Master's thesis
Theoretical and Computational Methods

Studies of jet energy corrections at the CMS experiment and their automation

Adelina Lintuluoto
February 2021

Supervisor: Asst. prof. Mikko Voutilainen
Advisor: Dr. Clemens Lange

Examiners: Asst. prof. Mikko Voutilainen
Dr. Kati Lassila-Perini

UNIVERSITY OF HELSINKI
DEPARTMENT OF PHYSICS

PB 64 (Gustaf Hällströms gata 2a)
00014 Helsingfors universitet

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI
MATEMAATTIS-LUONNONTIETEELLINEN TIEDEKUNTA
MATEMATISK-NATURVETENSKAPLIGA FAKULTETEN
FACULTY OF SCIENCE

Tiedekunta – Fakultet – Faculty Faculty of Science	Koulutusohjelma – Utbildningsprogram – Degree programme Master's programme	
Opintosuunta – Studieinrikting – Study track Theoretical and computational methods		
Tekijä – Författare – Author Adelina Lintuluoto		
Työn nimi – Arbetets titel – Title Studies of jet energy corrections at the CMS experiment and their automation		
Työn laji – Arbetets art – Level Master's thesis	Aika – Datum – Month and year February 2021	Sivumäärä – Sidoantal – Number of pages 69

Tiivistelmä – Referat – Abstract

At the Compact Muon Solenoid (CMS) experiment at CERN (European Organization for Nuclear Research), the building blocks of the Universe are investigated by analysing the observed final-state particles resulting from high-energy proton-proton collisions. However, direct detection of final-state quarks and gluons is not possible due to a phenomenon known as colour confinement. Instead, event properties with a close correspondence with their distributions are studied. These event properties are known as jets. Jets are central to particle physics analysis and our understanding of them, and hence of our Universe, is dependent upon our ability to accurately measure their energy. Unfortunately, current detector technology is imprecise, necessitating downstream correction of measurement discrepancies. To achieve this, the CMS experiment employs a sequential multi-step jet calibration process. The process is performed several times per year, and more often during periods of data collection.

Automating the jet calibration would increase the efficiency of the CMS experiment. By automating the code execution, the workflow could be performed independently of the analyst. This in turn, would speed up the analysis and reduce analyst workload. In addition, automation facilitates higher levels of reproducibility.

In this thesis, a novel method for automating the derivation of jet energy corrections from simulation is presented. To achieve automation, the methodology utilises declarative programming. The analyst is simply required to express *what* should be executed, and no longer needs to determine *how* to execute it. To successfully automate the computation of jet energy corrections, it is necessary to capture detailed information concerning both the computational steps and the computational environment. The former is achieved with a computational workflow, and the latter using container technology. This allows a portable and scalable workflow to be achieved, which is easy to maintain and compare to previous runs.

The results of this thesis strongly suggest that capturing complex experimental particle physics analyses with declarative workflow languages is both achievable and advantageous. The productivity of the analyst was improved, and reproducibility facilitated. However, the method is not without its challenges. Declarative programming requires the analyst to think differently about the problem at hand. As a result there are some sociological challenges to methodological uptake. However, once the extensive benefits are understood, we anticipate widespread adoption of this approach.

Avainsanat – Nyckelord – Keywords
Jets, calibration, event, simulation, automation, computational, workflow, container, reproducibility, declarative

Säilytyspaikka – Förvaringställe – Where deposited

Muita tietoja – Övriga uppgifter – Additional information

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI
MATEMAATTIS-LUONNONTIETEELLINEN TIEDEKUNTA
MATEMATISK-NATURVETENSKAPLIGA FAKULTETEN
FACULTY OF SCIENCE

Tiedekunta – Fakultet – Faculty	Koulutusohjelma – Utbildningsprogram – Degree programme	
Naturvetenskapliga fakulteten	Magisters program	
Opintosuunta – Studieinrikting – Study track Teoretiska och beräkningsbara metoder		
Tekijä – Författare – Author Adelina Lintuluoto		
Työn nimi – Arbetets titel – Title Studier av jet energi korrigeringar på CMS experimentet samt automatiseringen av dem		
Työn laji – Arbetets art – Level Magisters avhandling	Aika – Datum – Month and year Februari 2021	Sivumäärä – Sidoantal – Number of pages 69

