\mu = 200$ GeV
- Gluino mass $M_{\tilde{g}} = 800$ GeV
- Gaugino mass parameter $M_2 = 200$ GeV

The actual fitting is performed against the mass spectra for $\tan\beta = 30$, and changes in $\tan\beta$ modelled as a scaling of these shapes. Figure 7.2 shows the tau branching ratios and combined production cross sections for the MSSM Higgs, as a function of m_A and $\tan\beta$.

7.2 Template Extraction

To perform the fitting and limit extraction, we require a visible mass template for each of the major backgrounds in each channel, along with the variations of each caused by statistical and systematic errors, and the expected number of events for each background. As with the event selection, the template generation is handled very similarly for the $e\tau_{jet}$ and $\mu\tau_{jet}$ channels, and somewhat differently for the $e\mu$ channel.

The control regions and background extraction methods from Chapter 6 are used to find the expected number of events for the non-$Z \rightarrow \tau\tau$ backgrounds. The $Z \rightarrow \tau\tau$



(a) $Br(\Phi \to \tau\tau)$

(b) $\sigma(pp \to \Phi)$

(c) $\sigma(pp \to \Phi \to \tau\tau)$

Figure 7.2: Tau branching ratio, Higgs production cross section and combined $\Phi \to \tau\tau$ cross section, as a function of m_A and $\tan\beta$.

Mass/GeV	$\sigma(gg \to \Phi \to \tau\tau)$/pb	$\sigma(bb \to \Phi \to \tau\tau)$/pb
90	91.8	87.8
100	56.3	64.2
120	24.2	36.5
130	16.5	27.8
140	11.6	21.2
160	6.38	13.2
180	3.93	8.55
200	2.66	5.71
250	1.42	2.32
300	1.05	1.04

Table 7.1: SUSY Higgs mass points used in the analysis. Cross sections correspond to $tan\beta = 30$, and are calculated to NNLO. The sample integrated luminosities all exceed 1 fb^{-1}, and all contain $1.0 \times 10^5 - 1.1 \times 10^5$ events).

background however is obtained using Monte-Carlo[4], since it was defined previously as the number of "left-over" events, which in this case would guarantee zero Higgs signal. The cross section for Z production (in $Z \to ee$ and $Z \to \mu\mu$ events) has been found to be compatible with the NNLO prediction used[64].

For the lepton-tau channels, we define three templates:

- $Z/\gamma* \to \tau\tau$
- Electroweak
- QCD (including $\gamma + jet$ in the $e\tau_{jet}$ case).

For the $e\mu$ channel, the $t\bar{t}$ background is included as an extra template (being the largest background after $Z \to \tau\tau$ rather than a minor background in the other channels). Similar $Z \to \tau\tau$, electroweak and QCD templates are used.

All the templates use uniform 10 GeV binning, which balances sufficient shape resolution in the main peak ($\approx$ 40 GeV FWHM[5]) with sufficient events per bin for fitting.

[4] Unlike the previous chapter, no attempt is made to exclude $Z/\gamma*$ events with small invariant mass, although in practice this makes very little difference.

[5] **F**ull **W**idth **H**alf **M**aximum.

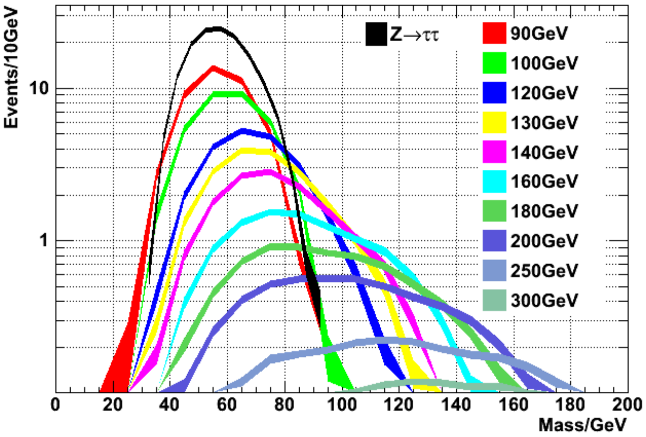


Figure 7.3: Visible mass shapes ($e\tau_{jet}$ channel) of MSSM $\Phi \to \tau\tau$ decays at the indicated m_A values. The $Z \to \tau\tau$ mass spectrum is shown for comparison. The line thickness shows the statistical error. Templates are normalised for $\ell = 36.0$ pb^{-1}.

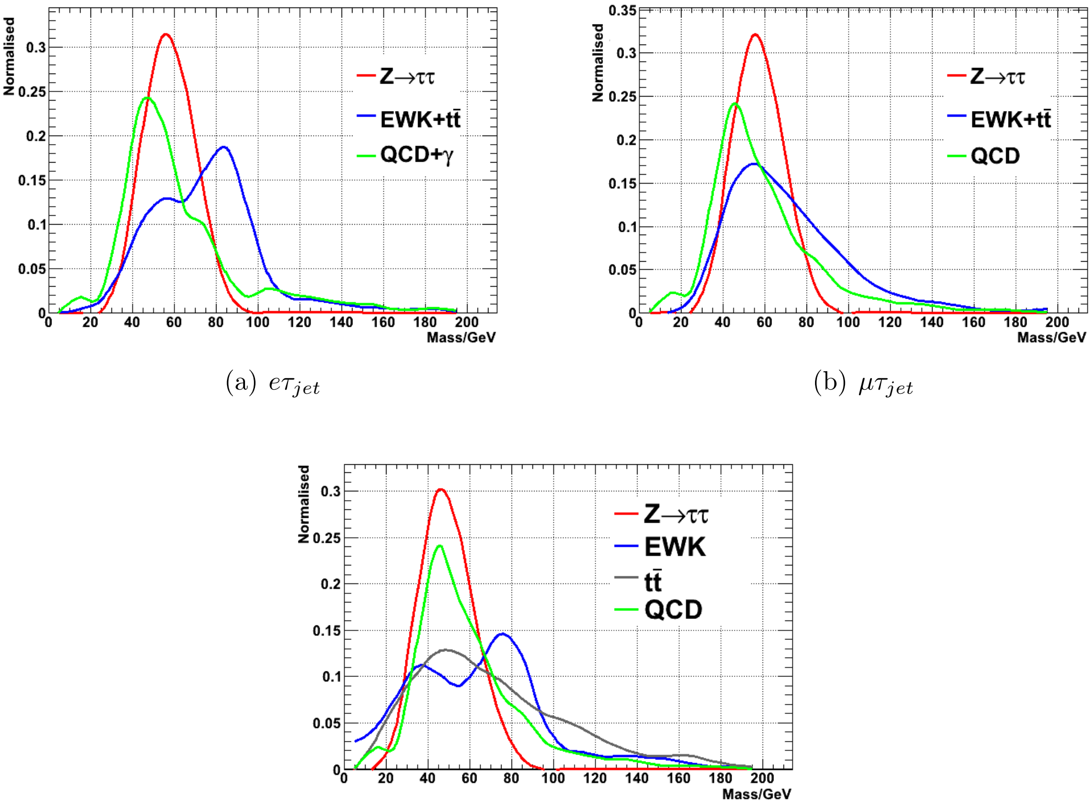
The templates for the MSSM Higgs signal are shown in Figure 7.3 (for the $e\tau_{jet}$ channel), and the normalised final templates for each channel in Figure 7.4.

Here we have assumed that Monte-Carlo models for the electroweak processes are relatively trustworthy; these are well-studied problems for which a good theoretical description exists, it has been shown elsewhere[64] that templates from these samples provide an excellent model for the data, and the available Monte-Carlo statistics are adequate.

7.2.1 Electroweak Template

The electroweak template includes $Z \to ll$, $W \to l\nu_l$ and $W \to \tau\nu_\tau$ processes. These processes have relatively small statistical and systematic errors, and the ratio between W and Z processes has a small enough error to model them as one template. The $t\bar{t}$ (except in the $e\mu$ channel) and di-boson backgrounds are included in this template since they are both very small[6] and have small errors, hence by subsuming them into the electroweak template the additional parameter space of another template is avoided.

[6] $\approx$ 1% of total background, $\approx$ 2.5% of the electroweak background for $e\tau_{jet}$, $\mu\tau_{jet}$.



(a) $e\tau_{jet}$

(b) $\mu\tau_{jet}$

(c) $e\mu$

Figure 7.4: Normalised visible mass background templates for each of the channels, shown fitted with smooth curves for illustration.

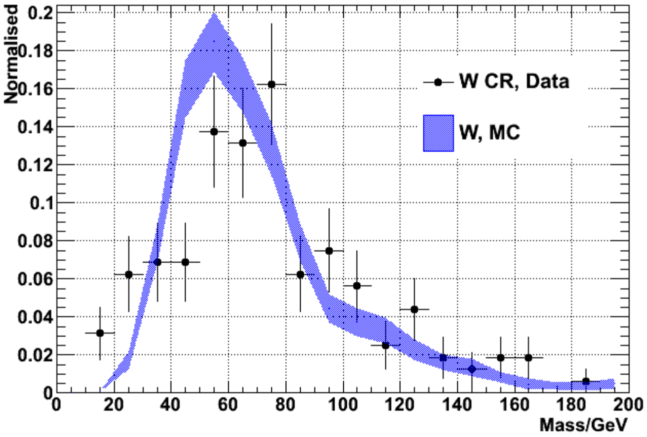


Figure 7.5: Normalised mass shapes ($\mu\tau_{jet}$ channel) for the $W \to \mu\nu_\mu$ opposite-charge control regions from data, and the mass shapes (from MC) of the $W \to \mu\nu_\mu$ contribution to the signal regions. The distribution includes subtraction of the $t\bar{t}$ contribution (which is in any case small and approximately flat).

The number of events in this region is calculated using the background extraction scheme described in Section 6.1.4,

$$N_{EWK} = \epsilon_{Zl\tau}N^{control}_{Zl\tau} + \epsilon_{Zllj}N^{control}_{Zllj} + \epsilon_{OS,Wl\nu}(N^{control}_{OS,Wl\nu} - N^{control}_{t\bar{t}}(MC))[+N_{t\bar{t}}(MC)] + N_{VV}(MC)$$

(using the same notation). None of the shapes obtainable from data in the three control regions show sufficient similarity to the Monte-Carlo final state shapes that they can be used directly.

Figure 7.5 compares the mass shape of the W (opposite-charge) control region and the W contribution to the final state. The statistics of the control region are limited, but the shape would appear to have a tighter peak. This can be partly explained by the control region containing almost no $W \to \tau\nu_\tau$ contribution while the final state does, with the second neutrino in the $W \to \tau\nu_\tau$ process resulting in greater smearing of the visible mass and hence a wider peak.

Figure 7.6 compares the mass shape of the $Z \to ll$ in the final state with the shapes from the $Z_{l\tau}$ (τ_{fake} from lepton) and Z_{llj} (τ_{fake} from recoil jet) control regions. For the shape of the Z_{llj} control region, MC must be used regardless since the region contains insufficient events to extract a meaningful shape (the $Z_{l\tau}$ region contains plenty). The admixture of the two clearly does not match the Monte-Carlo final state shape. The $Z_{l\tau}$ contribution in particular appears significantly wider in the final state than control region;

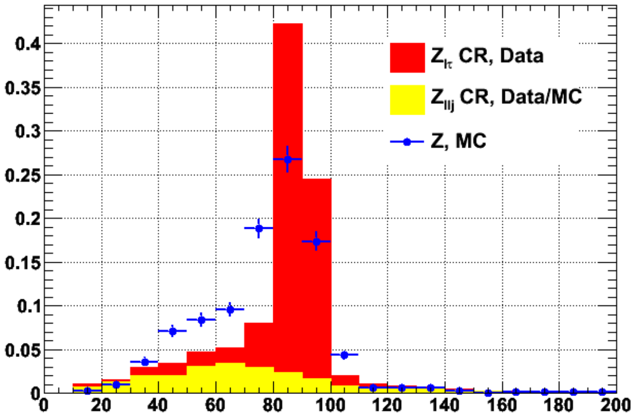


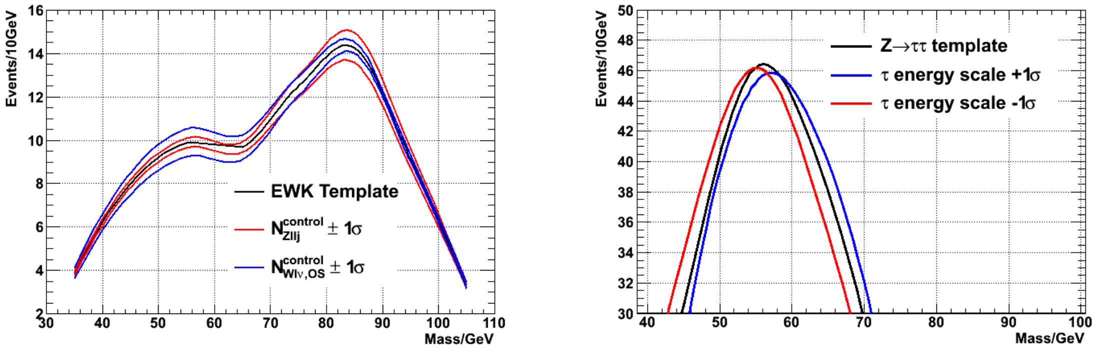
Figure 7.6: Normalised mass shapes ($e\tau_{jet}$ channel) for admixture of the $Z_{l\tau}$ and Z_{llj} control regions (with the former wholly from data and the latter using scaling from data and shape from MC), compared to the expected distribution in the signal region, showing poor agreement between the CR shapes and signal shape.

this is expected since the control region inverts electron rejection to select good electrons, whereas the final state contains electrons that have been reconstructed sufficiently badly to pass tau electron rejection, and hence are likely to have worse energy and position resolution and result in a wider peak.

Consequently, the template is built from the MC final-state shapes of the W and Z contributions. The effect of fluctuations in the number of observed events in the control regions, and the error on the control region efficiencies are modelled as shape fluctuations corresponding to $\pm 1\sigma$. By way of example, the effects of varying the number of events in the W and Z_{llj} control regions on the electroweak template and of varying the tau energy scale on the $Z \to \tau\tau$ template are shown in Figure 7.7.

7.2.2 QCD Template

The QCD template is inevitably the largest source of uncertainity (due to the large errors on estimating the QCD contribution, the theoretical uncertainities on modelling the hadronisation process and the limited lepton-enriched QCD and $\gamma + jet$ statistics available) in the fitting for the lepton-tau channels. For the $e\tau_{jet}$ channel, we include the $\gamma + jet$ contribution along with QCD in this template. The number of events in each



(a) Electroweak

(b) $Z \to \tau\tau$

Figure 7.7: Section of templates for the $e\tau_{jet}$ channel (fitted with smoothed curves for illustration), showing (a) the variation of the electroweak template resulting from $\pm 1\sigma$ variations of the number of events in the $Z \to ll + jet$ and $W \to l\nu_l$ OS control regions, and (b) the variation of the $Z \to \tau\tau$ template resulting from $\pm 1\sigma$ variation of the tau energy scale.

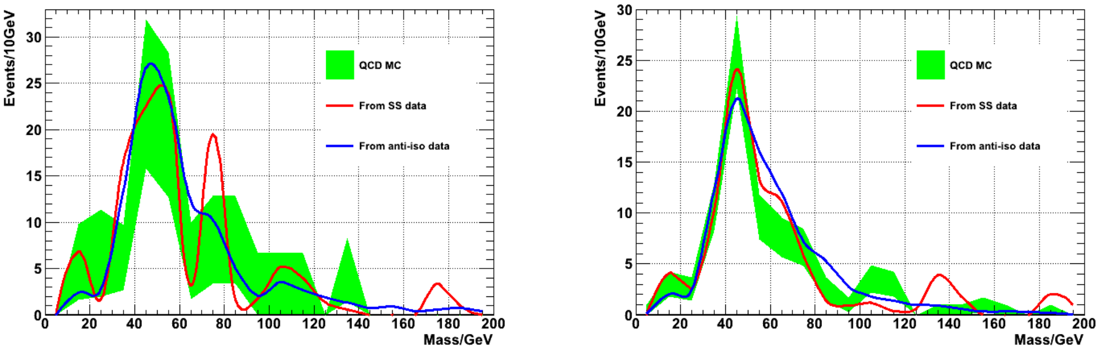
case is obtained as per Section 6.1.4,

$$N_{QCD} = \frac{N_{QCD}^{OS,anti-iso}}{N_{QCD}^{SS,anti-iso}}(N_{SS}^{control} - \epsilon_{Zllj}N_{Zllj}^{control} - \epsilon_{SS,Wl\nu}N_{SS,Wl\nu}^{control}[-\epsilon_{SS,\gamma}N_{\gamma}^{control}])[+\epsilon_{OS,\gamma}N_{\gamma}^{control}]$$

Given the poor statistics for both QCD and $\gamma + jet$ MC, the shapes for both must be extracted from data.

The QCD shape can be obtained in the same way as the estimated number of QCD events, by subtracting the shapes of the other backgrounds ($W \to l\nu_l$, $Z \to llj$ and also $\gamma + jet$ for $e\tau_{jet}$) from the same-charge control region then scaling the resulting shape according to the charge ratio of anti-isolated events. The shapes for the other backgrounds are derived from MC, since the number of collected events is small and the MC statistics good. However, the relatively small number of events in the same-charge control region contrive to make this a very uncertain shape.

In practice, the shape of the opposite-charge, anti-isolated region (used to estimate the QCD charge ratio) appears to be compatible with the isolated mass shape in MC (such as statistics allow), while producing a much smoother result than using the shape obtained from the same-charge region. Consequently, the anti-isolated shape, scaled using the number of events from the same-charge region is used. The comparison between MC, the subtracted SS shape and the anti-isolated shape is shown in Figure 7.8.



(a) $e\tau_{jet}$ (b) $\mu\tau_{jet}$

Figure 7.8: Comparison between the MC QCD shapes (with the proviso of the large statistical errors indicated) in the final state and the shape obtained either from the same-charge control region (after subtraction of contamination), or from the opposite-charge, anti-isolated control region (scaled by the number of events in the same-charge region). For the $\mu\tau_{jet}$ channel the QCD has been rescaled to match the predicted number of events; as noted in Chapter 6 there is otherwise a significant discrepancy.

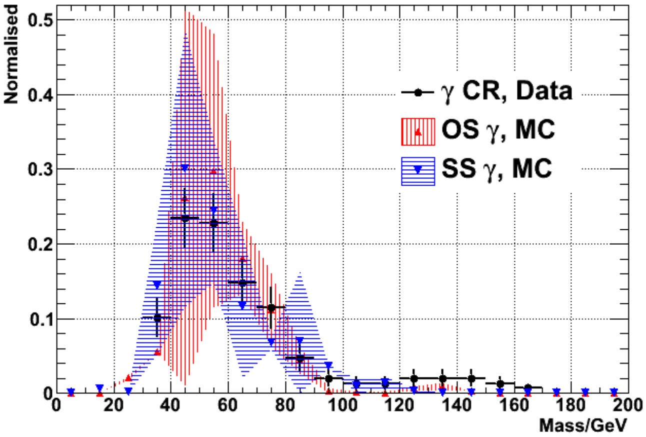


Figure 7.9: Normalised mass shapes ($e\tau_{jet}$ channel) for the $\gamma + jet$ control region (from data) compared to the expected distributions in the OS and SS signal regions, showing agreement (within the very large statistical errors).

The $\gamma + jet$ background is the most statistically limited, in both MC and data. The shape from data of the control region is used (scaled appropriately), which appears to be compatible with the MC shape in both the opposite-charge and same-charge regions. The comparison between the data and MC shapes is shown in Figure 7.9.

The overall QCD template (for the $e\tau_{jet}$ channel) is an admixture of anti-isolated QCD shape and the $\gamma + jet$ control region shape. As with the electroweak template, relative fluctuations between the two contributions are handled as shape systematics.

For the $e\mu$ channel the $\mu\tau_{jet}$ QCD shape is used. Only a handful of MC QCD events are selected for the $e\mu$ channel, corresponding to approximately one expected event, from which no distribution can be obtained. However, these events generally contain a real muon and a fake electron arising from a charged hadron (passing the looser electron identification used by this channel), so it is not unreasonable to suggest that the overall mass shape would be similar to $\mu\tau_{jet}$ QCD shape. In any case, the expected contribution is very small so the inaccuracy in this choice of mass shape is unlikely to make any difference.

7.2.3 $t\bar{t}$ Template

For the $e\mu$ channel, the $t\bar{t}$ contribution is modelled as a separate template. Given the small number of events (≈ 20) in the $t\bar{t}$ control region, the MC shape is used, scaled according to the calculated $t\bar{t}$ contribution. While the complete simulation of $t\bar{t}$ suffers from similar jet-related uncertainities to QCD, it is only a significant contribution in the $e\mu$ channel, in which we are principally only interested in the well-modelled lepton part of the decay.

7.3 Fitting

The fitting model is constructed from the background templates (Figure 7.4) and a Higgs production template (Figure 7.3) for the appropriate m_A value. The Higgs templates are normalised such that the template scaling factor, R_H is equal to the combined cross section $\sigma(pp \rightarrow \Phi \rightarrow \tau\tau)$. The fitting procedure is ultimately to extract a 95% confidence upper bound on this parameter, from which the $\tan\beta$ limit can be calculated.

Systematic errors that affect the mass shapes, such as particle energy scales, and the relative contribution to composite templates of different control regions are modelled using precalculated histograms corresponding to the central value and $\pm 1\sigma$. For parameter values within this range, the shape is interpolated between the centre and $\pm 1\sigma$ histogram by linear interpolation of the cumulative distribution functions[80][7] (for values outside this range, the appropriate 1σ template is used).

Fitting is performed using a profile likelihood method (as implemented by RooStats[81]), with the likelihood function given as a product of a poisson distribution for each bin (across all channels or a single channel), and a gaussian distribution for nuisance parameters (the systematic errors and uncertainities on control region efficiencies). The likelihood is given by

$$\ell = \prod_{i \in channels} \prod_{j \in bins} \frac{(R_H s_{ij} + b_{ij})^{d_{ij}} e^{-(R_H s_{ij} + b_{ij})}}{d_{ij}!} \prod_{k \in systs} \frac{e^{-\frac{(x_k - \mu_k)^2}{2\sigma_k^2}}}{\sqrt{2\pi\sigma_k^2}}$$

A maximum likelihood fit is performed using MINUIT[82], and the best fitted set of nuisance parameters (θ) obtained. The fitting is performed again for a set of fixed values of R_H with all other values floating. The likelihood ratio is then

$$\lambda(R_H) = \frac{\ell(R_H, \hat{\hat{\theta}})}{\ell(\hat{R_H}, \hat{\theta})}$$

By Wilks' theorem, $-2\log\lambda(R_H)$ tends to a single degree of freedom χ^2 distribution, and thus the 95% confidence limit can be obtained by finding the value of R_H at which the profile likelihood corresponds to a one-sided, 95% χ^2 value.

To accommodate the possibility of there being multiple minima in the maximum-likelihood function, the fitting procedure is run repeatedly with initial values of floating parameters generated representatively. The whole fitting is performed using both the actual dataset ("observed") and the background-only distributions from Monte-Carlo ("expected"), to find the limit the method can extract in a guaranteed no-signal environment.

The fitting consistently finds the maximum likelihood value of R_H is zero. The expected and observed values of the 95% confidence upper limit on $\sigma(pp \to \Phi \to \tau\tau)$ are given in Table 7.2 and shown in Figure 7.10. The expected and observed values are compatible within ±1σ across the entire mass range.

[7] Performing these interpolations requires > 75% of the CPU time for the overall fitting, since a large number of probabilities need to be sampled in CDF (Cumulative Distribution Function) space to interpolate even a simple PDF.

Figure 7.11 shows the breakdown of cross section limits per channel, showing that the limits are primarily derived from the $e\mu$ channel at low mass and the $\mu\tau_{jet}$ channel at high mass, with the $e\tau_{jet}$ channel not significantly contributing to the limit at any point. The slight kink in the combined limit around $m_A = 200$ GeV almost entirely arises from the $\mu\tau_{jet}$ fit; it is not obvious what feature causes this (but the limit remains within the 1σ expected band). The high mass fits are sensitive to the small number of data events and the relatively poorly-defined EWK and QCD templates outside of the $Z \to \tau\tau$ peak.

m_A/GeV	Expected	Observed	$\tan\beta$
90	131.0	145.6	27.0
100	107.2	105.8	28.1
120	73.1	55.1	28.6
130	58.6	46.7	30.8
140	42.5	35.0	31.0
160	28.1	26.9	35.4
180	19.3	19.8	38.4
200	12.6	9.0	31.3
250	5.4	8.1	47.5
300	4.3	6.9	> 60

Table 7.2: Observed and expected 95% CL limits on $\sigma(pp \to \Phi \to \tau\tau)$, and corresponding values of $\tan\beta$ for the observed limits. The $\tan\beta$ value for $m_A = 300$ GeV falls outside the range of the grids provided by [74], and has in any case significant uncertainity.

Comparing these results to the published MSSM limits from LEP[83] and Tevatron[84] (with 2.2 fb^{-1}) shows that a small improvement in exclusion compared to other experiments is achieved throughout the studied mass range. The exclusion limits are shown in Figure 7.12.

7.4 B-tagging

The $b\bar{b}$ Higgs production mode includes two heavy flavour jets in addition to the ($e\tau_{jet}$, $\mu\tau_{jet}$, $e\mu$) signature, and we can exploit this signature to increase the purity of selecting this production mode (although this will exclude the gluon production mode).

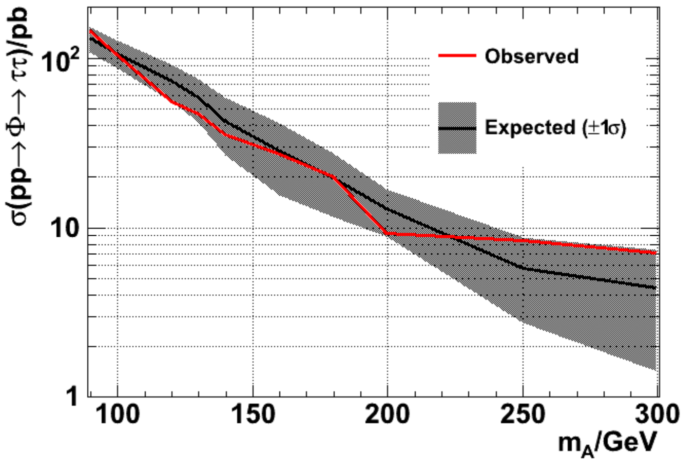


Figure 7.10: Observed and expected 95% CL limits on $\sigma(pp \to \Phi \to \tau\tau)$, and the 1σ error on the latter.

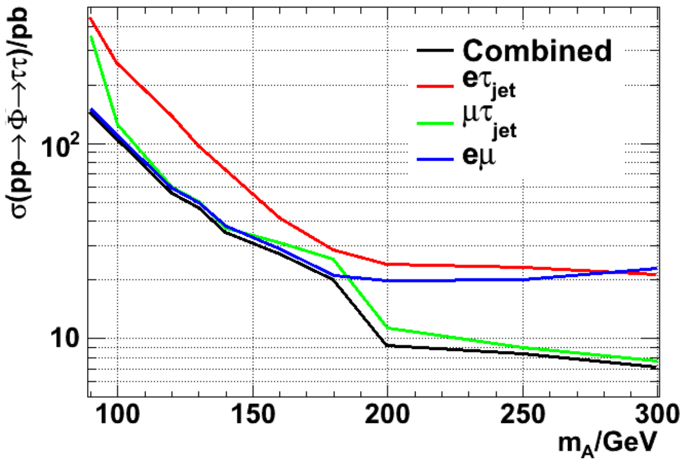


Figure 7.11: Observed 95% CL limits on $\sigma(pp \to \Phi \to \tau\tau)$, shown for each of the channels individually and the combination of the three channels.

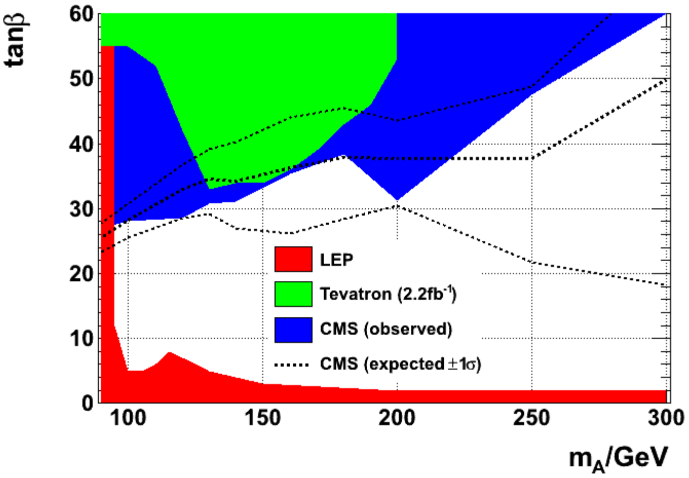


Figure 7.12: Observed 95% exclusion in the m_A, $\tan\beta$ plane, from this analysis, Tevatron[84] and LEP[83].

In addition to the basic selections described in Section 6.2.1, we require:

- At least one PF jet with $p_T > 15$ GeV and $|\eta| < 2.4$.
- Jet non-collinear with the selected $(e\tau_{jet}, \mu\tau_{jet}, e\mu)$.
- Jet passes b-tagging discriminant.

The b-tagging algorithm used is Track Counting, High Efficiency (TCHE)[85]. The tracks in the jet are sorted by the impact parameter significance, $\frac{IP}{\sigma_{IP}}$ (where σ_{IP} is the error on the impact parameter), and then cut based on the impact parameter significance of the second track (in decreasing order). We use the "loose" working point for this discriminator, corresponding to a cut of $\frac{IP}{\sigma_{IP}} > 1.7$ on the second track. This is expected to have an efficiency of 56% for b-jets, and a mistag rate of 2% for jet $p_T < 30$ GeV. The distribution of this discriminant for signal and selected backgrounds is shown in Figure 7.13.

As the figure shows, while the b-tagging allows us to select $b\bar{b}$ produced Higgs events, $t\bar{t}$ events also have a high efficiency to pass this cut. It is possible to further reduce the $t\bar{t}$ background by cutting on events in which there are more than two jets with $p_T > 15$ GeV and $|\eta| < 2.4$[8], and requiring exactly one b-tagged jet[9].

[8] $\approx$ 95% selected events with > 2 jets non-colinear to the lepton(s) and taus were $t\bar{t}$ in MC.

[9] Only around 10% of $b\bar{b}$ Higgs events yielded two b-tagged jets, whereas around half of $t\bar{t}$ events did.

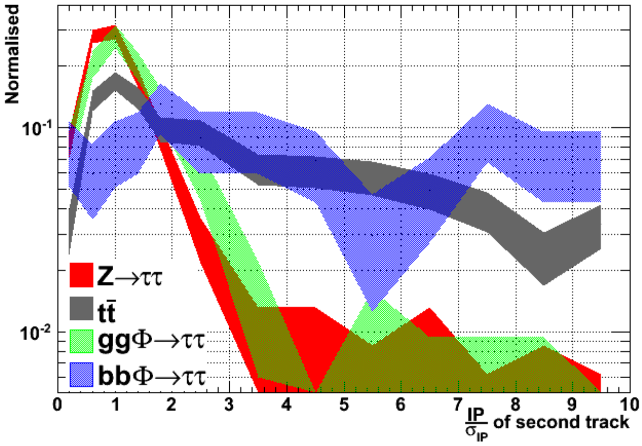


Figure 7.13: Normalised distributions of the TCHE discriminant (impact parameter significance of the second track), for the $e\tau_{jet}$ channel. The Higgs samples shown have $m_A = 100$ GeV. The "loose" cut is $\frac{IP}{\sigma_{IP}} > 1.7$.

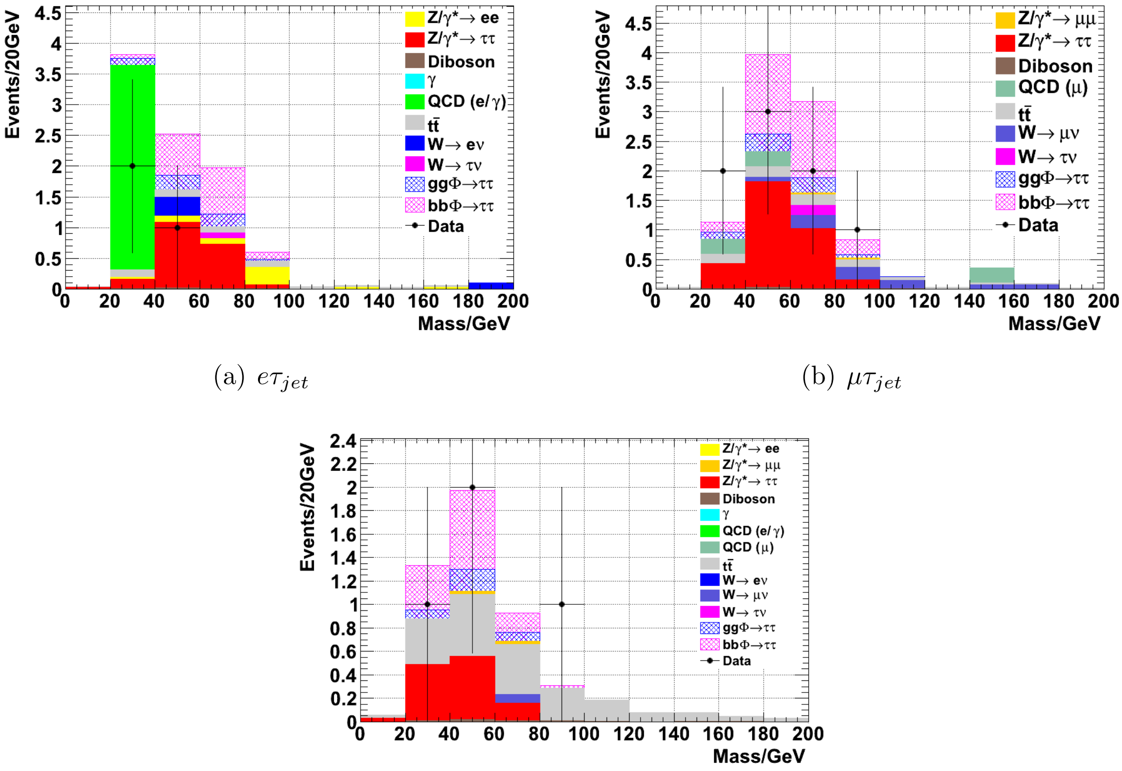
With these requirements, we are able to select a much purer ($\approx 50\%$ Higgs, for $m_A = 100$ GeV[10]) final state than without b-tagging, at the cost of extremely low efficiency[11]. The visible mass distributions are shown in Figure 7.14, with the $e\tau_{jet}$, $\mu\tau_{jet}$ and $e\mu$ channels selecting 3, 8 and 4 events respectively.

Clearly the number of events selected by this approach is too small for meaningful interpretation with the available data, but given the high purity compared to the non-tagged selection this approach should provide a useful extra channel given more data[12].

[10] The $e\tau_{jet}$ channel has a lower purity of 31%, but one third of the MC is a single highly-weighted QCD event.

[11] Approximately an order of magnitude less than the baseline $\tau\tau$ selections including jet kinematics, b-tagging and $t\bar{t}$ rejection.

[12] Here we have used b-tagging as a strict subset of the $e\tau_{jet}$, $\mu\tau_{jet}$ and $e\mu$ selections, but it is likely that the addition of b-tagging would allow other parts of the selection to be loosened to increase the event yield. This is still unlikely to make these channels practical given the 2010 dataset, however.



(a) $e\tau_{jet}$

(b) $\mu\tau_{jet}$

(c) $e\mu$

Figure 7.14: Visible mass distributions for each channel after b-tagging. Plotted Higgs signal is for $m_A = 100$ GeV and $\tan\beta = 30$.

Chapter 8

Computing Monitoring Pages

8.1 CMS Computing

The CMS experiment has a large number of computer-based systems required for the day-to-day running of the experiment, with varying degrees of criticality. The health and status of these systems need to be monitorable in a way that does not require developer-level knowledge of the underlying systems or even, ideally, specialised client software.

This allows a small number of people on shift ("shifters") to monitor a large number of systems from remote operation centres or the main CMS control room, and (somewhat) frees the original developers from the workload of running a production system.

The ideal technical solution to these requirements is a web-based[86] monitoring component integrated into these systems, which is easily portable and allows rich, interactive monitoring systems to be provided.

Since there is also inevitably a large amount of shared functionality between different monitoring systems (such as authentication of approved individuals, caching generated data, validation of user inputs, etc), it is advantageous to provide as much common functionality as reasonably possible in a shared framework, which reduces the effort to monitor a new service and ensures that security-critical code does not exist as a large number of poorly-written implementations.

8.2 The DQMGUI Framework

The DQM[1]GUI[2] was not originally developed[3] as a framework for computing monitoring, but rather to address a separate requirement for DQM shifters to quickly sift through a large number of plots generated from each run, checking that distributions appear roughly as they are expected.

It consists of a C++/ROOT library which handles the extraction of data and plotting of a large number of histograms from express-reconstructed data, and a python web framework side based on the *CherryPy*[87] server, interacting with a javascript[88] web application used by the DQM shifter to access the plots. This is very similar to what is required for a computing monitoring framework, and the non-DQM-specific parts were adapted into a computing monitoring system dubbed "Overview".

The system is designed to be highly modular, so individual monitoring tools can be added to the system without the author requiring a deep knowledge of the framework internals. Separate monitoring tools are referred to as *workspaces*, generally one per underlying system, each of which contains one or more *views* representing different aspects of the monitored system and/or different ways of looking at the data.

Interaction between the server and the user's web browser uses the REST[4] model. For each user, the server maintains a *state*, containing information such as their current *view*, which plots they have open, and authentication information to prevent their session being hijacked. Each action the user performs triggers a request to the server to update their *state*. The request is checked for validity, the user's *state* updated and finally the server responds to the user with their complete, updated *state* as JSON[89][5]. This ensures:

- The complete *state* is only maintained at one point, to avoid the need for two copies to be kept in synchronisation.
- The server has guarantees about the validity and internal consistency of the *state*, since it cannot rely on the client to perform these checks[6].

[1] **D**ata **Q**uality **M**onitoring

[2] **G**raphical **U**ser **I**nterface

[3] Predominantly by Lassi Tuura.

[4] **Re**presentational **S**tate **T**ransfer

[5] **J**ava**S**cript **O**bject **N**otation, a lightweight and (somewhat) human readable data serialisation format, based on javascript syntax.

[6] Since the client is free to modify the javascript code running locally.

- The server can sanitise all input before it is returned to the client[7].
- The user's session can be saved between uses, which is not yet reliably possible on the client side[8], providing that they return to the same URL (which includes their session ID). This is the normal behaviour of most modern browsers (remembering open tabs on exit, etc).

This approach does require a large number of requests to the server, but the requests involved are small (typically $O(100)$ bytes) and the latency for state updates is very low (typically < 30 ms, even outside CERN), and a single server can handle the request volume from $O(1000)$ users without congestion.

On the client side, the application is implemented as a single page loaded by the user, associated with a new, empty, session *state* when they first connect to *Overview*. The page contains the *Overview* header and a blank canvas for *workspaces* to draw on. The page makes requests back to the server for a list of available workspaces and populates the *Overview* menu appropriately. When a user requests a particular workspace, the application in their browser requests the (compressed[90]) javascript for that workspace. The javascript workspace[9] then populates the canvas[10] DOM[11] tree with the appropriate HTML[12] elements.

On the server side, a *workspace* consists of one or more python classes. The *workspace* class handles events such as setting up a default *state* when first accessed by a user. Actions performed by the user (such as changing the parameters of a plot) are handled by *view* class, or by the *workspace* class if the functionality of the *views* are either sufficiently trivial or sufficiently similar for them not to need their own class.

The *workspace* and/or *view* classes do not generally handle the actual data the *workspace* presents themselves, but the details of how the user is viewing it. The data

[7] It is necessary to sanitise any text being directly returned to the user to avoid the Cross-Site Scripting class of attacks, where javascript from an untrusted source is echoed back to the user and executed in the context of their session cookies and circumvents the same-origin policy.

[8] Through cookies, HTML5 local storage, etc.

[9] Consisting of a single nested function with methods to handle construction, teardown, user actions and responses from the server. Javascript is a prototype-based language with no concept of a class, but a mixture of javascript objects and functions can be used to approximately the same effect.

[10] This model breaks the web principle of progressive enhancement; that is, that the most important content should arrive and be displayed first and the structure and decoration be successively built around it (and a semi-loaded page will still be somewhat useable), whereas Overview will show all of a page or none of it. Progressive enhancement is largely unsupportable where the page is an application rather than a document.

[11] **D**ocument **O**bject **M**odel

[12] **H**yper**T**ext **M**arkup **L**anguage

are handled by a *source* class, which does not distinguish between individual users but rather serves plots or data entirely specified by the URL[13]. The client-side *workspace* generates the appropriate URLs for the necessary data after each state change. Since data may take some time to generate and load, depending on the requested complexity, the separation between changing the *state* and requesting the new data ensures that the client *workspace* remains responsive while waiting for the data.

Figure 8.1 illustrates how the model works in the context of the *Prodmon* application (Section 8.3).

The *Overview* framework provides core functions such as authenticating users, storing their states between sessions[14], caching (where appropriate) and directing requests from users to the appropriate *workspace* or *source* classes.

Overview was first written largely as a technology demonstrator. Initially, the only interactive *workspace* was a plotter for the PhEDEx[91, 92][15] data transfer system. Compared to their existing legacy website which served static pages, the *Overview workspace* was significantly more responsive when displaying transfer plots (although it was never intended to replicate most of the features of the existing website). A *workspace* was also provided for monitoring CPU and disk usage in the CAF[16], but this relied on periodically scraping images generated by other monitoring systems and did not provide any meaningful interactivity.

Section 8.3 and 8.4 cover the implementation of two new interactive *workspaces* for *Overview*.

8.3 Prodmon - Monte-Carlo Production Monitoring

In order to develop and validate analysis strategies for CMS, very large quantities of Monte-Carlo simulated data need to be generated, marshalled and made available for users. Recent production campaigns have generated as many as 1.2×10^9 events, requiring approximately one millenium of CPU time and generating around $3PiB$ of output.

[13] **U**niversal **R**esource **L**ocator

[14] Using Python's *pickle* object serialisation format.

[15] **Ph**ysics **E**xperiment **D**ata **Ex**port, although this may be a backronym.

[16] CERN **A**nalysis **F**acility, a large batch farm available for priority uses to a subset of LHC experimental users.