Tiivistelmä – Referat – Abstract

På experimentet Compact Muon Solenoid (CMS) vid CERN (Europeiska organisationen för kärnforskning) undersöks universums byggstenar genom att analysera de observerade sluttillstånds partiklarna som härrör från proton-proton kollisioner med hög energi. Direkt detektering av sluttillstånds kvarkar och gluoner är dock inte möjligt på grund av ett fenomen känt som färg inneslutning. Istället studeras händelse egenskaper med en nära överensstämmelse med deras distributioner. Dessa händelse egenskaper är kända som jets. Jets är centrala för partikelfysik analys och vår förståelse av dem, och därmed förståelsen av vårt universum, är beroende av vår förmåga att noggrant mäta deras energi. Tyvärr är nuvarande detektorteknik inte exakt, vilket kräver korrigering av mät avvikelser. För att uppnå detta använder CMS-experimentet en sekventiell flerstegs jet kalibreringsprocess. Processen utförs flera gånger per år, och oftare under datainsamling perioder.

Att automatisera jet kalibreringen skulle öka effektiviteten i CMS-experimentet. Genom att automatisera kod körningen kunde arbetsflödet utföras oberoende av analytikerns närvarande. Detta skulle i sin tur påskynda analysen och minska analytikerns arbetsbelastning. Dessutom möjliggör automatisering högre nivåer av reproducerbarhet.

I denna avhandling presenteras en ny metod för att automatisera härledningen av jet energi korrigeringar från simulering. För att uppnå automatisering använder metoden deklarativ programmering. Av analytikern krävs enbart att uttrycka *vad* som ska utföras och hen behöver inte längre avgöra *hur* den ska utföras. För att framgångsrikt automatisera beräkningen av jet korrigeringar är det nödvändigt att fånga detaljerad information om både beräknings stegen och beräknings miljön. Det förstnämnda uppnås med ett beräknings arbetsflöde och det senare med container teknologi. Detta möjliggör ett bärbart och skalbart arbetsflöde,, vilket är enkelt att förvalta och jämföra med tidigare körningar.

Resultaten av denna avhandling tyder starkt på att det är både uppnåeligt och fördelaktigt att fånga komplexa experimentella partikelfysik analyser med deklarativa arbetsflödes språk. Produktiviteten hos analytikern förbättrades och reproducerbarheten underlättades. Metoden är dock inte utan sina utmaningar. Deklarativ programmering kräver att analytikern tänker annorlunda om det aktuella problemet. Som ett resultat finns det ett par sociologiska utmaningar för metodens upptagningen. Men när de omfattande fördelarna har förståtts, förväntar vi oss en bred tillämpning av detta tillvägagångssätt.

Avainsanat – Nyckelord – Keywords
Jets, kalibrering, simulation, automation, beräkningsbar, arbetsflöde, container, reproducerbarhet, deklarativ

Säilytyspaikka – Förvaringställe – Where deposited

Muita tietoja – Övriga uppgifter – Additional information

Acknowledgements

First and foremost I would like to thank my thesis advisor Dr. Clemens Lange. Thanks for all the knowledge you have shared with me, thanks for all the support, and particularly thanks for providing me with constant constructive feedback.

I would also like to thank some people who indirectly helped me with the thesis. Thank you to Dr. Tibor Šimko, Diego Rodríguez Rodríguez and Marco Vidal García who I worked together with, developing REANA. I learnt a lot working with you.

I would like to acknowledge Dr. Kati Lassila-Perini for all the the support and encouragement she has given me. Thank you for being so exceptional and providing me with a great role model to look up to.

Thanks to associate professor Mikko Voutilainen for acting as my supervisor.