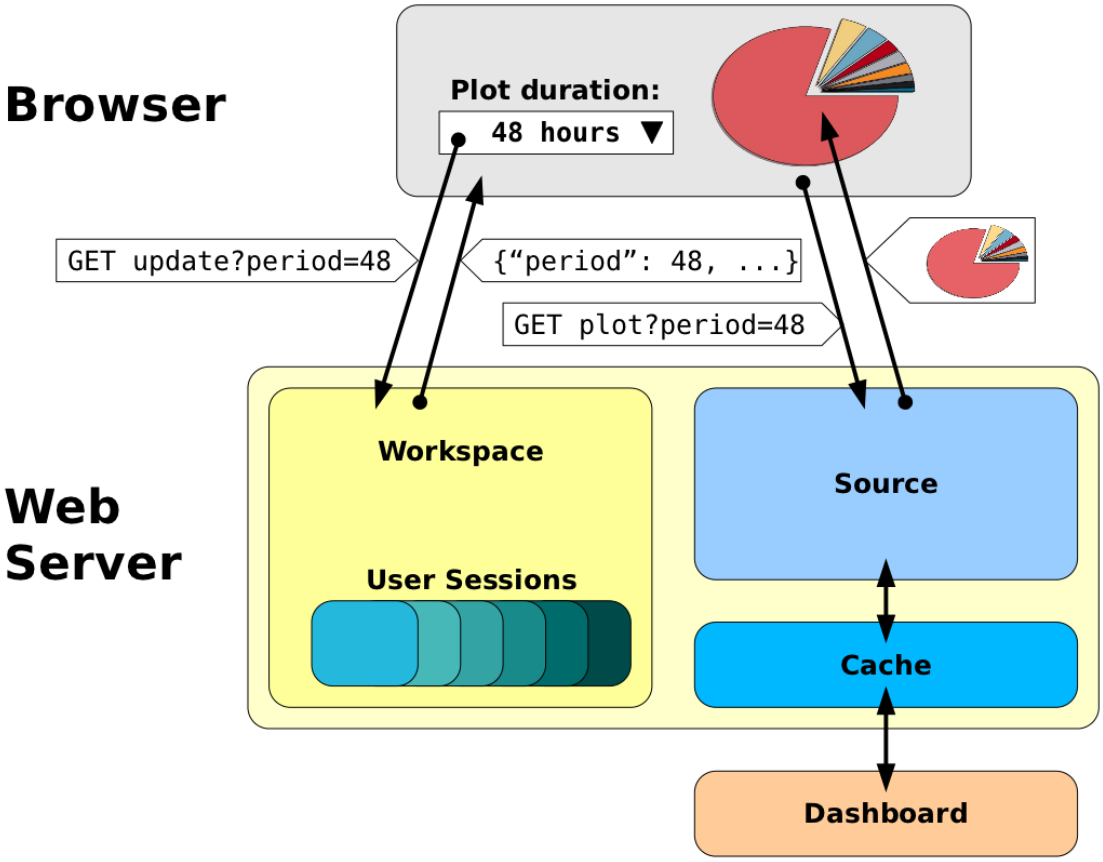


Figure 8.1: Diagram illustrating the flow resulting from the user changing a plot option in *Prodmon*. The user changes the period of time the plot represents, which triggers a request to the server. The *workspace* class receives this request, checks that it comes from a valid user and that the requested value is valid and consistent with the user's other options. The *workspace* then responds with the user's complete *state* (since the change might have caused other variables to be changed, or been rejected). The browser receives the updated state, and uses this to generate a request for a new plot. The *source* checks the plot parameters, and responds with an updated plot, if possible from an internal cache.

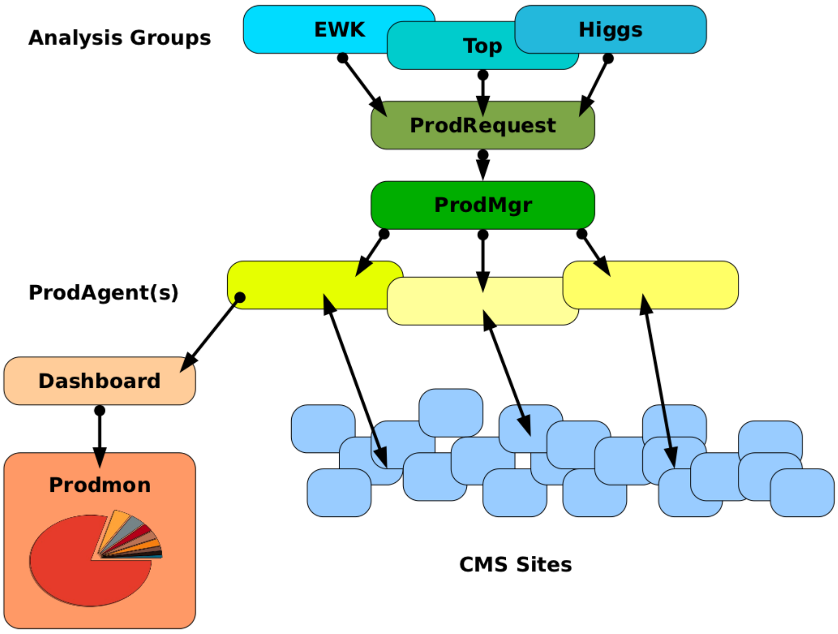


Figure 8.2: Diagram illustrating the flow of CMS Monte-Carlo production, from physics groups requesting datasets through *ProdAgent(s)* dispatching jobs to computing sites, and finally the summary being sent to the *Dashboard* and displayed by *Prodmon*.

Requests for production of Monte Carlo samples originate from the different Physics Analysis Groups (PAG) or Physics Object Groups (POG) and are collected by the *ProdRequest*[93] server. Production is predominantly done in major campaigns 2-3 times per year, but requests can be made at any time. Requests are approved to ensure that CPU and storage capacity is shared appropriately between groups. Approved workflows are added to the *ProdMgr* server[17]. The workflows are pulled from the *ProdMgr* by one or more *ProdAgents*, each of which manages MC production on an assigned group of CMS sites[18]. The *ProdAgent* dispatches MC production jobs as capacity becomes available at the client sites, and monitors those jobs as they run.

When production jobs dispatched by the *ProdAgent* finish, successfully or otherwise, an XML[19] report[20] is sent to a collector running on the *Dashboard*[94] job-monitoring

[17] It should be noted that that *ProdRequest* and *ProdMgr* are currently in variable states of availability, and a large number of workflows bypass these steps and are injected manually into the *ProdAgent*.

[18] Production takes place predominantly at Tier-2 sites, but it is planned to use spare Tier-1 capacity where available.

[19] e**X**tensible **M**arkup **L**anguage.

[20] Containing details such as number of events generated, CPU time consumed, and exit code of the generator.

system. *Dashboard* periodically summarises these incoming reports into daily, weekly and monthly summary tables in an Oracle database. The whole flow is illustrated in Figure 8.2.

Finally, Dashboard provides a basic XML data service (*PAquery*) to allow these tables to be queried. It is this data service which *Prodmon* is built upon.

The Prodmon *workspace* provides two *views*.

- *Summary*, which displays three fixed pie charts showing production broken down by country, site and dataset, and brief statistics.

- *Freeplot*, an interactive plotting environment allowing multiple concurrent plots to be specified.

The summary view in shown in Figure 8.3. This is the default landing page and was designed to show "at a glance" the current status of Monte-Carlo production, both in aggregate and broken down by the most productive sites and ProdAgents, and the most produced datasets. No user interaction is provided for beyond the fact that the plots are hyperlinked to a configurable version of themselves in the freeplot workspace.

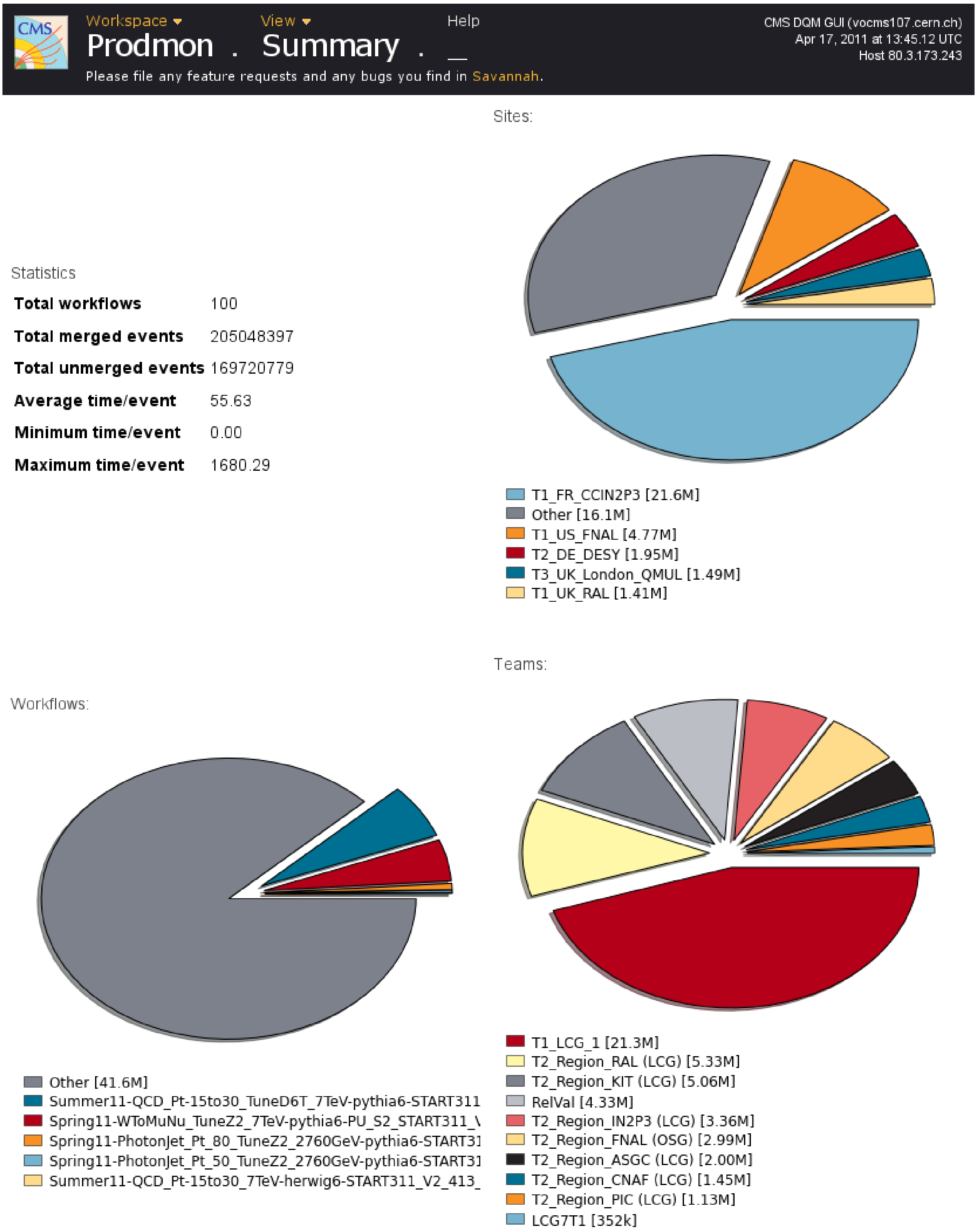
The Freeplot workspace handles all operations more complex than showing summaries. Figure 8.4 shows a screenshot of a possible session. The basic design is a multi-document interface, containing a maximised plot the user is currently viewing and zero or more minimised plots which the user wishes to be able to return to quickly without having to manually specify each time. Providing the user visits the same URL the next time they return (which would normally be the case with a modern browser that remembers tabs or windows when closed, as previously observed), the plot selection will have been saved and be available for subsequent use.

The top options bar shows the required options when selecting a plot, which are:

- The quantity to plot; eg, number of events generated, quantity of CPU time used, the number of concurrent jobs running. For quantities which can be divided into successful and unsuccessful jobs, these subcategories are available.

- The parameter by which to group the quantity; eg, computing site the job ran at, the dataset to which it belongs, or the originating prodagent.

- Whether to show the actual production jobs or the subsequent jobs merging the resulting output into consistent-sized files.

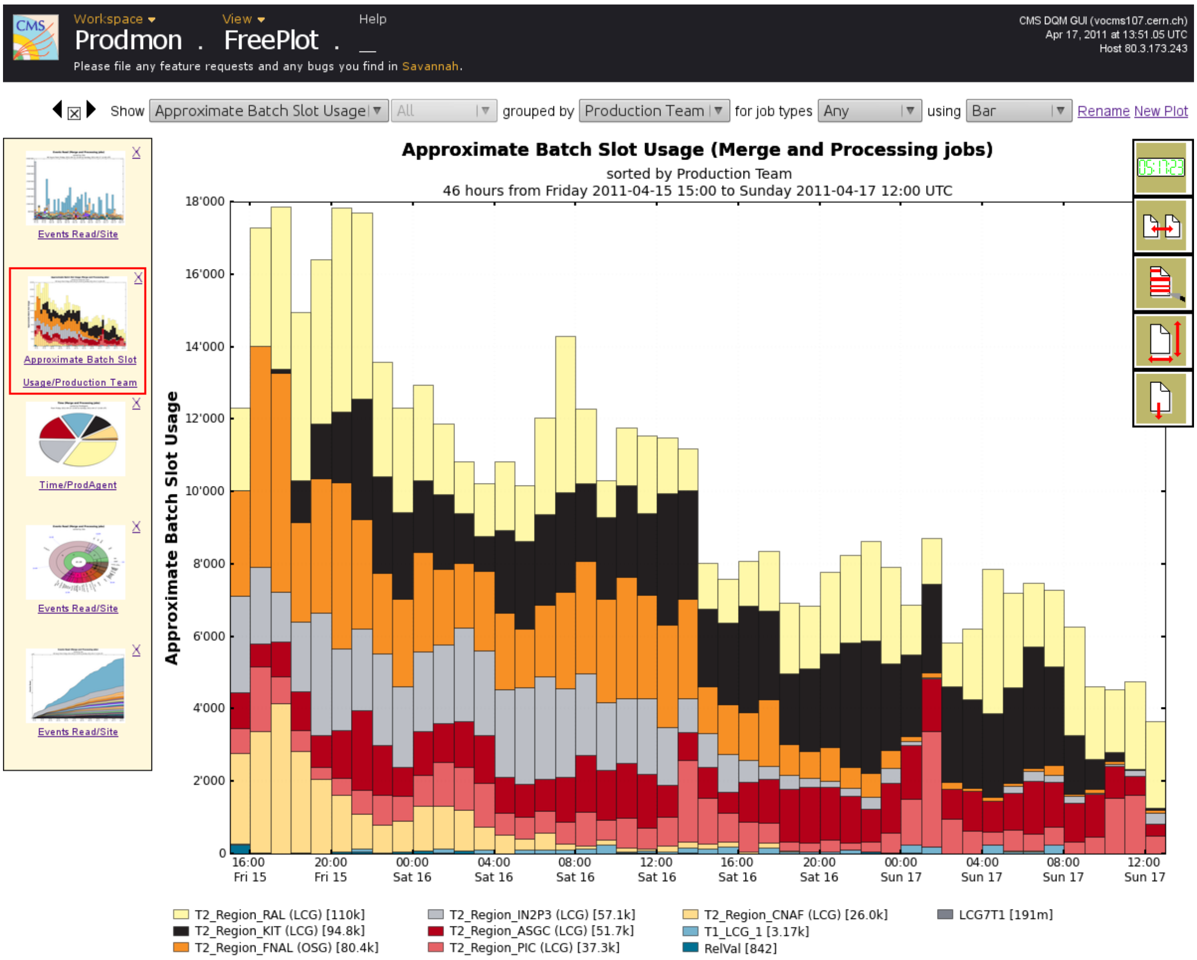

CMS
Workspace
View
Help
Prodmon . Summary . _
Please file any feature requests and any bugs you find in Savannah.
CMS DQM GUI (vocms107.cern.ch)
Apr 17, 2011 at 13:45.12 UTC
Host 80.3.173.243
Sites:
Teams:
Workflows:

Statistics

Total workflows	100
Total merged events	205048397
Total unmerged events	169720779
Average time/event	55.63
Minimum time/event	0.00
Maximum time/event	1680.29

T1_FR_CCIN2P3 [21.6M]
Other [16.1M]
T1_US_FNAL [4.77M]
T2_DE_DESY [1.95M]
T3_UK_London_QMUL [1.49M]
T1_UK_RAL [1.41M]
Other [41.6M]
Summer11-QCD_Pt-15to30_TuneD6T_7TeV-pythia6-START311
Spring11-WToMuNu_TuneZ2_7TeV-pythia6-PU_S2_START311_\
Spring11-PhotonJet_Pt_80_TuneZ2_2760GeV-pythia6-START31
Spring11-PhotonJet_Pt_50_TuneZ2_2760GeV-pythia6-START31
Summer11-QCD_Pt-15to30_7TeV-herwig6-START311_V2_413_
T1_LCG_1 [21.3M]
T2_Region_RAL (LCG) [5.33M]
T2_Region_KIT (LCG) [5.06M]
RelVal [4.33M]
T2_Region_IN2P3 (LCG) [3.36M]
T2_Region_FNAL (OSG) [2.99M]
T2_Region_ASGC (LCG) [2.00M]
T2_Region_CNAF (LCG) [1.45M]
T2_Region_PIC (LCG) [1.13M]
LCG7T1 [352k]

Figure 8.3: Prodmon Summary *view*, showing MC production in April 2011.



Workspace ▾ View ▾ Help

Prodmon . FreePlot . _

Please file any feature requests and any bugs you find in Savannah.

CMS DQM GUI (vocms107.cern.ch)
Apr 17, 2011 at 13:51.05 UTC
Host 80.3.173.243

Show Approximate Batch Slot Usage ▾ | All ▾ | grouped by Production Team ▾ | for job types Any ▾ | using Bar ▾ | Rename | New Plot

CMS
Events Read/Site
Approximate Batch Slot Usage/Production Team
Time/ProdAgent
Events Read/Site
Events Read/Site
Approximate Batch Slot Usage (Merge and Processing jobs)
sorted by Production Team
46 hours from Friday 2011-04-15 15:00 to Sunday 2011-04-17 12:00 UTC
Approximate Batch Slot Usage
18'000
16'000
14'000
12'000
10'000
8'000
6'000
4'000
2'000
0
16:00 Fri 15
20:00 Fri 15
00:00 Sat 16
04:00 Sat 16
08:00 Sat 16
12:00 Sat 16
16:00 Sat 16
20:00 Sat 16
00:00 Sun 17
04:00 Sun 17
08:00 Sun 17
12:00 Sun 17
T2_Region_RAL (LCG) [110k]
T2_Region_KIT (LCG) [94.8k]
T2_Region_FNAL (OSG) [80.4k]
T2_Region_IN2P3 (LCG) [57.1k]
T2_Region_ASGC (LCG) [51.7k]
T2_Region_PIC (LCG) [37.3k]
T2_Region_CNAF (LCG) [26.0k]
T1_LCG_1 [3.17k]
RelVal [842]
LCG7T1 [191m]

Figure 8.4: Prodmon Freeplot *view*, showing data from April 2011. On the left hand side are the user's open plots (cached versions of all of which are available allowing rapid switching), with red highlight showing the currently open plot. The controls in the top centre specify the basic plot options (quantity, grouping, plot type, etc), and allow the plot to be closed or new plots opened. The buttons on the right hand side can be expanded into panels for controlling the time period, grouping, filtering, resolution and download in alternative formats.

- The type of plot to generate; bar, pie, cumulative or baobab (Section 8.4). Some quantities or groupings may not make sense for certain plots.

The expandable buttons to the left allow finer control of the plot:

- Change the plot interval (from the default 48 hours). Durations of up to a year are supported, although granularity is reduced from hourly to daily for times longer than 168 hours/one week[21]. It is also possible to specify absolute intervals rather than relative to the current time.
- Change the data grouping. For instance, the basic group-by-site option can be modified to group by site tier (Section 4.4.3), country or both.
- Filter the displayed data. This allows both filtering the displayed data by name (using regular expressions[95]), or filtering logical categories, such as only showing contributions from selected (or excluded) sites or datasets.
- Change the image size. An appropriate size is guessed from the reported window dimensions, but for display purposes custom sizes may be desirable.
- Download the current image in other formats (the lossless PNG[22] is served by default, but PDF[23] or SVG[24] might be useful for presentation purposes).

Each time the user makes a settings change client-side, the javascript application running in their browser sends the change back to the server (as a GET request with the data encoded in the URL, rather than as POST), which validates the request and, if appropriate, updates the state of the user's session server side. The response to this request then contains the complete state of the user's session, encoded as JSON. The client-side application interprets this and rebuilds the user's page. Any new images (ie, plots) are requested from the server at this point. This means that the page stays responsive, and shows, eg, a "working" animation for a new plot request or rescales the existing image for a large image request, while data is collected and plots rendered server-side (potentially a long process if large queries or long data durations are involved).

[21] The reduction in granularity is imposed by the schema of the Dashboard database, rather than the being organic to Prodmon.

[22] **P**ortable **N**etwork **G**raphic format[96].

[23] **P**ortable **D**ocument **F**ormat.

[24] **S**calable **V**ector **G**raphics.

Heavy use of caching is made in the plotting part of *Prodmon*, since all the steps involved (fetching data, parsing it and rendering plots) are expensive[25]. A typical plot containing 48 hours of data takes $4 - 6$ seconds[26] for data to be fetched from *Dashboard*, parsed and rendered, whereas it can be served from the cache in < 100 ms. Caching is implemented as a central overview service, although the design was primarily motivated by the needs of *Prodmon*. It is important to note that this isn't quite the same as caching a particular URL (a generic function that HTTP frameworks tend to provide by themselves). In this case the same URL is likely to yield a different result after a short period of time, and in order to prevent client-side caching the actual request URLs are appended with an extraneous parameter (the Unix time) in order to ensure they make a fresh server request for the image each time[27]. Cached data is stored in a timed mapping with a thread which clears out old data when it expires, or if the volume of data in the cache exceeds a threshold.

Since a large number of possible plots use the same underlying data request (eg, filtered versions of the same underlying plot, different types of plot on the same data), the XML received from the data service is cached after deserialisation. The requested plot is then rendered (using the matplotlib[97] library), and the resulting image cached before being returned to the user. The lifetime of data in the cache is determined by the underlying data granularity, which is hourly for data less than one week old and daily for older data.

Experimental support for HTML *ImageMap*[98] overlay elements was added during development, allowing users to quickly select filters by clicking on the appropriate parts of an image. While this proved successful on simple plots, on complex plots with curved edges and hundreds of areas browser support proved inconsistent and the feature was consequently disabled.

8.4 Filelight - Site Storage Visualisation

The *Filelight* workspace was designed both as a visualisation tool for the data stored in site storage elements, and as a technology demonstrator showing that a fully-functional

[25] Clients automatically refresh every five minutes, and since all plots are typically valid for at least an hour any one plot is generally guaranteed to be fetched at least ten times.

[26] For longer durations this increases significantly, with a 100-day plot taking about a minute.

[27] It is in theory possible to achieve this with HTTP cache-control headers, but this "quick and dirty" solution removes the need for a framework-level ability to control returned headers.

workspace could be implemented relatively quickly by re-use of the caching and plotting infrastructure developed for *Prodmon*.

The basic structure is very similar to that described for *Prodmon* (Section 8.3), with a *workspace* handling plot configuration and a *source* handling the plotting, with both parsed JSON from the data service and produced plots being cached.

The storage element for a Tier-2 site typically contains $O(1PiB)$, organised in $O(100)$ real and Monte-Carlo datasets. The locations of datasets are known centrally, both by DBS[99][28] and PhEDEx, but no central system provided a graphical overview of the data present at a site, to give administrators a clear view of how their space is allocated.

The GNOME and KDE desktop environments include the utilities *Baobab*[100] and *Filelight*[101] respectively, which allow the file system to be displayed as a hierarchical, segmented pie chart, with subdirectories appearing as outer rings within the arc of their parent directories. This is a very useful visualisation for quickly understanding disk utilisation, and the *Filelight* workspace applies this concept to storage elements (or at least, the centrally-known datasets they contain).

Filelight provides two *views*, *Site* (Figure 8.5), which shows the data located at one site, and *Group* (Figure 8.6) which shows the locations of data managed by analysis groups. Data for both comes from the PhEDEx `datasvc` API. Although the plots could ultimately be generated from the LFNs[29] of dataset files, we can produce a more interesting range of plots.

For each block of data, a number of properties can be obtained either as metadata in the JSON document or by decomposing the document name, such as the primary dataset, production era, simulated machine conditions or whether the block is locally complete. The Filelight workspace lets users rearrange the displayed hierarchy with any combination of (or subset of) the block properties. It is also possible to filter datasets selectively, using regular expressions.

Filelight plots are slow to generate due to the size of the JSON document listing all blocks at a site. For the larger Tier 1 sites, this may exceed $50MiB$ in size[30], and in the worst case take almost 2 minutes to fetch, parse and render. The overall latency as seen

[28] **D**ataset **B**ookkeeping **S**ervice

[29] **L**ogical **F**ile **N**ame, global abstract file names for CMS managed data files which are mapped to Physical File Names (PFNs) at each site.

[30] Deserialisation requires around ten times as much memory as the raw document and takes about $1sec/MiB$, limiting the number of concurrent requests Filelight can handle. This also proved a significant problem in DAS (Chapter 9).

by the user is typically 45% fetching from the data service, 35% parsing and transforming the data and 20% rendering the plot. However, as with Prodmon the results are cached and can subsequently be accessed quickly until expiry[31].

8.5 Plotfairy - Plotting Service

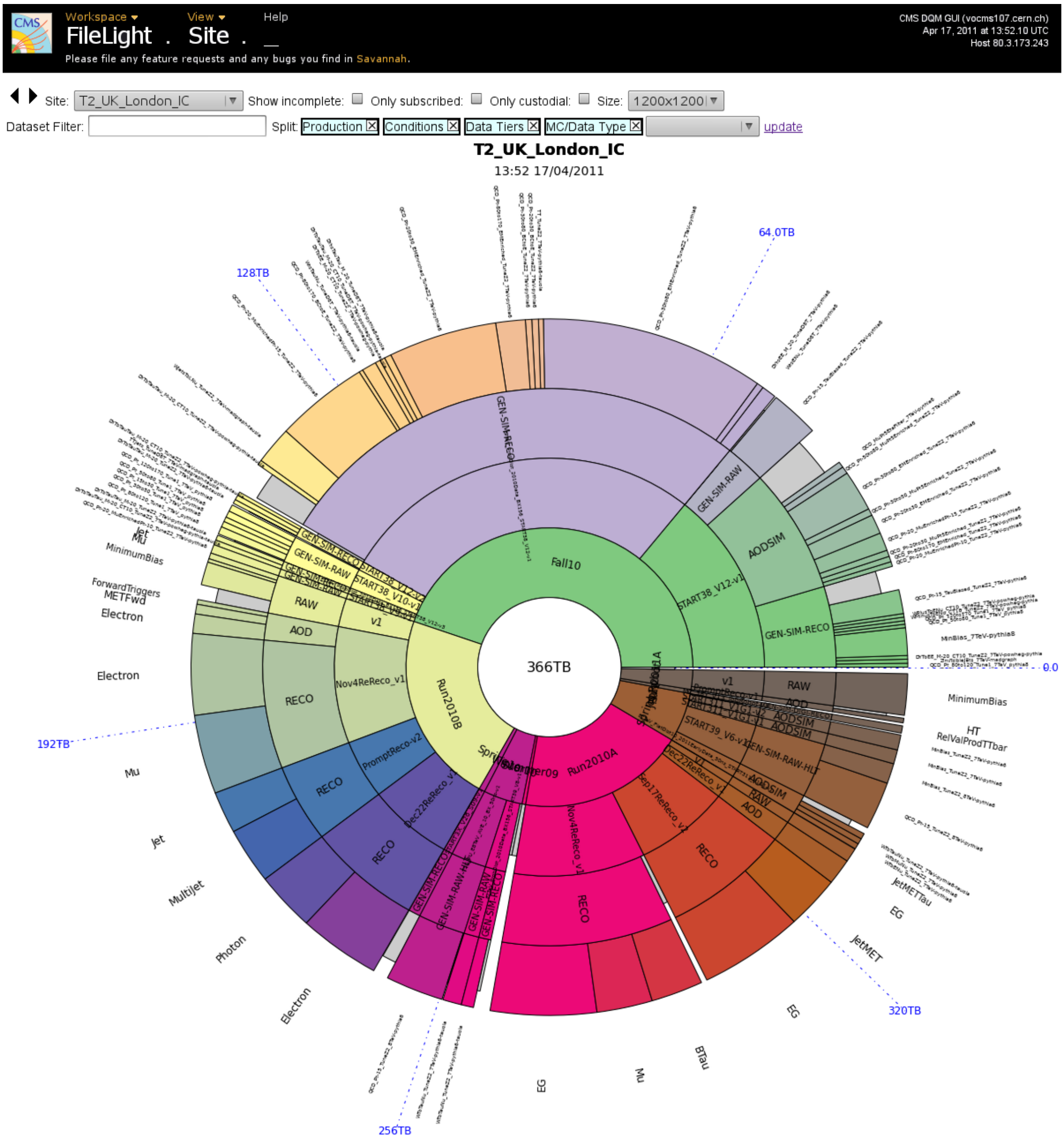
In Overview, plotting was an integral part of the web service, running in the same server that handled client state and accessed back-end data services. The plotting code, while relatively independent of any one Overview workspace was however tied into the framework. A number of other CMS web projects also required similar plotting services, and it was considered useful to have plotting implemented as a single central service rather than require a number of competing implementations of the same basic concept. Plotfairy[32] was intended to provide a standalone plotting server (or at least a service that could be run in a CherryPy instance without any code-level integration), with plots entirely specified by the URL (for a HTTP GET request) or the body of a HTTP POST request. This also aimed to insulate web applications from potential plotting problems, since the compiled extension modules for *matplotlib* were found to leak memory under some circumstances, which would ultimately lock up the rest of the server (whereas a stateless plotting server is easy to periodically restart, if necessary).

Although initially designed with the Overview plotting code in mind, Plotfairy ended up largely being written from scratch, to provide a cleaner and more consistent implementation. Plotfairy ultimately provided:

- Pie and Baobab radial plots (as per Section 8.4)
- Bar (numerical or labelled)
- Two-dimensional heat maps
- Scatter plots
- Sparklines (small unlabelled line charts intended to be used inline with text)
- Wave (interpolated stacked bar chart centred on the x axis)

[31] The PhEDEx `datasvc` reports the served data to be valid for only 5 minutes, but Filelight assumes that the possible changes are small within an hour and caches data for the latter period.

[32] From the name of the URL endpoint in Overview that served non-session data.



Workspace ▾ View ▾ Help

FileLight . Site . _

Please file any feature requests and any bugs you find in Savannah.

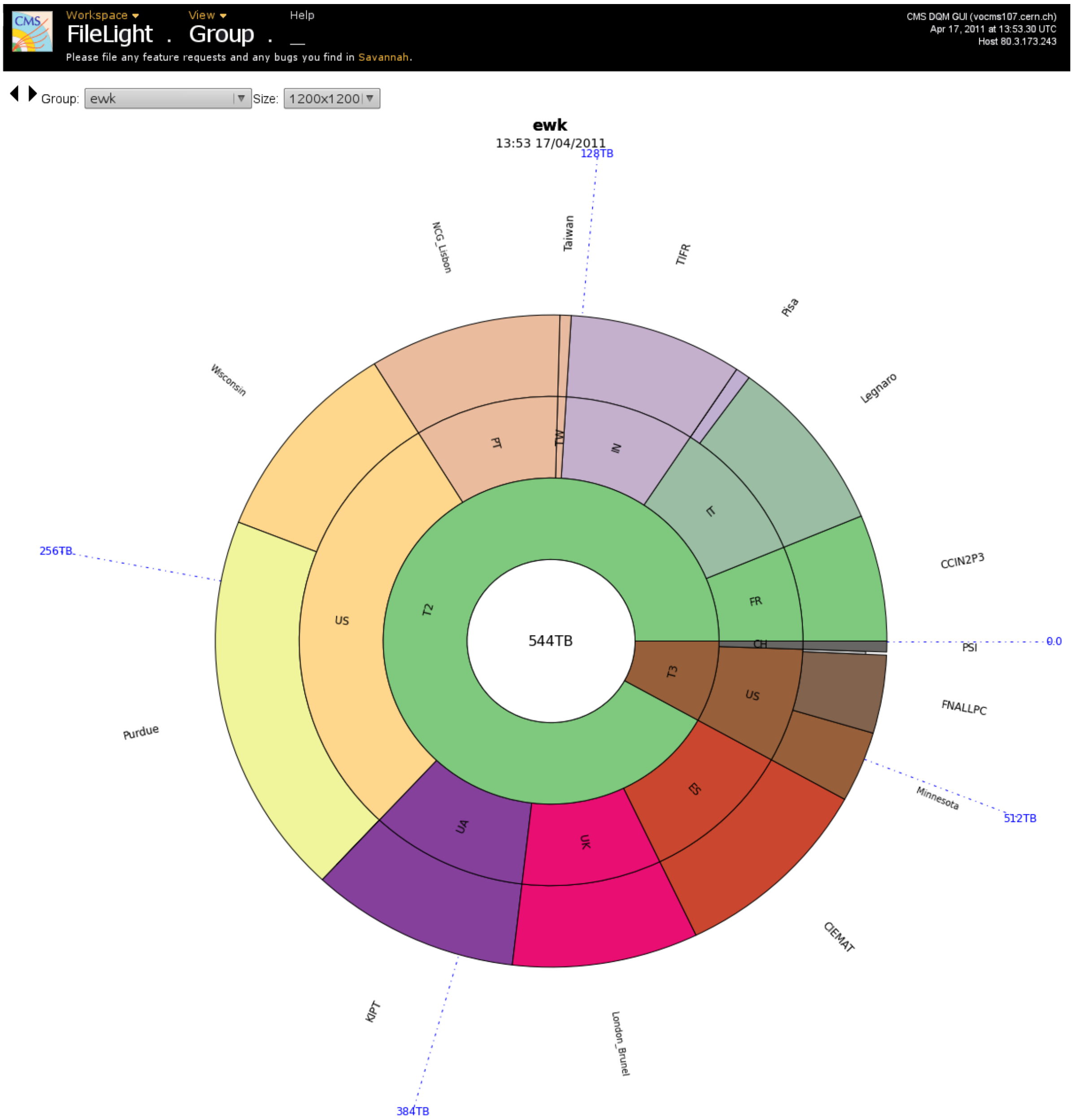
CMS DQM GUI (vocms107.cern.ch)
Apr 17, 2011 at 13:52.10 UTC
Host 80.3.173.243

Site: T2_UK_London_IC ▾ Show incomplete: □ Only subscribed: □ Only custodial: □ Size: 1200x1200 ▾

Dataset Filter: Split: Production ☒ Conditions ☒ Data Tiers ☒ MC/Data Type ☒ ▾ update

T2_UK_London_IC
13:52 17/04/2011

Figure 8.5: Filelight Site *view*, showing data stored at the Imperial College storage element in April 2011. The controls at the top control the site being viewed, the data selected for viewing (by categories or regular expressions) and the resolution. The boxes at the bottom right of the panel set the hierarchy used for drawing, which can be added, removed and re-ordered. These images are typically rendered at $1200x1200px$ or larger and hence are somewhat compressed to fit the page.



Please file any feature requests and any bugs you find in Savannah.

CMS DQM GUI (vocms107.cern.ch)
Apr 17, 2011 at 13:53:30 UTC
Host 80.3.173.243

Group: ewk

Size: 1200x1200

Figure 8.6: Filelight Group *view*, showing the geographic distribution between computing sites of data owned by the CMS electroweak group in April 2011.

The design was made as modular as possible employing mixin-classes for all shared functionality (*TitleMixin*, *StyleMixin*, *XNumericAxisMixin*, etc). Plotting was defined as a series of steps (parameter validation, data extraction and transformation, axes construction, pre-plotting, plotting, post-plotting and finalisation), and each mixin would define operations to occur at the appropriate time. For relatively simple plot types, this resulted in a fully-customisable plot class[33] that required only $O(10)$ lines of code specific to that plot type.

Each mixin included a set of parameters, along with the expected type and acceptable ranges, defaults and if necessary more complex validation functions. These are introspected and used to dynamically generate documentation for each available plot type, and generate images containing a specific error message in the case that bad data have been supplied.

Plotfairy is being used in-production by the Tier-0 monitoring system[34].

[33] The final plot classes were not actually the direct result of multiply-inheriting the mixin classes, but rather were constructed by a custom metaclass that reorganised all the operations into functions within a single class.

[34] `t0mon`

Chapter 9

Data Aggregation System

9.1 Introduction

In Chapter 8 we introduced a subset of the CMS data service ecosystem. Associated with approximately $5PiB$ of raw and reconstructed data to be taken each year is around $1TiB$ of metadata, including the machine conditions, luminosity measurements and locations of the data. The systems for managing each class of metadata were developed separately and involve a plethora of different underlying storage mechanisms and end user access methods. Prior to LHC running, this situation was largely acceptable; users generally only needed to query simple information about dataset locations and segmentation.

However, following start-up it became necessary to perform queries across multiple data services, such as looking up the luminosity, machine conditions and software configuration used for a reconstructed dataset, which is currently only possible in a piecemeal fashion.

The Data Aggregation System (DAS) is intended as a "search engine" for CMS, to provide a single point of access for users, from which they can perform queries on one or more underlying services (without needing knowledge of how data is split between underlying services), and provides a caching layer for the data services.

This chapter provides an overall description of the DAS implementation[1]. The author worked mainly on the parser, analytics and keylearning aspects.

[1] DAS 0.5.x series.

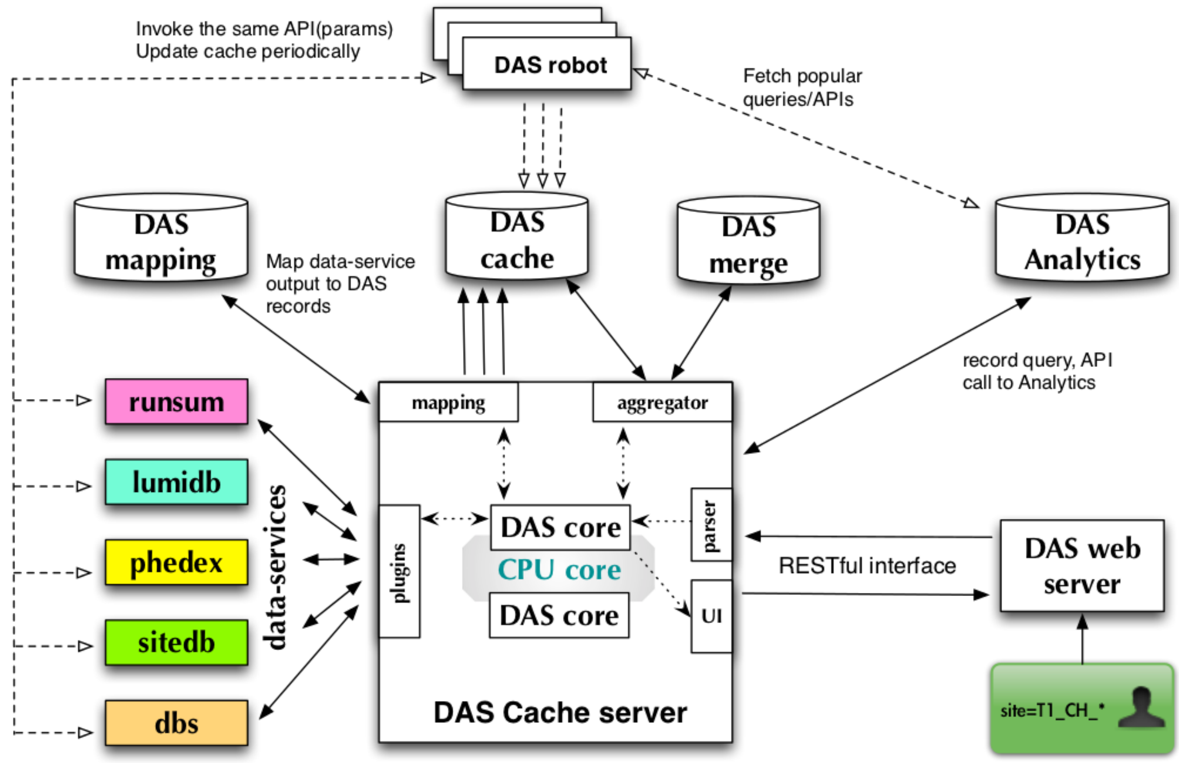


Figure 9.1: Diagram of the basic architecture of DAS, showing the relationships between the web- and cache-servers, data services, and the MongoDB storage. [103].

9.2 Architecture

DAS is designed as a number of independent components, which are scalable from the whole system running on a single node to multiple instances of each individual component running on separate nodes. The broad architecture is shown in Figure 9.1. It is implemented as a number of separate, multithreaded python daemons running on Linux[42, 43] nodes, plus one or more MongoDB[102][2] instances for persistent storage. Users access the service via either a web interface or a command-line interface.

Common data concepts across multiple data services are represented by a single *DAS key*. Users making a query need to know the appropriate *DAS key* for the information they are seeking. Where possible, the same notation is used as the underlying services, but this is not always possible (for instance, the *DAS key* **run_number** is variously called **Run** and **runNumber** by other services).

Figure 9.2 shows the relationships between a subset of data services and *DAS keys*.

[2] From "humongous".

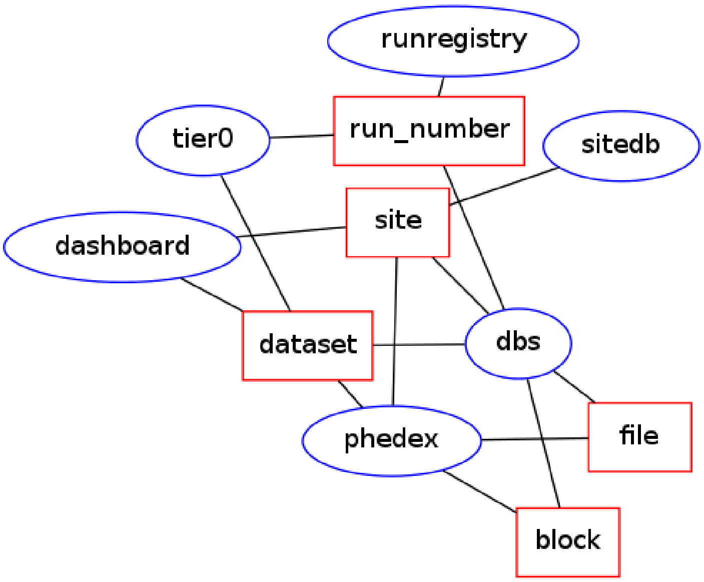


Figure 9.2: Diagram showing a subset of the data services used by DAS (blue ellipses) and subset of the DAS keys (red rectangles) representing common concepts between them.

It will be useful to define briefly some terminology before discussing aspects of DAS in detail:

- **User**, principally meaning a member of CMS wishing to access data, but users may also be other automated systems.
- **DAS key**, the keys by which DAS is indexed and searched, representing a concept found in one or more *data services*.
- **Data service**, any CMS (or other) system from which DAS acquires data in a structured manner. Data services provide one or more **APIs**[3] which DAS accesses.
- **Document**, structured data originating from a *data service* and stored in the *cache*.
- **Cache**, the set of documents from data services stored in MongoDB, divided into the **Raw Cache** (invidual responses from data services) and the **Merge Cache** (documents obtained by combining one or more documents from the *raw cache*).
- **Query**, a request from a user for information in DAS, consisting of conditions that select a set of data and possibly operations to be performed on the selected data.

[3] **A**pplication **P**rogramming **I**nterface

- **Mapping**, the relationship between DAS keys and the names used by data services, allowing translation between the systems.
- **Workflow**, the process necessary for DAS to respond to a *query* from a *user*.

9.2.1 Data Services

The fundamental purpose of DAS is to access underlying data services. The idea of a "data service" is wholly abstract; it is any computational resource, which can be called with *DAS keys* (appropriately transformed) as arguments, and returns data which can be transformed back into DAS format, and obeys certain invariants such as consistent handling of wildcards. Data services could be, for instance, a relational database cursor, the standard output from a subshell command or a web-page scraper, but in practice most are web (HTTP[4]) services which accept arguments as URL parameters and return data formatted as either JSON or XML. Regardless of the underlying data format, it is transformed into JSON representation within DAS.

Each data service is described by a *mapping*, which defines, for each available API call provided by the service, the mapping from DAS keys to input parameters, the mapping from the output back to DAS keys, and supplemental metadata such as the valid lifetime of returned data[5]. For HTTP data services using a common format, no additional code is required (whereas for other input types, a plugin class must be written that performs whatever invocation is necessary, and handles the input and output data transformations. The mapping also includes presentation information, which can be used to produce more human-friendly output. Mappings are defined using the YAML[6] format.

The current DAS implementation includes mappings for the major CMS data services

- PhEDEx (data transfer and replication between sites)
- DBS (dataset metadata and heritage)
- SiteDB (computing site metadata)
- Dashboard (grid job information)
- RunRegistry (LHC run information)

[4] **H**yper**t**ext **T**ransfer **P**rotocol

[5] If not supplied along with the response.

[6] **Y**et **A**nother **M**arkup **L**anguage, a superset of JSON designed to be more human-readable.

- Tier-0 Monitoring (reconstruction of newly acquired data)

A number of other services have also been implemented, such as the CERN *xwho* person lookup system, but these have only been used for testing purposes thus far.

9.2.2 Storage

Storage for DAS is handled using the MongoDB document store, which natively stores JSON data structures. DAS communicates with MongoDB using sockets, and can work with anything from a single instance on the same node to a cluster of sharded[7] instances. A single namespace is used to store all DAS data, both the primary cache collections (*raw* and *merge*), as well as logging, analytic information, and key mapping and learning.

For the DAS use-case, a document store proves much more suitable than a traditional relational database; the data we wish to store is generally hierarchical rather than flat, we do not know the data schema until run-time, and we wish to be able to look-up documents by the values of nested fields. Some work was done testing the contemporary CouchDB[104] document store, but we found it was less suitable for our purposes, being less scalable across multiples nodes and less easily able to perform the queries we required.

DAS is exposed to a potentially large amount of data; approximately $O(10^6)$ records and $O(1TiB)$ of data per running year. Further, the raw data itself is a small fraction of the total disk space required by DAS, since the *raw* and *merge* collections require a large number of indexes[8] for efficient lookups which are typically significantly larger than the actual data. The hardware requirements are mitigated however as the data can be replaced from the source services (so RAID[9] is required only for performance, rather than backup), and the storage can be configured with collections of capped size that automatically drop the oldest documents[10] if space is exhausted.

MongoDB imposes a size limit (when encoded using the internal BSON[11] representation) of $4MiB$[12] per document. Some services (particularly PhEDEx) occasionally return

[7] Where the document store is spread over multiple nodes, each is termed a *shard*.

[8] For example, on the expiry time, query hash, keys in the query, and the internal ID used by DAS to identify each record.

[9] Redundant Array of Inexpensive Disks

[10] Where age is given by insertion order, rather than intrinsic age of the data in each document.

[11] Binary JSON, a more compact and more easily parsed form for JSON-type documents, albeit lacking human-readability.

[12] Although this is being raised to $16MiB$ in current development versions, which should handle all documents except the file lists for Tier 1 sites.

documents exceeding this size. MongoDB provides a storage mechanism called GridFS for storing outsize documents in multiple chunks, but queries involving such documents become significantly more expensive.

9.2.3 Cache Server

The core workflow of DAS takes place on one or more *cache servers*. The cache server owns a pool of worker threads which handle user queries, with each query being wholly handled by a single worker thread. Typically two threads are run per physical CPU core available to DAS, since the workflow involves repeatedly waiting for responses from storage and hence will not individually saturate a core.

Queries originate from either the web frontend or a command-line tool, and are converted by the parser (Section 9.2.4) into a structure representing the query conditions and any additional operations to be performed. If a query with identical conditions is already in the DAS merge cache, and has not expired, it is returned immediately to the user. If no identical query exists, the cache is checked for queries with the same set of condition keys and values that are a superset of the requested values[13].

Queries for which no matching or superset query exists require calls to the underlying data services. The set of data service APIs which can answer a given query is those for which the set of required DAS keys is equal or a subset of the set of DAS keys specified in the query conditions. For each API call required, the raw cache is first checked for any still-valid[14] results from an earlier call to the same API with the same or superset conditions[15], negating the need for a fresh call. Otherwise, the API call is made and the received data parsed and transformed according to the API-specific mapping record, using generator expressions to reduce the concurrent memory footprint. Since some services require a significant wait while they generate data in response to an API call (performing their own database queries, etc), making each API call serially represents a significant performance bottleneck. Work is underway to parallelise this part of the workflow.

[13] For example, the query `site=T2_UK_London_IC` is a subset of the query `site=T2*`.

[14] Some data services specify a time-to-live as part of their responses. For those that do not, a per-API value is specified in the mapping based on an estimate of how quickly the data changes, with values between 10 minutes and 24 hours.

[15] Since different API systems do not necessarily obey the same wildcard semantics (eg, whether a `*` character is explicitly required), data services need to supply a superset test if it does not obey shell-like behaviour.

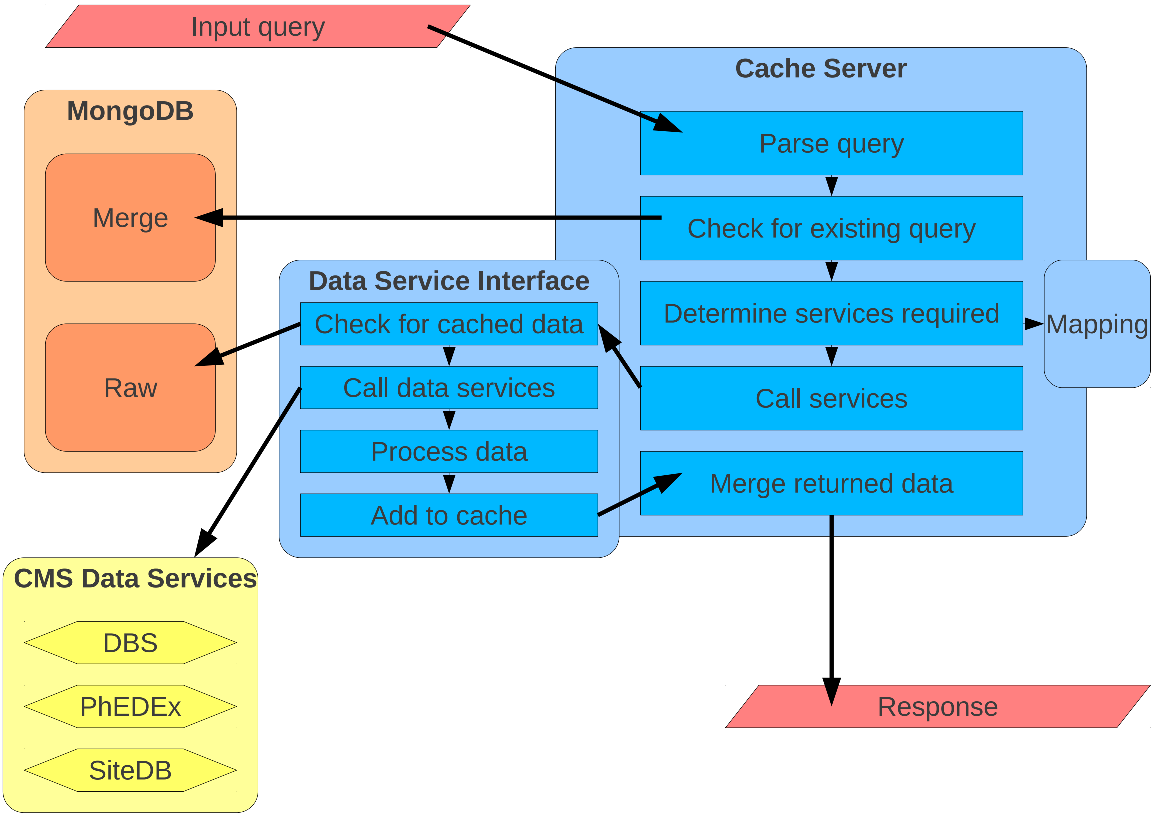


Figure 9.3: Diagram showing the DAS workflow. The input query is parsed and the merge cache checked for an existing answer. If this is unavailable, the required data services are determined from the mapping. The data service interface first checks for existing responses to each API call, and makes the call if no data is available. The returned data is parsed and added to the raw cache, before finally all the required documents are combined into a single document, which is added to the merge cache and returned to the user.

Once all API calls have either been made (or valid data determined already to exist), all of the relevant records in the raw cache are joined into a record in the merge cache, which is then returned to the user. The workflow is illustrated in Figure 9.3.

Several steps in this process (such as starting a new query, successfully finding data in the cache and each API call made) are recorded in the analytics collection to allow internal performance to be optimised.

9.2.4 Query Language

DAS uses a text-based query language, structured in the style of pipelined operations on the UNIX command line. Earlier designs proposed a more SQL[16]-like syntax (`select X where Y`), drawing on an existing query language used by DBS (DBS-QL[105][17]), but it was felt that this made some complex operations (aggregation of results, map-reduce) difficult to express. The shell-like syntax (`X | grep Y`) should also be more familiar to the majority of users, all of whom will have worked with the UNIX shell but few of whom will have worked with SQL.

A query consists of a selection, optionally followed by one or more operations on the data, separated by the pipe character:

- *Conditions*, consisting of a list of DAS keys, possibly followed by conditions. The set of keys specified determines which APIs are called (or retrieved from the cache) to answer the query. Conditions can be specified as a simple match, possibly with wildcards (`dataset=/foo/bar`), membership of a list (`site in [T1_UK_RAL,T2_UK_London_IC]`), a range (`run_number between [100000,130000]`) or for data including time information, a relative interval (`jobsummary last 24h`). Keys for which no condition is given are implicitly a wildcard query on that key.

- *Filter*, consisting of a list of one or more filter operations. The main filter is `grep`, which removes all properties from the returned data not matching the given string. For example, `site | grep site.name` would first fetch data on all sites (since `site` is interpreted as `site=*`), then only return the `site.name` element from each.

- *Aggregator*, consisting of a list of one or more aggregator operations. These specify a simple function that runs on every property of a given name within the data specified by the query. For example, `dataset | sum(dataset.size), avg(dataset.size)` would return the total size and average size of datasets known to DAS. Aggregation currently only works on numerical properties, but there is no intrinsic reason why aggregation could not be performed on other types of keys, if useful aggregations were found.

- *Map-Reduce*, consisting of the name of a custom function for performing a more complex analysis on the data returned by the query (`dataset=/foo* | function_name`).

[16] **S**tructured **Q**uery **L**anguage, as used by traditional relational database systems.

[17] While DBS includes a complex syntax, the vast majority of queries are a (possibly wildcarded) dataset name, which is implicitly converted to `select dataset where dataset.name=foo`.

> The function must already by specified on the server by an operator, since it runs server-side and hence needs to be approved before use.

The parser for DAS is built using the PLY[18][106] library. This performs an equivalent function to the `lex`[107] and `yacc`[108] tools for compiled languages. Regular expressions describing each token are built into a tokenizer that converts the query string into a series of basic tokens[19]. The list of tokens are then converted into a meaningful form (an AST[20]) using a finite state machine (constructed by PLY from the query language grammar).

A graphical representation of the grammar is shown in Figure 9.4.

Parsing is a relatively expensive operation, and it is expected that a large number of queries will be identical, or at least extremely similar[21]. Before parsing, the query string is normalised by removing extraneous whitespace, and a hash value computed using the MD5[109][22]. Previously parsed queries are stored in a collection and the cached result (or cached error, as appropriate) is retrieved if available.

9.2.5 Web Server

Users primarily interact with DAS through a web interface. User requests arrive at a front end web server (using the *CherryPy* server), which serves a static set of pages. When the user makes a query, it is sent (using AJAX via the YUI[110] javascript library) to the web server. The web server hashes the query and immediately responds to the user with the hash of the query (as described in Section 9.2.4), ensuring the client remains responsive[23]. The web server passes the query on to the cache server, which processes it when a worker thread becomes available.

The query is indexed by its hash value throughout the lifecycle of the query, which has a side-effect that concurrent identical queries[24] are only processed once, and a previously-

[18] **P**ython **L**ex-**Y**acc

[19] For instance, the earlier example `site in [T1_UK_RAL,T2_UK_London_IC]` is tokenised to `das_key operator_in start_list string comma string end_list`.

[20] **A**bstract **S**yntax **T**ree

[21] Principally those for popular datasets.

[22] Although this algorithm is now deprecated for the security use due to the relative ease of collisions being found, it is not used for any security-critical application here and the high performance is a benefit.

[23] This is essentially the same REST model as used by Overview plotting (Section 8.3).

[24] Such as those resulting from users mashing keys like demented monkeys.

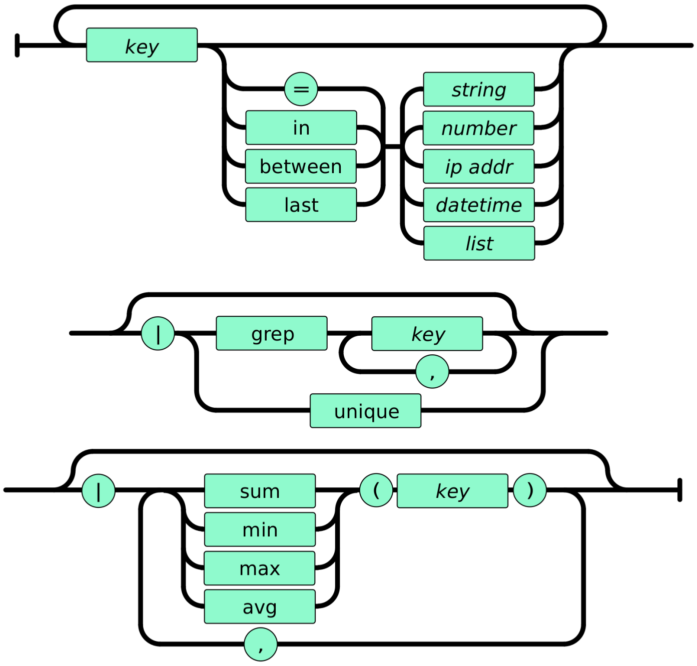


Figure 9.4: "Railtrack" diagram representing the finite state machine for DAS-QL parsing. The three (concated) sections represent the *conditions*, *filter* and *aggregation* sections of an overall query.

processed query for which none of the data has yet expired will return immediately. The client periodically polls the web server for the status of this query hash, and when the query is complete the client requests a results page[25], shown in Figure 9.5. Unlike Overview (Section 8.2), where the server almost entirely serves JSON and the pages are built by the client, DAS serves complete HTML pages, templated using the Cheetah[111] library.

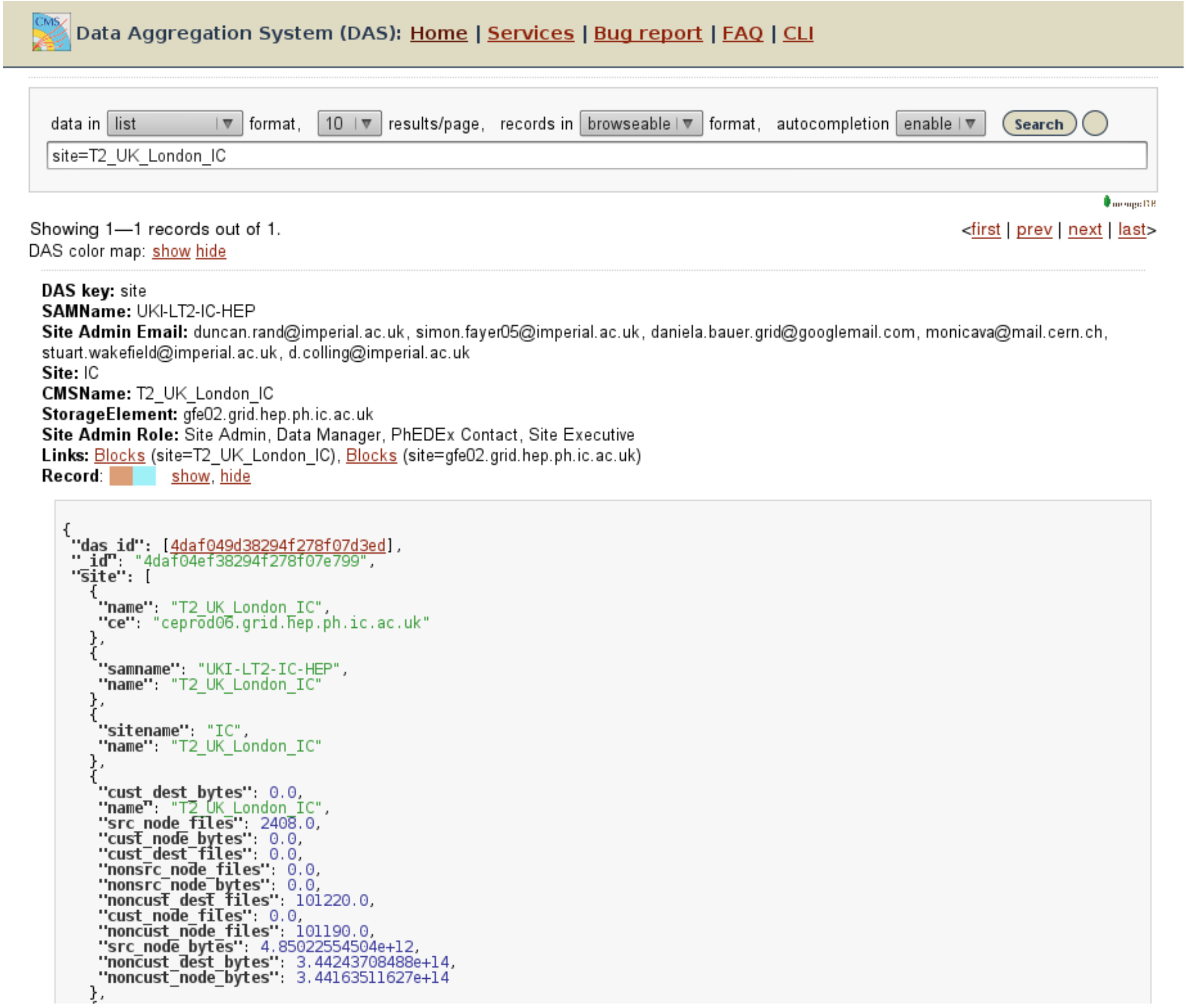
9.2.6 Analytics

The term "analytics" is something of a misnomer, as the analytics system is a general purpose task scheduler for running complex tasks on the DAS datastore, which encompasses high-level data-mining, recurring analysis for purposes of optimisation and carrying out clean-up and maintenance tasks.

The analytics system consists of a daemon, which owns a set of worker processes on which analytics tasks can be performed. A simple webserver is also attached[26], to

[25] This is a pull model, with the server making no explicit notification that a task has completed.

[26] In production this would only be accessible to operators by tunnelling to the appropriate system.



Data Aggregation System (DAS): Home | Services | Bug report | FAQ | CLI

data in list format, 10 results/page, records in browseable format, autocompletion enable Search

site=T2_UK_London_IC

Showing 1—1 records out of 1. <first | prev | next | last>

DAS color map: show hide

DAS key: site
SAMName: UKI-LT2-IC-HEP
Site Admin Email: duncan.rand@imperial.ac.uk, simon.fayer05@imperial.ac.uk, daniela.bauer.grid@googlemail.com, monicava@mail.cern.ch, stuart.wakefield@imperial.ac.uk, d.colling@imperial.ac.uk
Site: IC
CMSName: T2_UK_London_IC
StorageElement: gfe02.grid.hep.ph.ic.ac.uk
Site Admin Role: Site Admin, Data Manager, PhEDEx Contact, Site Executive
Links: Blocks (site=T2_UK_London_IC), Blocks (site=gfe02.grid.hep.ph.ic.ac.uk)
Record: show, hide

```
{
 "das_id": [4daf049d38294f278f07d3ed],
 "_id": "4daf04ef38294f278f07e799",
 "site": [
   {
    "name": "T2_UK_London_IC",
    "ce": "ceprod06.grid.hep.ph.ic.ac.uk"
   },
   {
    "samname": "UKI-LT2-IC-HEP",
    "name": "T2_UK_London_IC"
   },
   {
    "sitename": "IC",
    "name": "T2_UK_London_IC"
   },
   {
    "cust_dest_bytes": 0.0,
    "name": "T2_UK_London_IC",
    "src_node_files": 2408.0,
    "cust_node_bytes": 0.0,
    "cust_dest_files": 0.0,
    "nonsrc_node_files": 0.0,
    "nonsrc_node_bytes": 0.0,
    "noncust_dest_files": 101220.0,
    "cust_node_files": 0.0,
    "noncust_node_files": 101190.0,
    "src_node_bytes": 4.85022554504e+12,
    "noncust_dest_bytes": 3.44243708488e+14,
    "noncust_node_bytes": 3.44163511627e+14
   },
```

Figure 9.5: Screenshot showing the results page for DAS query `site=T2_UK_London_IC`. The query resulting in underlying calls to PhEDEx and SiteDB, and the output shows a number of elements (for which presentation information exists) have been laid out in a table, while the complete raw information is shown (truncated) as HTML-prettified JSON.

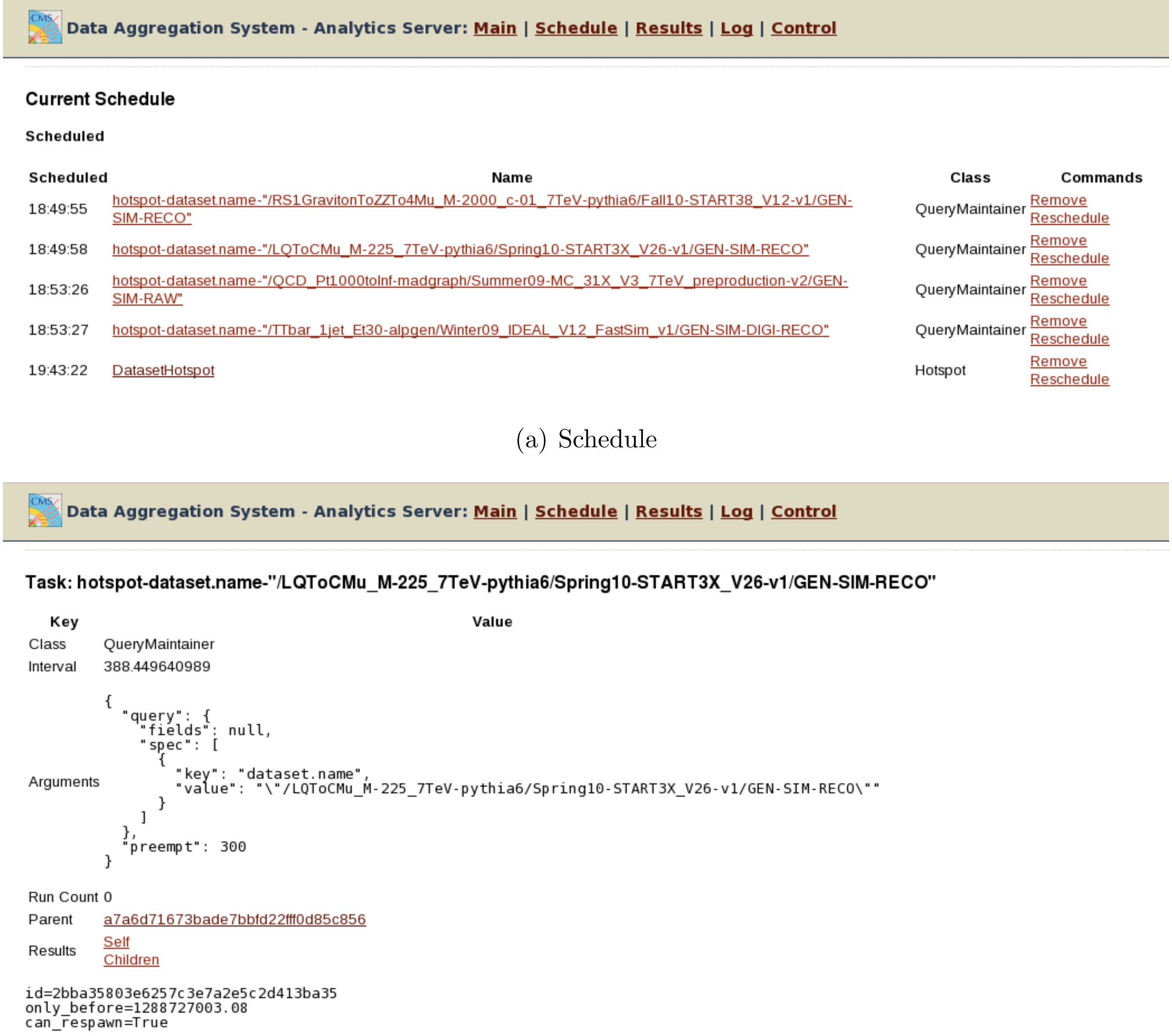
provide information and accept commands. At implementation level, tasks are python classes which accept a DAS and logging instance, and one or more parameters, perform some interaction with the database and then return a well-formatted output. They can either be specified in a configuration file for the analytics server or added dynamically using the web interface or command line tools. Tasks are executed (or at least, added to the execution queue) at fixed intervals, and by their return values can control their own respawning and, if necessary, request the spawning of other tasks[27].

At the maintenance level, the analytics framework allows regular expunging of expired data to be performed on the various DAS collections. Some clean-up is performed each time a DAS query is performed, but this is an inefficient way of performing expensive maintenance, since the time structure of queries cannot be guaranteed and since only a single thread is used to service each query, performing clean-up synchronously with user operations adds unnecessary latency.

Optimisation tasks are intended to ensure that as many queries as possible are served directly from the cache, without any data service calls being made. The latency to serve a simple dataset query (`dataset=/foo/bar...`) is $O(100\text{ ms})$ from the cache, compared with $O(60\text{ s})$ to query all the necessary data services and merge the result. Examining the logs for both DBS and CRABserver, we find that approximately 50% of queries pertain to the top 15% of datasets. By ensuring the top datasets are kept permanently in the cache, a significant fraction of user queries can be handled expeditiously.

The default analytics configuration includes a number of so-called “hotspot” analysers. In each case the behaviour is similar, but the parameters of a different DAS key are analysed. The analysers typically run every 4 hours, counting all arguments to the relevant DAS key and emitting a summary document. All the summary documents for an appropriate period are loaded, and used to identify the top datasets by an appropriate metric. The appropriate period for this moving average varies by key; for datasets 30 days is used, since this is long enough for an accurate average but shorter than the periods between data reprocessings and MC production campaigns. For each selected dataset, a cache-populating task is spawned. The cache populator fetches all data for the requested dataset, checks the shortest expiry time and reschedules itself shortly before the data will expire, until the next time the hotspot analyser runs, at which point all existing child tasks are killed and new ones spawned.

[27] The task scheduler prevents any task continually respawning itself or fork-bombing the system, and will stop a task after a specified number of exceptions have been raised.



Data Aggregation System - Analytics Server: Main | Schedule | Results | Log | Control

Current Schedule

Scheduled

Scheduled	Name	Class	Commands
18:49:55	hotspot-dataset.name-"/RS1GravitonToZZTo4Mu_M-2000_c-01_7TeV-pythia6/Fall10-START38_V12-v1/GEN-SIM-RECO"	QueryMaintainer	Remove Reschedule
18:49:58	hotspot-dataset.name-"/LQToCMu_M-225_7TeV-pythia6/Spring10-START3X_V26-v1/GEN-SIM-RECO"	QueryMaintainer	Remove Reschedule
18:53:26	hotspot-dataset.name-"/QCD_Pt1000toInf-madgraph/Summer09-MC_31X_V3_7TeV_preproduction-v2/GEN-SIM-RAW"	QueryMaintainer	Remove Reschedule
18:53:27	hotspot-dataset.name-"/TTbar_1jet_Et30-alpgen/Winter09_IDEAL_V12_FastSim_v1/GEN-SIM-DIGI-RECO"	QueryMaintainer	Remove Reschedule
19:43:22	DatasetHotspot	Hotspot	Remove Reschedule

(a) Schedule

Data Aggregation System - Analytics Server: Main | Schedule | Results | Log | Control

Task: hotspot-dataset.name-"/LQToCMu_M-225_7TeV-pythia6/Spring10-START3X_V26-v1/GEN-SIM-RECO"

Key	Value
Class	QueryMaintainer
Interval	388.449640989

Arguments

```
{
  "query": {
    "fields": null,
    "spec": [
      {
        "key": "dataset.name",
        "value": "\"/LQToCMu_M-225_7TeV-pythia6/Spring10-START3X_V26-v1/GEN-SIM-RECO\""
      }
    ]
  },
  "preempt": 300
}
```

Key	Value
Run Count	0
Parent	a7a6d71673bade7bbfd22fff0d85c856
Results	Self Children

```
id=2bba35803e6257c3e7a2e5c2d413ba35
only_before=1288727003.08
can_respawn=True
```

(b) Task

Figure 9.6: Screenshots of the DAS analytics web interface. The scheduler shows a *DatasetHotspot* instance scheduled to run infrequently, and several *QueryMaintainer* instances spawned to keep the named datasets in the cache. The task view shows one of those tasks, showing the query it is configured to issue, the interval at which it will do so, and a link to the parent (DatasetHotspot) task.

Figure 9.6 shows an example of a Hotspot analyser running, and the web interface for DAS analytics.

The final class of tasks are for performing queries that are too complex to be expressed in DAS QL, and cannot readily be performed as a single map-reduce task[28]. An example of a complex query that could be performed using analytics might be:

[28] For instance, because it requires historical data or a complex state to be stored.

"Find the number of events analysed for each dataset (*dashboard*); find the number of events in each dataset (*DBS*); find the number of sites hosting each dataset (*PhEDEx*); calculate the usage density for each dataset; identify sites with under-utilised disk and CPU capacity (*SiteDB*); create an advisory list of the most densely used datasets and under-utilised sites that could hold additional copies".

9.2.7 Key Learning

By design, DAS does not know anything about the attributes of each DAS key (although it is assumed that the set of attributes returned by a given query are invariant within the lifetime of the response). The actual information a user wishes to view, however, is usually the values of one or more attributes of a DAS key (eg, when looking up a dataset one may actually wish to know the total disk footprint (`dataset.size`) or the number of events (`dataset.events`)). The names of all the attributes are usually fairly predictable, but if one wishes to include a filter or aggregation step in the query, the exact names of the attributes need to be known.

The key-learning system in DAS is a means of performing introspection on the retrieved data, in order to build up internal mappings of underlying queries performed to the names of the attributes (including both their old and new names, if any transformations are performed by the DAS mapping). This is implemented as an analytics task (Section 9.2.6) which runs $O(daily)$, sampling a random subset of all queries retrieved during the previous epoch and inserting the $query \rightarrow attribute$ mapping into a collection. The found attributes are stemmed (prefixes and suffixes removed) to reduce derived words to their root forms, to make searching for attributes more intuitive.

The information gleaned from key-learning is used to provide an autocompleter for the main DAS search page. If the user pauses while typing (for more than 100 ms), the current query text is sent asynchronously to the DAS web server (Section 9.2.5), which performs regular expression matches against a number of possible patterns. Suggestions are provided for a number of patterns:

- Fragment of DBS QL (`find X...`): suggests to the user how to recast it in DAS QL (providing it is relatively straightforward).
- Raw site or dataset name (`Tx_yy...` or `/dataset...`): converts this into a simple DAS query (`site=Tx_yy...`).

- Valid DAS syntax but a key or attribute unknown to DAS (`foo=bar`): suggests a lexographically nearby DAS key if it exists, or performs a text search for attributes with a similar name and suggests the appropriate DAS key for them.
- Valid DAS query with the user starting to type a filter or aggregator (`site | grep s...`): searches for attribute names, based on those expected to be returned from a query with the given DAS keys.
- Other snippet of text (`foobar`): perform a text search on these terms.

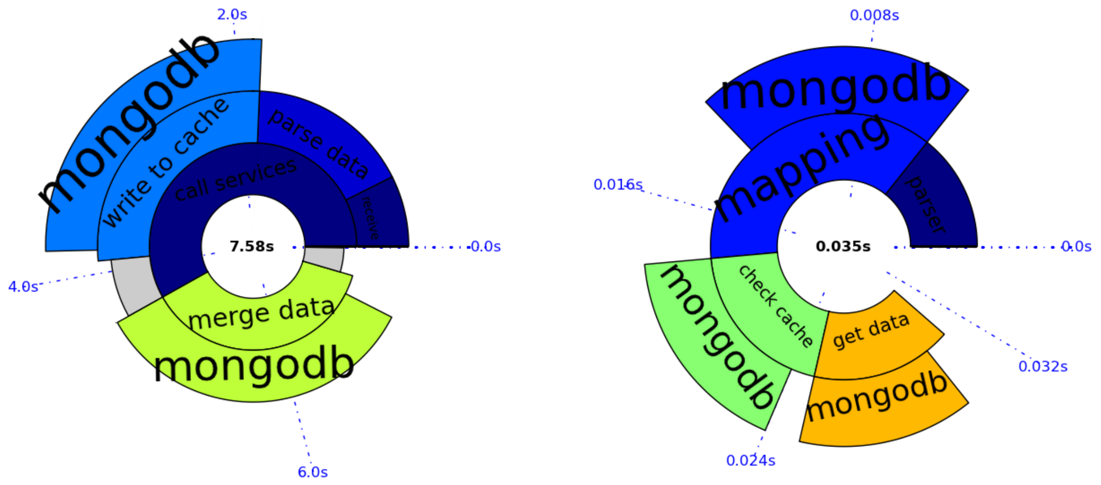
The server responds with a list of possible completions to show, along with styling information allowing suggestions to be presented to the user as purely informational, warnings or definite errors. Autocompletion is intended as a helpful service to enable new users to adapt to DAS and correct common mistakes, but is also noted as an irritation by some users, and as such can be completely disabled (by setting a cookie) if desired.

9.3 Performance

DAS remains a system in development, and as such performance has not been of primary concern so much as building a viable system. However, performance is likely to become a more important issue with deployment and potential large-scale use, thus some examination is necessary.

The use of Python inevitably imposes a performance penalty compared to a compiled language, since every operation requires multiple layers of type-checking and namespace searches by the interpreter. Performance penalties as high as factor 10^3 are often mentioned for purely numeric code in Python compared to C, but the potential performance gain (in terms of user latency) is likely to be significantly smaller in DAS, since time spent waiting for either MongoDB storage or data services is language-independent, and compiled languages lose much of their performance advantage when traversing arbitary data structures (such as the JSON documents which DAS manipulates), as they have to perform type-checking and cannot optimise the operation before run-time.

Within the existing architecture, performance can be gained principally by caching or by converting localised hotspots into compiled code. The former approach is used in several places (for example, to avoid the need to parse identical query strings), while the latter is still largely unused as internal interfaces are not yet entirely stable.



(a) Nothing cached

(b) Everything cached

Figure 9.7: Results of profiling DAS while it processes the query `dataset=/Electron/Run2010B-Nov4ReReco_v1/RECO` (real data used in Chapter 6 and 7), showing the time taken to fully process a completely new query, compared to the time taken to subsequently serve it from the cache. The time shown is that required to fetch the requested data to memory and does not include the overheads of handling web requests and returning the data to the user. Multiple layers show components of operations, such as the time a python function is waiting for a response from MongoDB. It should be viewed with the caveat that profiling Python is not an entirely precise science. The plots are generated using the *PlotFairy* service described in Section 8.5.

Figure 9.7 shows the time consumed by the various internal parts of DAS, and the tremendous difference in execution time between fetching for the first time and serving from cache.

A tool called "queryspammer" was written to generate and request large numbers of random queries for the purposes of performance, analytics and correctness testing. *Queryspammer* consists of three parts:

- *Producers*, which generate queries in DAS Query Language (Section 9.2.4). These range from trivial types which produce simple, repeated queries through to more complex examples which produce queries against multiple DAS keys (using the DAS mapping to determine valid combinations), including filter and aggregation steps.
- *Filters*, which randomly mutate the query in ways designed to mimic the incompetencies of real users, such as mis-spelling key words, missing or doubling spacing, forgetting to quote strings where necessary and truncating the query. These are primarily intended to test the parser to ensure correct operation under "real world" conditions.
- *Submitters*, which submit the query to DAS via different interfaces, or dump the result for debugging purposes.

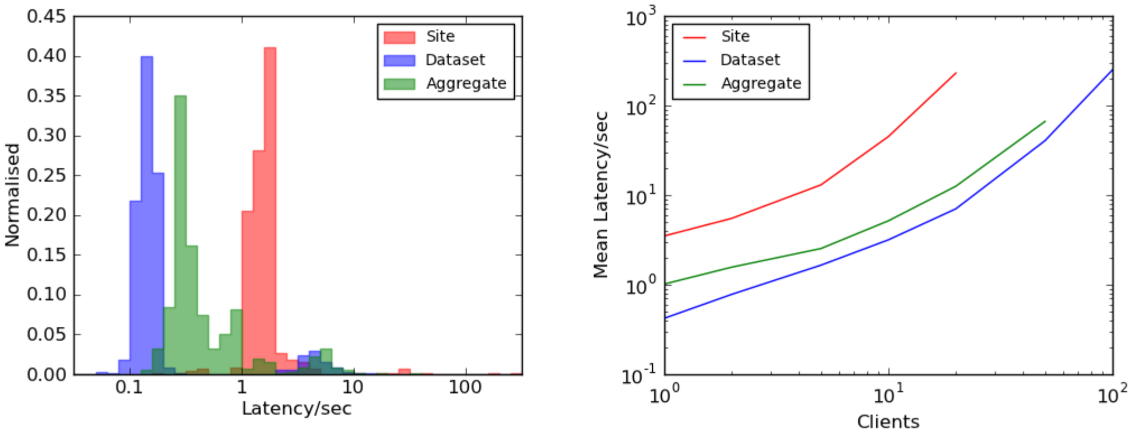
Queryspammer generates queries randomly from a corpus of data downloaded from the various data services also used by DAS. The distribution of queries can be uniform or weighted according to the expected distribution (based on mining the logfiles of existing services)[29]. Submission is multithreaded to simulate multiple concurrent users.

A limited demonstration of DAS performance can be made using *Queryspammer* to issue a concurrently issue a large number of weighted dataset queries, and examine the latency users experience as a function of number of concurrent users. These tests were performed using a quad-CPU node[30], with the clients requesting data via the web interface, and all of the web server, cache server and MongoDB running on the same system.

Three workflows were tested:

[29] For example, the model used for the distribution of datasets in `dataset=value` queries is $P(i) = 3 \cdot normal(\mu = i, \sigma^2 = 0.15 \cdot n) + \frac{1}{n}$ (normalisation factor omitted), where i is the dataset index and n is the number of datasets.

[30] Intel Core2 Q9400@2.66 GHz, with 4GiB RAM, running Centos 5 (64bit) and MongoDB 1.8. This is relatively weak compared to the dedicated server hardware that DAS would use in production.



(a) Serial

(b) Parallel

Figure 9.8: User latency (time for a request to the *web server* to return the complete JSON document) for three query scenarios. (a) shows a representative distribution of user latency, (showing the small number of queries which include the time necessary to fetch and process the data), when the query is being requested serially. (b) shows the latency when a large number of users attempt to fetch data in parallel (which has already been added to the DAS cache). Testing was stopped for each workflow when the average latency exceeded one minute.

- `dataset=X`, with a test corpus of 32 datasets with data volumes of $100KiB - 1MiB$, and requests a subset of "popular" datasets. Queries of this type are expected to make up the majority of DAS load, replacing those currently handled directly by DBS.

- `site=X`, with a test corpus of 32 sites with data volumes of $2MiB - 20MiB$, and requests distributed evenly. These are some of the largest requests likely to be fielded by DAS (since a `site` query pulls in information about all the blocks and files located at the site).

- `file dataset=X | sum(file.size), avg(file.size)`, with the same dataset corpus and distribution as above, but adding an aggregation function to test DAS performance at traversing the data.

The distribution of latencies under serial requests and the scaling of average latency with the number of concurrent clients is shown in Figure 9.8. This shows that providing the data is cached (reinforcing the need for good analytics Section 9.2.6), even large requests are served expeditiously. In the concurrent case latency increases moderately with small numbers of concurrent users (≈ 10 for this hardware), before reaching a point at which resource starvation occurs. In the *dataset* and *aggregate* cases, the limitation was CPU time (for MongoDB to retrieve output, and DAS to transform and possibly aggregate it), whereas in the *site* case memory was the limiting factor, as trying to process multiple large JSON documents exhausted physical memory and the system started swapping heavily, leading to near-lockup.

The number of concurrent users possible would be expected to scale approximately linearly with the number of CPUs available (assuming an equivalent amount of memory was available per CPU core). Conceivably a production-scale DAS system could be run on a single heavyweight node, but most likely several will be required to ensure acceptable performance.

Chapter 10

Conclusion

$Z \to \tau\tau$ Cross Section

The processes $Z \to \tau\tau \to e\tau_{jet}$, $\mu\tau_{jet}$ and $e\mu$ were studied, and their backgrounds characterised. For each final state, and their combination, the cross section $\sigma(pp \to Z \to \tau\tau)$ was extracted and compared with the results obtained from the $Z \to ee$ and $Z \to \mu\mu$ processes. The $Z \to \tau\tau$ cross sections were found to be:

- $e\tau_{jet}$: $(1100 \pm 180_{stat} \pm 110_{syst} \pm 45_{lumi} \pm 170_{\tau ID})$pb
- $\mu\tau_{jet}$: $(880 \pm 95_{stat} \pm 40_{syst} \pm 40_{lumi} \pm 180_{\tau ID})$pb
- $e\mu$: $(895 \pm 110_{stat} \pm 40_{syst} \pm 35_{lumi})$pb
- Combined: $(990 \pm 75_{stat} \pm 60_{syst} \pm 40_{lumi} \pm 110_{\tau ID})$pb

The limits were found to be compatible with the NNLO prediction and measurements made in the $Z \to ee$ and $Z \to \mu\mu$ channels.

MSSM $\Phi \to \tau\tau$ Limits

Studies of the light, neutral, MSSM Higgs boson were undertaken using the same three final states. Templates for the signal and major backgrounds for each channel were extracted, and the observed data and Monte-Carlo expectation fitted to those templates. Ten mass points for m_A between 90 and 300 GeV were considered, and for each the 95% CL limit on $\sigma(pp \to \Phi \to \tau\tau)$ extracted. The cross section limits were then used to constrain the $m_A \times \tan\beta$ plane, slightly improving on the Tevatron exclusion. The possibilities of using b-tagging for this search are discussed.

Computing Monitoring

The Overview monitoring framework was introduced, and the development of two monitoring components for this system discussed. The Prodmon workspace provides a rich plotting environment for monitoring the status of CMS Monte-Carlo production. The Filelight workspace provides a visualisation of the disposition of centrally-managed datasets at CMS computing sites. Finally, a standalone version of the Overview plotting library for general CMS computing use was briefly discussed.

Data Aggregation System

The desirability of being able to perform queries across multiple CMS data services, and of caching those requests to improve latency and reduce load was discussed. The implementation of the Data Aggregation Service was described, including the server components, analytics, query language and key introspection, and preliminary performance figures provided.

Appendix

A.1 CMSSW Config File Visualisations

Any of the tasks for which CMSSW is used (triggering, analysis, skimming and reconstruction) are controlled by configuration files. These include not only the definitions of the analysis and filtering steps involved in a job but also all the metadata required, for example detector geometry, bad channels, energy correction maps and magnetic field data. A configuration file typically imports a large number of others and if completely flattened may run to hundreds of kilobytes, which are highly opaque to end users and make finding errors or re-definitions of important values difficult.

Since CMSSW version 2, a custom text-based language for job configuration was replaced with executable scripts written in the Python[112] programming language using a set of classes representing CMSSW services, modules and individual configuration elements[1].

The configuration can now be inspected as live python objects in memory, and the python classes used can be dynamically modified ("monkey patched") with whatever extra monitoring we want to add.

This project was initially started as a standalone script, but was later merged into the CMSSW Config Browser[113] tool, to take advantage of shared configuration parsing code. The graph and HTML tools are available both as plugins for the Config Browser tool and as standalone scripts[2] in a standard CMSSW installation.

The simplest form of output is a script which dumps all the python configuration, resolves it into a hierarchy of paths and sequences and then renders it as an HTML file.

[1] Conceptually configuration could be entirely represented using python primitive types, but it proves necessary to use a hierarchy of objects that check for consistency and ensure that data can be translated to the appropriate C++ types (eg, typed lists to ensure that they can be translated to `vector<type>` rather than heterogenous containers).

[2] *edmConfigToGraph* and *edmConfigToHTML*

Javascript helpers are added to search within the file, and support the collapsing and expanding of each module and path. Compared to trying to understand a configuration split across multiple files and possibly including redefinition, this is significantly easier.

The graph view converts the tree of python objects into the DOT graph representation language[114] used by the *Graphviz* suite. While the HTML view shows the order in which modules are laid out (and are not guaranteed to run, because of the path optimisations performed by CMSSW), a more interesting view for the user is the connections of data between different modules; ie, which modules use the output of any given one. This is a much more useful representation of the "physics dependencies", for the purposes of understanding what a given configuration file actually does. Figure A.1 and A.2 show examples of the graph and HTML views applied to the configuration used for skimming in Section 6.1.3.

These two views are complementary, with the graphical representation quickly identifying modules of interest and the HTML view providing the detailed information about that module.

A.2 Pyanalysis

Pyanalysis is a python analysis package developed for the purpose of performing complex analysis tasks on ntuples.

It was designed to leverage the significant python standard library along with the ROOT framework to allow significant sharing of code between analysis while retaining the flexibility to easily and quickly make changes, which would have been more difficult or impossible under the constraints of using ROOT/Cint. The majority of analyses can also be represented as a sequence of highly generic operations (numeric operations on lorentz vectors or other object leaves, construction of composite particles, etc), which lend themselves very well to a duck-typed language. Examples are shown in Figure A.3.

It is also designed to address an entire analysis at once, including different Monte-Carlo or data samples and their scaling in a single pass, which is an abstraction notably not handled by CMSSW.

For a typical benchmark, python performs between two and three orders of magnitude worse than well-written C or C++. This situation is somewhat more complex, since the normally expensive deserialisation is performed by compiled ROOT code, but there

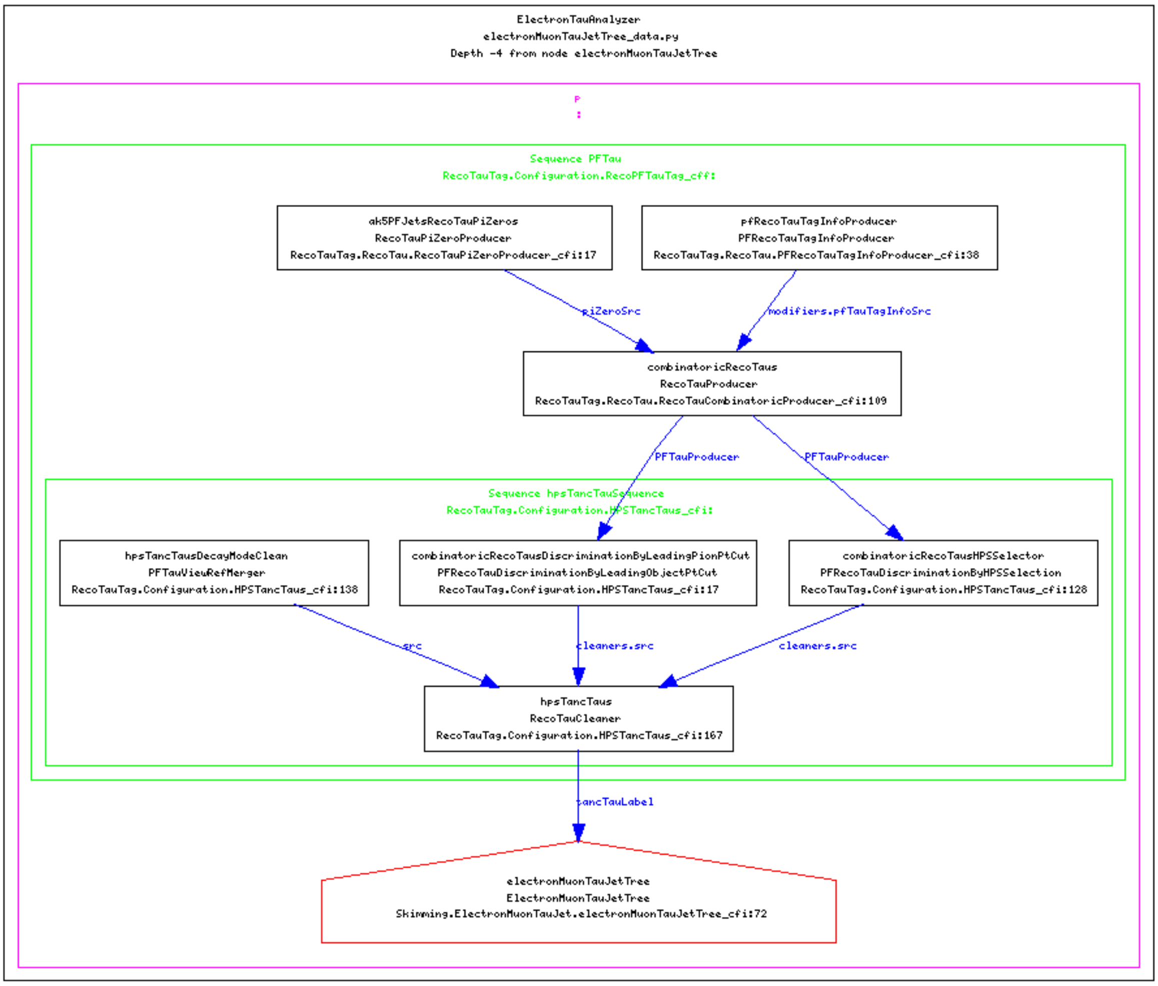
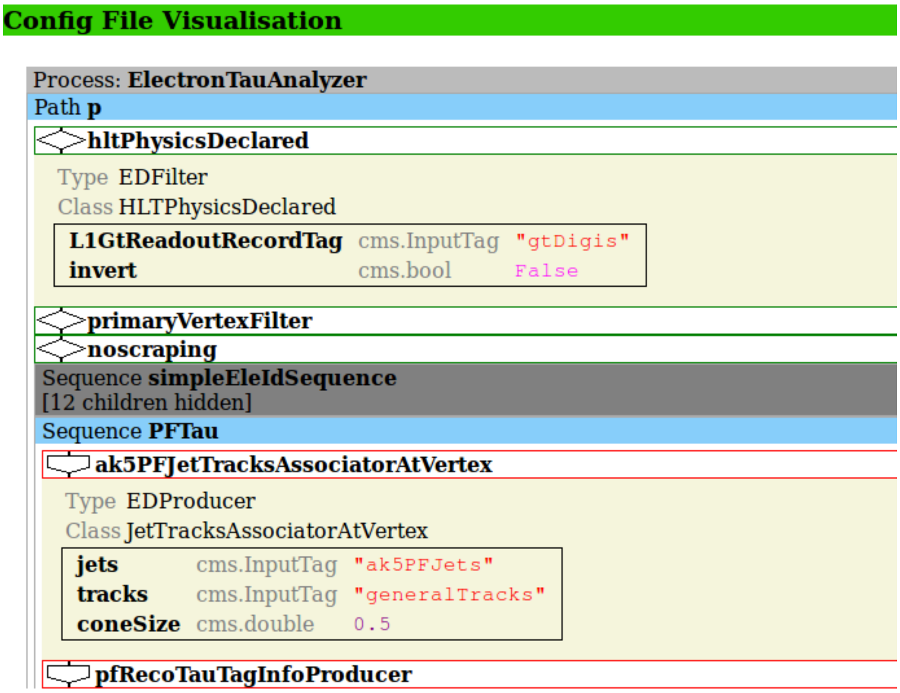

ElectronTauAnalyzer
electronMuonTauJetTree_data.py
Depth -4 from node electronMuonTauJetTree
p
:
Sequence PFTau
RecoTauTag.Configuration.RecoPFTauTag_cff:
ak5PFJetsRecoTauPiZeros
RecoTauPiZeroProducer
RecoTauTag.RecoTau.RecoTauPiZeroProducer_cfi:17
pfRecoTauTagInfoProducer
PFRecoTauTagInfoProducer
RecoTauTag.RecoTau.PFRecoTauTagInfoProducer_cfi:38
piZeroSrc
modifiers.pfTauTagInfoSrc
combinatoricRecoTaus
RecoTauProducer
RecoTauTag.RecoTau.RecoTauCombinatoricProducer_cfi:109
PFTauProducer
PFTauProducer
Sequence hpsTancTauSequence
RecoTauTag.Configuration.HPSTancTaus_cfi:
hpsTancTausDecayModeClean
PFTauViewRefMerger
RecoTauTag.Configuration.HPSTancTaus_cfi:138
combinatoricRecoTausDiscriminationByLeadingPionPtCut
PFRecoTauDiscriminationByLeadingObjectPtCut
RecoTauTag.Configuration.HPSTancTaus_cfi:17
combinatoricRecoTausHPSSelector
PFRecoTauDiscriminationByHPSSelection
RecoTauTag.Configuration.HPSTancTaus_cfi:128
src
cleaners.src
cleaners.src
hpsTancTaus
RecoTauCleaner
RecoTauTag.Configuration.HPSTancTaus_cfi:167
tancTauLabel
electronMuonTauJetTree
ElectronMuonTauJetTree
Skimming.ElectronMuonTauJet.electronMuonTauJetTree_cfi:72

Figure A.1: Graph visualisation of the immediate "physics dependencies" of the skimming module "electronMuonTauJetTree". This shows the tau reconstruction being recalculated as part of the skimming. The purple box denotes the overall *Path*, the green boxes the re-usable *Sequences* and the connected shapes *Producer*, *Filter* or *Analyser* modules. Within each module the label, C++ class name and location (file and line) at which it was defined is given. The names attached to the interconnecting lines show the label under which the latter module uses the output of the former.



Config File Visualisation

Process: **ElectronTauAnalyzer**

Path **p**

hltPhysicsDeclared

Type EDFilter

Class HLTPhysicsDeclared

L1GtReadoutRecordTag	cms.InputTag	"gtDigis"
invert	cms.bool	False

primaryVertexFilter

noscraping

Sequence **simpleEleIdSequence**
[12 children hidden]

Sequence **PFTau**

ak5PFJetTracksAssociatorAtVertex

Type EDProducer

Class JetTracksAssociatorAtVertex

jets	cms.InputTag	"ak5PFJets"
tracks	cms.InputTag	"generalTracks"
coneSize	cms.double	0.5

pfRecoTauTagInfoProducer

Figure A.2: HTML visualisation of the first few items of the skimming configuration, showing the rendering of module parameters and the grouping and collapsing of sequences.

```
ROOTLoadStep(prefixes=["electron", "muon", "tau"],
             clones={"electron": ["loose_electron"]})
ObjectCountCut("muon", ">=", 1)
ObjectLVCut("electron", "Eta", "<", 2.5, abs=True)
ObjectLambdaCut("tau", lambda tau: tau.hcal/tau.lv.Pt() > 0.1,
                "HCAL/Pt>0.1")
BuildComposite("pair", "electron", "muon", dr=0.5)
Hist2D("TauPtEta",
       lambda event: [(tau.lv.Pt(), tau.lv.Eta()) for tau in event.taus],
       100, 0, 100, 60, -3, 3)
```

Figure A.3: Sample pyanalysis operations.

are significant overheads involved in interfacing between the python and CINT object and memory models. Benchmarking the relative performance of calling the ROOT TLorentzVector class via PyROOT and a pure-python implementation of the same methods shows approximately equal performance[3], and for operations requiring repeatedly creating new objects (such as reducing a large list) the pure-python implementation actually provides slightly better performance.

However, python does impose a significant performance penalty in practice. Although it was possible to more than double performance by redesigning the python-side code which expanded entries in the ROOT tree into the python representation, it proved impossible to improve performance much beyond 1 kHz (somewhat dependent on the event size and complexity).

To improve performance further, it was necessary to perform some operations in native code. The cut sequence is analysed in order and a subset of simple cuts are converted to C++. The emitted code is assembled into a class, which acts as a generator yielding the indices of events passing a basic set of cuts for full analysis. The compiled code is accessed through ROOT's *reflex* reflection mechanism rather than by building a directly loadable python extension module[4].

The generated code is produced by hand-written templates in the appropriate cut classes, rather than any more complex scheme of inspecting and recasting the abstract syntax tree. The C++ representations of cuts are not required to have exactly the same semantics as the python variants, but rather be guaranteed to be equal or a superset of the operation. For example, a pair of sequential isolation cuts on different subdetectors would, in the python implementation, require the same lepton to pass both cuts, whereas the C++ code would pass the event if at least one lepton passes each cut. For events selected by the compiled index generator, all of the python cuts are run, including those cuts for which the superset operation has already been performed. Using the index generator improved performance to approximately 10 kHz, sufficient to process the entire data and Monte-Carlo set for any one of the $Z \to \tau\tau$ channels in approximately an hour.

Deserialised events loaded from a ROOT tree[5] and assembled into small batches, after which each batch is completely analysed. This is implemented using co-routines for

[3] Python 2.6.4 vs ROOT/PyROOT 5.26.0

[4] As an aside, the *pypy* project is investigating using reflex as a generic mechanism for python to access compiled code

[5] Or other source; it would be feasible, albeit expensive, to load directly from "full" CMSSW AOD or RECO files, using the Reflex dictionaries for each object class. It would also be possible to load directly from an event generator; eg using the ROOT interface to PYTHIA6/8.

each analysis step, so all analysis levels remain in scope throughout the analysis, and the memory footprint is minimised[6]. Optimum performance was found to be with bunches of $100 - 1000$ events (balancing co-routine yield overhead with memory allocation overhead).

Pyanalysis analyses are controlled by instantiating a process object (interactively or in a script) and setting the necessary process properties (maximum events, output files, etc), list of samples to analyse, and the steps to perform. The resulting configuration files are not entirely dissimilar to those of CMSSW, but the fact that the analysis is invoked in the script (whereas in CMSSW, the interpreter is embedded in the executable and sets itself up based on the final python object graph rather than any invoked function) allows multiple tasks with related parameters or different mass points to be launched and configured in a single script, and configuration is not restricted to constant parameters; in many case closures are expected.

Once running, an `ncurses`-based GUI is optionally displayed to the user, showing progress and statistics such as estimated completion time (a headless mode for batch operation is also available), and output is stored in both a ROOT file (for any plotting or graphing tasks) and a comma-separated text file (for run statistics).

Some work was done on multiprocessing (which could have been easily extended into processing across multiple physical hosts while operating as one logical process, rather than the multiple independent processes CMSSW uses for parallelism). This is necessary to avoid a manual merge step, which would remove the advantage of running on all samples as a single process (plus it allows some operations to be done in a single pass, such as template extraction and fitting). However, resolving the differences between the ROOT and python locking models (particularly the role of the python Global Interpreter Lock), and finding ways to make the resulting copy-on-write semantics invisible to analysis code proved impossible within a reasonable period of time.

A.3 CERN

The European Organisation for Nuclear Research[7] is an international particle physics research facility, located on the Swiss-French frontier a short distance from Geneva.

[6] Typically $< 100 MiB/core$.

[7] The acronym CERN derives from the original French name, ***C**onseil **E**uropéenne pour la **R**eserche **N**ucléaire*

CERN was established by 12 western european nations in 1952, with the convention establishing the Swiss/French site agreed two years later[8].

The first major accelerator, the 26 GeV Proton Synchrotron (PS) was constructed in 1959. This serves as a source for a number of fixed target experiments, including the Gargamelle bubble chamber in which the weak neutral current was discovered in 1974[22]. This was followed by the Intersecting Storage Rings, the first dedicated proton-proton collider in 1971.

The next major accelerator, in 1982, was the 450 GeV Super Proton Synchrotron (SPS) was constructed in a 6 km circular tunnel, the space to construct it above ground being lacking. Originally built to provide beams for fixed targets (for which it is still used today), following advances in stochastic cooling making a stable antiproton beam viable it was rebuilt as a proton-antiproton collider. This allowed the UA1 and UA2 experiments to observe both the W and Z bosons in 1983[6].

Shortly after the successes of the SPS, construction started on LEP[9], a 27 km electron-positron collider. This was initially operated in 1989 at 45 GeV per beam to maximise production of Z bosons (also 40 GeV operation to maximise W production). The beam energies were later increased to 105 GeV per beam for Higgs search, but aside from a handful of possible candidate events[115] nothing was seen by the time it was shut down in November 2000.

A.4 LHC Incident

First beams were circulated in the LHC (at 450 GeV injection energy) on 10th September 2008, and a small number of collisions conducted. On the 19th, while the magnets were being powered to 9.3 kA (for 5.5 TeV per beam), an electrical fault in sector 3-4 caused serious electrical and mechanical damage. The sector had previously been successfully powered to 7 kA. The incident is fully described by [116].

The fault occurred in the superconducting bus-bar that provides power for the dipole magnets (see Figure A.4). A badly connected splice between two sections had an abnormally high resistance (approximately 200 nΩ) which at this current caused

[8] The parties to the original convention were Belgium, Denmark, France, West Germany, Greece, Italy, the Netherlands, Norway, Sweden, Switzerland, the United Kingdom and Yugoslavia, and ratification was completed on 29th September 1954.

[9] Large Electron-Positron

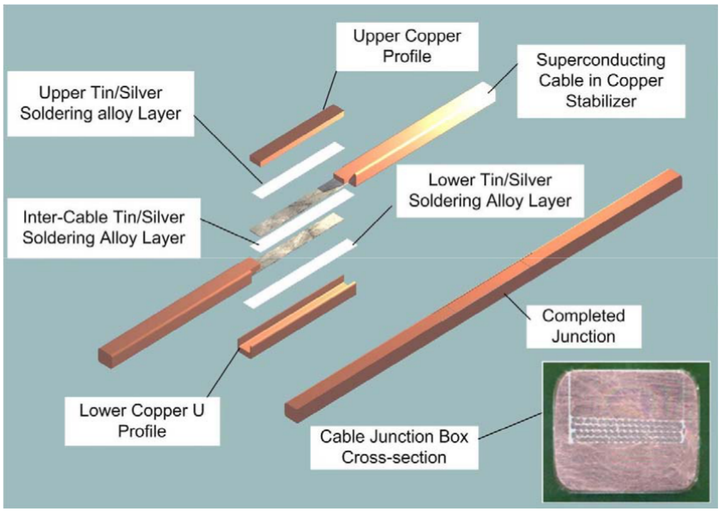

Upper Copper Profile
Superconducting Cable in Copper Stabilizer
Upper Tin/Silver Soldering alloy Layer
Lower Tin/Silver Soldering Alloy Layer
Inter-Cable Tin/Silver Soldering Alloy Layer
Completed Junction
Lower Copper U Profile
Cable Junction Box Cross-section

Figure A.4: Diagram of the superconducting busbar joint which failed in Sector 3-4. [116].

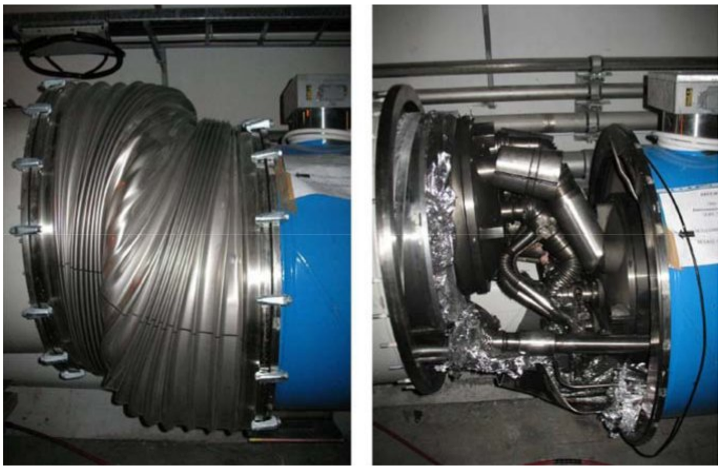


Figure A.5: Inter-dipole connection mechanically deformed by helium overpressure. [116].

significant heating. While highly sensitive voltage sensors were placed to protect the individual dipoles from a superconducting quench, the interconnect had a much higher threshold, and was expected to be electrically and thermally stabilised by being embedded in a thick copper bar. In this case an arc between the two superconducting cables developed, dissipating around 4 MW and completely vapourising the original interconnect and copper backup, as well as puncturing the helium cryostat (see Figure A.5).

The loss of cryostat pressure and electrical heating resulted in a rapid outflow of helium, with approximately 20 kg/s reverting to gas. The resulting forces caused mechanical deformation in several adjacent magnets, resulting in additional helium loss. Damage to both the electrical and cryogenic circuits resulted in the loss of approximately 6000 kg of helium and the remaining magnets in the sector discharging at quadruple the design fast discharge rate.

Around 750 m of beamline was damaged, with 37 magnet units needing to be removed and rebuilt on the surface, or replaced with spares. Soot from vapourised electrical components contaminated over a kilometre of the beamline, which had to be manually cleaned. Over a year of work after the accident was required to analyse it, replace the damaged accelerator components and install new quench sensors and helium release valves to the remaining sectors. The LHC remains limited to lower magnet currents (6 kA, for 3.5 TeV) until a further shutdown allows detailed inspection of all the splices (amongst other work), currently anticipated after 2012.

Bibliography

[1] A. Asner et al. “ECFA-CERN Workshop on Large Hadron Collider in the LEP Tunnel”. *CERN* (1984). CERN 84-10. cds:1296749.

[2] F. Halzen and A. D. Martin. *Quarks and Leptons*. Wiley, 1985.

[3] D. Griffiths. *Introduction to Elementary Particles*. Wiley, 1987.

[4] G. Altarelli and M. W. Grunewald. “Precision electroweak tests of the standard model”. *Phys. Rept.* 403-404 (2004), pp. 189–201. arXiv:hep-ph/0404165.

[5] G. Arnison et al. “Experimental observation of events with large missing transverse energy accompanied by a jet or a photon in $p\bar{p}$ collisions at $\sqrt{s} = 540$ GeV”. *Physics Letters B* 139.1-2 (1984), pp. 115 –125. doi:10.1016/0370-2693(84)90046-7.

[6] G. Arnison et al. “Experimental observation of lepton pairs of invariant mass around 95 GeV at the CERN SPS collider”. *Physics Letters B* 126.5 (1983), pp. 398 –410.

[7] P. Jenni. “Experimental review of W and Z production and decay at the CERN $p\bar{p}$ collider.” *Nuclear Physics B* 3 (1987), pp. 341–366. cds:183638.

[8] S. Glashow. “Partial-symmetries of weak interactions”. *Nuclear Physics* 22.4 (1961), pp. 579–588. doi:10.1016/0029-5582(61)90469-2.

[9] S. Weinberg. “A Model of Leptons”. *Phys. Rev. Lett.* 19.21 (1967), pp. 1264–1266. doi:10.1103/PhysRevLett.19.1264.

[10] A. Salam and J. C. Ward. “Electromagnetic and weak interactions”. *Physics Letters* 13.2 (1964), pp. 168 –171. doi:10.1016/0031-9163(64)90711-5.

[11] P. W. Higgs. “Broken Symmetries and the Masses of Gauge Bosons”. *Phys. Rev. Lett.* 13.16 (1964), pp. 508–509. doi:10.1103/PhysRevLett.13.508.

[12] G. S. Guralnik, C. R. Hagen, and T. W. B. Kibble. “Global Conservation Laws and Massless Particles”. *Phys. Rev. Lett.* 13.20 (1964), pp. 585–587. doi:10.1103/PhysRevLett.13.585.

[13] J. Goldstone. “Field theories with Superconductor solutions”. *Il Nuovo Cimento (1955-1965)* 19 (1 1961), pp. 154–164. doi:10.1007/BF02812722.

[14] LEP and Tevatron Electroweak Working Groups. “Precision Electroweak Measurements and Constraints on the Standard Model” (2008). arXiv:0811.4682.

[15] M. Goebel and Gfitter Group. “Status of the global fit to electroweak precisions data”. *PoS* ICHEP2010 (2010), p. 570. arXiv:1012.1331.

[16] K. Riesselmann. “Limitations of a standard model Higgs boson” (1997). arXiv:hep-ph/9711456.

[17] Y. Fukuda et al. “Measurements of the solar neutrino flux from Super-Kaminokande's first 300 days”. *Phys. Rev. Lett.* 81 (1998), pp. 1158–1162. arXiv:hep-ex/9805021.

[18] I. J. R. Aitchison. “Supersymmetry and the MSSM: An Elementary introduction” (2005). arXiv:hep-ph/0505105.

[19] W. de Boer and C. Sander. “Global electroweak fits and gauge coupling unification”. *Phys. Lett.* B585 (2004), pp. 276–286. arXiv:hep-ph/0307049.

[20] V. A. Bednyakov and H. V. Klapdor-Kleingrothaus. “Update of the direct detection of dark matter and the role of the nuclear spin”. *Phys. Rev.* D63 (2001), p. 095005. arXiv:hep-ph/0011233.

[21] S. P. Martin. “A Supersymmetry Primer” (1997). arXiv:hep-ph/9709356.

[22] F. J. Hasert et al. “Observation of neutrino-like interactions without muon or electron in the Gargamelle neutrino experiment”. *Nuclear Physics B* 73.1 (1974), pp. 1 –22.

[23] S. D. Drell and T.-M. Yan. “Massive Lepton-Pair Production in Hadron-Hadron Collisions at High Energies”. *Phys. Rev. Lett.* 25.5 (1970), pp. 316–320. doi:10.1103/PhysRevLett.25.316.

[24] K. H. et al. (Particle Data Group). “Review of Particle Physics”. *J. Phys. G* 37 (2010), p. 075021. http://pdg.lbl.gov/.

[25] M. L. Perl et al. “Evidence for Anomalous Lepton Production in e^+e^- Annhiliation”. *Phys. Rev. Lett.* 35.22 (1975), pp. 1489–1492. doi:10.1103/PhysRevLett.35.1489.

[26] *Views of the tunnel LHC sector 1-2*. 2008. cds:1139628.

[27] Tevatron Collaboration. “Design Report Tevatron 1 Project”. *Fermi National Accelerator Laboratory* (1982). FERMILAB-DESIGN-1982-01.

[28] HERA Collaboration. “HERA, A Proposal for a Large Electron-Proton Colliding Beam Facility at DESY”. *DESY* (1981). DESY HERA 81/10.

[29] *LHC Homepage*. `http://lhc.web.cern.ch/lhc/`.

[30] *The CERN accelerator complex*. 2008. `cds:1260465`.

[31] *CMS Luminosity Overview*. 2011. `http://cms-service-lumi.web.cern.ch/cms-service-lumi/overview.php`.

[32] CMS Collaboration. “CMS Technical Proposal”. *CERN* (1994). CERN/LHCC/94-38.

[33] CMS Collaboration. “CMS Letter of Intent”. *CERN* (1994). CERN/LHCC/92-3.

[34] CMS Collaboration. “CMS Physics Technical Design Report, Volume 1”. *CERN* (2006). CERN/LHCC/2006-001. `cds:922757`.

[35] CMS Collaboration. “CMS Tracker Technical Design Report”. *CERN* (1998). CERN/LHCC/98-6.

[36] CMS Collaboration. “CMS Electomagnetic Calorimeter Technical Design Report”. *CERN* (1997). CERN/LHCC/97-33.

[37] R. Brown. “The CMS lead tungstate electromagnetic calorimeter”. *Journal of Physics: Conference Series* 110.9 (2008), p. 092005.

[38] CMS Collaboration. “CMS Hadronic Calorimeter Technical Design Report”. *CERN* (1997). CERN/LHCC/97-31.

[39] CMS Collaboration. “CMS Muon Project Technical Design Report”. *CERN* (1997). CERN/LHCC/97-32.

[40] CMS Collaboration. “Performance of CMS muon reconstruction in cosmic-ray events”. *JINST* 5.03 (2010). CMS-CFT-09-014, p. 22. `arXiv:0911.4994v2`.

[41] B. Hegner. “A tour through the CMS Physics Analysis Model”. *CERN* CMS-CR-2011-022 (2011). `cds:1326922`.

[42] R. Stallman. “new Unix implementation”. *net.unix-wizards* (1983).

[43] L. Torvalds. “What would you like to see most in minix?” *comp.os.minix* (1991).

[44] R. Brun and F. Rademakers. *ROOT, a data analysis framework*. 1995. `http://root.cern.ch/`.

[45] LCG TDR Editorial Board. “LHC Computing Report Technical Design Report”. *CERN* (2005). CERN/LHCC/2005-024. `cds:840543`.

[46] *GridMap Visualizing the State of the Grid*. http://gridmap.cern.ch/.

[47] CMS Collaboration. “CMS Computing Project Technical Design Report”. *CERN* (2005). CERN/LHCC/2005-023. cds:838359.

[48] CMS Collaboration. “Performance of muon identification in pp collisions at $\sqrt{s} = 7$ TeV”. *CERN* (2010). CMS PAS MUO-10-002. cds:1279140.

[49] K. Pearson. “On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling”. *Philosophical Magazine* 50.302 (1900), pp. 157–175.

[50] R. Frühwirth. “Application of Kalman filtering to track and vertex fitting”. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 262.2-3 (1987), pp. 444 –450. ISSN: 0168-9002. doi:10.1016/0168-9002(87)90887-4.

[51] W. Adam et al. “Track reconstruction in the CMS tracker”. *CERN* (2006). CMS AN2006/041. cds:934067.

[52] CMS Collaboration. “Electron reconstruction and identification at $\sqrt{s} = 7$ TeV”. *CERN* (2010). CMS PAS EGM-10-004. cds:1299116.

[53] W. Adam et al. “Reconstruction of Electrons with the Gaussian-Sum Filter in the CMS Tracker at the LHC”. *CERN* CMS-NOTE-2005-001 (2005). cds:815410.

[54] CMS Collaboration. “Particle-Flow Event Reconstruction in CMS and Performance for Jets, Taus and E_T^{miss}”. *CERN* (2009). CMS PAS PFT-09-001. cds:1194487.

[55] M. Cacciari, G. P. Salam, and G. Soyez. “The anti-kt jet clustering algorithm”. *Journal of High Energy Physics* 2008.04 (2008), p. 063. arXiv:0802.1189v2.

[56] CMS Collaboration. “CMS strategies for tau reconstruction and identification using particle-flow techniques”. *CERN* (2008). CMS PAS PFT-08-001. cds:1198228.

[57] L. Bianchini. “Measurement of the $e \to \tau_{had}$ fake rate with 33 pb^{-1} of pp collision data at $\sqrt{s} = 7$ TeV”. *CERN* (2011). CMS AN-2011/420.

[58] M. Bachtis et al. “Performance of tau reconstruction algorithms in 2010 data collected with CMS”. *CERN* (2011). CMS NOTE TAU-11-001. cds:1337004.

[59] CMS Collaboration. “Study of tau reconstruction algorithms using pp collisions data collected at $\sqrt{s} = 7$ TeV”. *CERN* (2010). CMS PAS PFT-10-004. cds:1279358.

[60] M. Bachtis, S.Dasu, and A. Savin. “Prospects for measurement of $\sigma(pp \to Z) \cdot Br(Z \to \tau^+\tau^-)$ with CMS in pp Collisions $\sqrt{s} = 7$ TeV”. *CERN* (2010). CMS AN-2010/082.

[61] J. Conway et al. “The Tau Neural Classifier algorithm: tau identification and decay mode reconstruction using neural networks”. *CERN* (2010). CMS AN-2010/099.

[62] CMS Collaboration. “Missing Transverse Energy Performance in Minimum-Bias and Jet Events from Proton-Proton Collisions at $\sqrt{s} = 7$ TeV”. *CERN* (2010). CMS PAS JME-10-004. cds:1279142.

[63] Z. Zhang. “Tau lepton physics at LEP”. *PoS* HEP2005 (2006), p. 290. arXiv:hep-ex/0602044.

[64] CMS Collaboration. “Measurement of the W and Z inclusive production cross sections at $\sqrt{s} = 7$ TeV with the CMS experiment at the LHC”. *CERN* (2011). CMS PAS EWK-10-005. cds:1337017.

[65] T. Sjöstrand, S. Mrenna, and P. Skands. “PYTHIA 6.4 Physics and Manual”. *JHEP* 05 (2006), p. 026. arXiv:hep-ph/0603175.

[66] R. Field. “Early LHC Underlying Event Data - Findings and Surprises” (2010). arXiv:1010.3558.

[67] Z. Was. “Tauola the library for tau lepton decay”. *Nucl. Phys. Proc. Suppl.* 98 (2001), pp. 96–102. arXiv:hep-ph/0011305.

[68] CMS Collaboration. “Inelastic pp cross section at 7 TeV”. *CERN* (2011). CMS PAS FWD-11-001. cds:1373466.

[69] J. Allison et al. “Geant4 Developments and Applications”. *IEEE Transactions on Nuclear Sciences* 53 (2006), pp. 270–278.

[70] M. Bachtis et al. “Search for neutral Higgs boson decaying into τ pairs using 35 pb^{-1} at $\sqrt{s} = 7$ TeV.” *CERN* (2010). CMS AN-10-430.

[71] CMS Collaboration. “Absolute Luminosity Normalisation”. *CERN* (2011). CMS DPS-2011/002. cds:1335668.

[72] CMS Collaboration. “Measurement of the Inclusive Z Cross Section via Decays to Tau Pairs in pp Collisions at $\sqrt{s} = 7$ TeV” (2011). arXiv:1104.1617.

[73] K. Melnikov and F. Petriello. “Electroweak gauge boson production at hadron colliders through $O(\alpha_s^2)$”. *Phys. Rev.* D74 (2006), p. 114017. arXiv:hep-ph/0609070.

[74] LHC Higgs Cross Section Working Group. "Handbook of the LHC Higgs Cross Sections: 1. Inclusive Observables". *CERN-2011-002* (2011). cds:1318996.

[75] S. Heinemeyer, W. Hollik, and G. Weiglein. "FeynHiggs: a program for the calculation of the masses of the neutral CP-even Higgs bosons in the MSSM". *Comput. Phys. Commun.* 124 (2000), pp. 76–89. arXiv:hep-ph/9812320.

[76] R. V. Harlander and W. B. Kilgore. "Higgs boson production in bottom quark fusion at next-to- next-to-leading order". *Phys. Rev.* D68 (2003), p. 013001. arXiv:hep-ph/0304035.

[77] R. V. Harlander and W. B. Kilgore. "Next-to-next-to-leading order Higgs production at hadron colliders". *Phys. Rev. Lett.* 88 (2002), p. 201801. arXiv:hep-ph/0201206.

[78] M. Spira. "HIGLU: A Program for the Calculation of the Total Higgs Production Cross Section at Hadron Colliders via Gluon Fusion including QCD Corrections" (1995). arXiv:hep-ph/9510347.

[79] M. Carena et al. "MSSM Higgs boson searches at the Tevatron and the LHC: Impact of different benchmark scenarios". *The European Physical Journal C - Particles and Fields* 45 (3 2006), pp. 797–814. ISSN: 1434-6044. arXiv:hep-ph/0511023.

[80] A. L. Read. "Linear interpolation of histograms". *Nuclear Instruments and Methods in Physics Research A* 425 (1999), pp. 357–360.

[81] L. Moneta et al. "The RooStats Project". *PoS* ACAT2010 (2010), p. 057. cds:1289965.

[82] F. James and M. Roos. "Minuit: A System for Function Minimization and Analysis of the Parameter Errors and Correlations". *Comput. Phys. Commun.* 10 (1975), pp. 343–367.

[83] LEP Working Group for Higgs Boson Searches. "Search for neutral MSSM Higgs bosons at LEP". *The European Physical Journal C - Particles and Fields* 47 (3 2006), pp. 547–587. arXiv:hep-ex/0602042.

[84] T. W. Group. "Combined CDF and D upper limits on MSSM Higgs boson production in tau-tau final states with up to 2.2 fb^{-1} of data". *Fermilab* (2009). FERMILAB-PUB-09-394-E.

[85] CMS Collaboration. "Commissioning of b-jet identification with pp collisions at $\sqrt{s} = 7$ TeV". *CERN* (2010). CMS PAS BTV-10-001. cds:1279144.

[86] T. Berners-Lee and R. Cailliau. “WorldWideWeb: Proposal for a HyperText Project”. *CERN* (1990). http://www.w3.org/proposal.html.

[87] R. Delon et al. *CherryPy, a pythonic, object-orientated HTTP framework*. http://www.cherrypy.org/.

[88] ECMA International. *Standard ECMA-262, ECMAScript Language Specification, 5th edition*. 2009. http://www.ecma-international.org/publications/standards/Ecma-262.htm.

[89] D. Crockford. *The application/json Media Type for JavaScript Object Notation (JSON)*. 2006. http://www.ietf.org/rfc/rfc4627.txt.

[90] J. Ziv and A. Lempel. “A Universal Algorithm for Sequential Data Compression”. *IEEE Transactions on Information Theory* 23.3 (1977), pp. 337–343.

[91] J. Rehn. “PhEDEx high-throughput data transfer management system”. *Proceedings of CHEP06, Mumbai, India*. 2006.

[92] L. Tuura et al. “Scaling CMS data transfer system for LHC start-up”. *Journal of Physics: Conference Series* 119.7 (2008), p. 072030. cds:1176898.

[93] D. Evans et al. “The CMS Monte Carlo Production System: Development and Design”. *Nuclear Physics B - Proceedings Supplements* 177-178 (2008), pp. 285 –286. cds:1073698.

[94] J. Andreeva et al. “Experiment Dashboard for Monitoring Computing Activies of the LHC Virtual Organizations”. *Journal of Grid Computing* 8 (2 2010), pp. 323–339. doi:10.1007/s10723-010-9148-x.

[95] S. Kleene. “Representation of Events in Nerve Nets and Finite Automata”. *Automata Studies*. Ed. by C. Shannon and J. Mccarthy. Princeton, N.J.: Princeton University Press, pp. 3–42.

[96] T. Boutell. *PNG (Portable Network Graphics) Specification*. 1997. http://www.ietf.org/rfc/rfc2083.txt.

[97] J. D. Hunter. “Matplotlib: A 2D graphics environment”. *Computing In Science & Engineering* 9.3 (2007), pp. 90–95.

[98] J. Seidman. *A Proposed Extension to HTML : Client-Side Image Maps*. 1996. http://www.ietf.org/rfc/rfc1980.txt.

[99] A. Afaq et al. “The CMS dataset bookkeeping service”. *Journal of Physics: Conference Series* 119.7 (2008), p. 072001. doi:10.1088/1742-6596/119/072001.

[100] *Disk Usage Analyzer (aka Baobab)*. http://www.marzocca.net/linux/baobab/.

[101] *Filelight*. http://www.methylblue.com/filelight/.

[102] 10gen, Inc. *MongoDB, a scalable, high-performance, open-source, document-orientated database*. http://www.mongodb.org/.

[103] G. Ball et al. "Data Aggregation System - a system for information retrieval on demand over relational and non-relational distributed data sources". *CERN* (2010). CMS-CR-2010-230. cds:1319363.

[104] *The Apache CouchDB project*. http://couchdb.apache.org/.

[105] V. Kuznetsov et al. "The CMS DBS query language". *Journal of Physics: Conference Series* 219.4 (2010), p. 042043. cds:1196159.

[106] D. Beazley. *PLY (Python Lex-Yacc)*. http://www.dabeaz.com/ply/.

[107] M. E. Lesk and E. Schmidt. "Lex-a lexical analyzer generator" (1990).

[108] S. C. Johnson. "Yacc: Yet Another Compiler-Compiler" (1975).

[109] R. Rivest. *The MD5 Message-Digest Algorithm*. 1992. http://www.ietf.org/rfc/rfc1321.txt.

[110] Yahoo!, Inc. *YUI Library*. http://developer.yahoo.com/yui/.

[111] T. Rudd, M. Orr, and I. Bicking. "Cheetah: The Python-Powered Template Engine". *Tenth Internation Python Conference*. 2002.

[112] G. van Rossum and Python Software Foundation. *Python Programming Language*. 1991. http://www.python.org/.

[113] M. Erdmann et al. "Visualization of the CMS python configuration system". *Journal of Physics: Conference Series* 219.4 (2010), p. 042008. cds:1196158.

[114] J. Ellson et al. "Graphviz and dynagraph - static and dynamic graph drawing tools". *Graph Drawing Software*. Spring-Verlag, 2003, pp. 127–148.

[115] R. Barate et al. "Search for the standard model Higgs boson at LEP". *Phys. Lett.* B565 (2003), pp. 61–75. arXiv:hep-ex/0306033.

[116] M. Bajko et al. "Report of the task force on the incident of 19 September 2008 at the LHC". *CERN* (2009). LHC Project Report 1168. cds:1168025.

[117] D. J. Mackay. "Information Theory, Inference and Learning Algorithms". Cambridge University Press, 2003.

Colophon

This thesis was typeset in LaTeX using the **hepthesis**[10] class, maintained with **git**[11], edited with **vim**[12] and **TeXmaker**[13] and could not have been produced without copious quantities of *Coffea arabica* seed extract.

```
git-commit:   4bee059796141c4f3244c1c552fe374550e5bae6
git-date:     Wed Nov 9 21:04:08 2011 +0000
latex-ver:    pdfTeX 3.1415926-1.40.10-2.2 (TeX Live 2009/Debian)
bibtex-ver:   BibTeX 0.99c (TeX Live 2009/Debian)
metafont-ver: Metafont 2.718281 (TeX Live 2009/Debian)
compiled-by:  chronitis@chronosphere
compiled-at:  23/11/2011-15:05
word-count:   39250
```

[10] http://www.insectnation.org/projects/hepthesis

[11] http://git-scm.com

[12] http://www.vim.org

[13] http://www.xm1math.net/texmaker

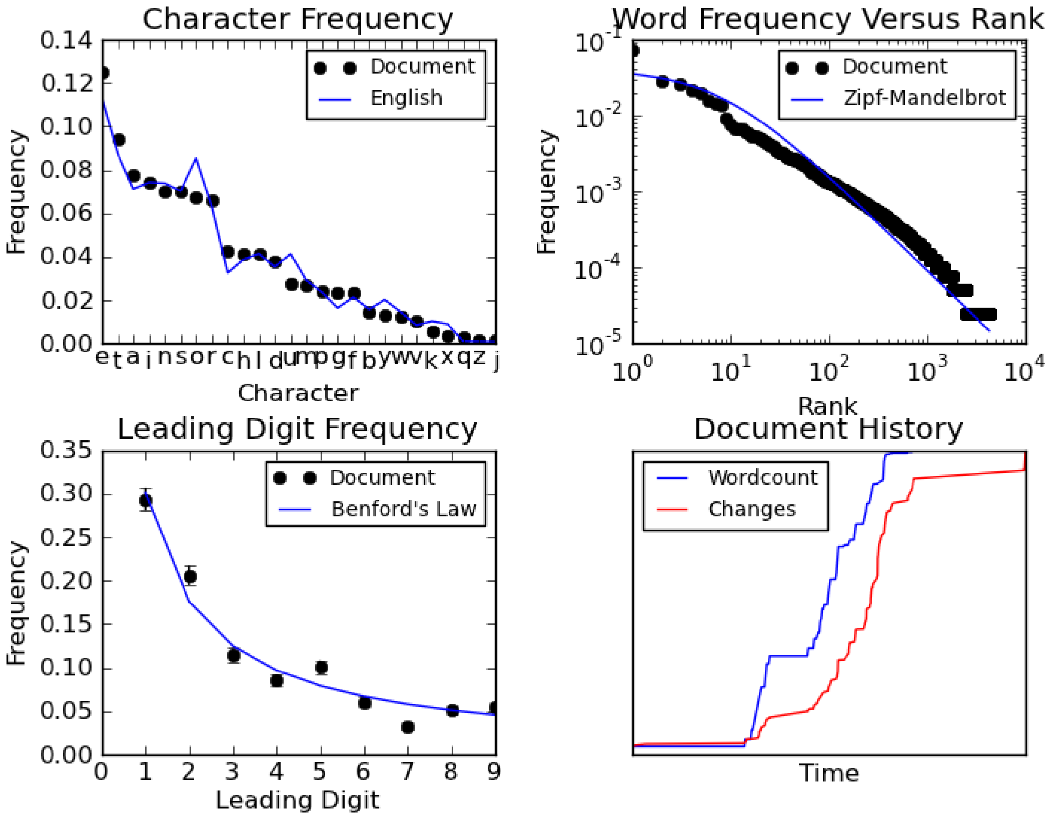


Figure 6: Statistics for this document. Nominal distributions from [117].