Adelina Lintuluoto
Geneva, February 2021

Contents

Symbols and abbreviations

The next list describes several symbols and abbreviations that will be later used within the body of the document

Abbreviations

CD	Continuous Delivery
CERN	European organization for nuclear research
CHS	Charged-Hadron Subtraction
CI	Continuous Integration
CMS	Compact Muon Solenoid, a LHC experiment
CMSSW	CMS SoftWare
CTF	Combinatorial Track Finder
CVMFS	CERN VM FileSystem
DAG	Directed Acyclic Graph
ECAL	Electromagnetic CALorimeter
EM	ElectroMagnetic
FSR	Final-State Radiation
HCAL	Hadronic CALorimeter
HPC	High Performance Computing
ISR	Initial-State Radiation
JER	Jet Energy Resolution
JES	Jet Energy Scale
JME	Jet and Missing Energy, group at CERN

LHC	Large Hadron Collider, particle accelerator at CERN
LO	Leading-Order
LV	Leading Vertex
MPI	Multiple Parton-parton Interactions
NLO	Next-to-Leading-Order
NNLO	Next-to-Next-to-Leading-Order
PDF	Parton Distribution Function
PF	Particle-Flow
PUPPI	PileUp Per Particle Identification
QCD	Quantum ChromoDynamics
RC	Random Cone
REANA	REproducible ANAlysis, platform created at CERN
RMS	Root-Mean-Squared
SM	Standard Model
VM	Virtual Machine

Symbols

α	Local metric used by the PUPPI algorithm
η	Pseudorapidity
$\langle\mu\rangle$	Mean number of inelastic interactions per bunch crossing or average number of pileup interactions
$\mathcal{L}$	Luminosity
ϕ	Azimuthal angle
θ	Polar angle
p_T	Transverse momentum
p_T^{ptcl}	Particle-level transverse momentum
p_T^{reco}	Reconstructed transverse momentum
R	Jet distance parameter
R_{ptcl}	Momentum response
y	Rapidity

Chapter 1

Introduction

The Compact Muon Solenoid (CMS) experiment at CERN (European Organization for Nuclear Research) facilitates investigation of the building blocks of the Universe through the analysis of high energy proton-proton collisions. The collisions generate complex cascades of particles, whose signals are recorded as they traverse through the detector. The signals recorded by the various subdetectors are then combined to reconstruct the particles generated by the collisions.

The task of experimental particle physics is to analyse the observed final-state particles resulting from collisions of the initial-state particles. A common strategy is to compare experimentally derived data and simulated data against theoretical models. The simulations start by generating the hard process at parton level, followed by confining the partons into hadrons. A simulation of the detector response is then applied to account for experimental factors in the discrepancies between observation and simulation.

Detection of final-state quarks and gluons is not possible because of colour confinement. Instead, event properties with a close correspondence with their distributions are studied. The final-state particles are clustered according to best estimates of the initial-state quark or gluon from which they originated. These clusters (event properties) are known as *jets*.

For a number of reasons, including electronic noise and detector effects, the measured energy of the jets does not precisely equal the real energies of the constituent particles. However, jets are central to particle physics analysis, and our understanding of jets is dependent upon our ability to accurately

measure their energy. It is therefore important to accurately account for and correct the discrepancies. To achieve this, the CMS experiment utilises a sequential multi-step jet calibration process. In this thesis I present a novel method for automating the calibration. This method functions by deriving the necessary corrections from simulated samples, and automates the computation of corrections for the effects of soft collisions overlaying the signal (known as pileup) and of the non-linear response of the detector.

Automation of the jet calibration process brings a plethora of advantages. Firstly, automation reduces analyst workload. This saves time, speeding up analyses and enabling fast feedback. This is highly valuable, as jet calibration must be performed several times per year (and more frequently during periods of data collection). Secondly, and perhaps even more importantly, automation facilitates higher levels of reproducibility.

To achieve automation, a methodology centered around the declarative, rather than the imperative, paradigm is developed. The analysis process therefore focuses on *what* to execute in each step without paying particular attention to *how* the individual computations might be performed by the computer. To achieve this, it is necessary to structure the description of the analysis by capturing detailed information concerning both the computational *steps* and the computational *environment*. The former is achieved with a computational workflow, and the latter using container technology. This allows a portable and scalable workflow to be achieved, which is easy to maintain and compare to previous runs.

However, automation is not without its challenges. Without human decision-making, it is necessary to introduce tools to monitor and diagnose individual steps within the workflow. Additionally, it is important to provide an easy way to restart a workflow at each step following manual intervention.

Within this thesis, Chapter 2 will provide an overview of the Standard Model, before going on to introduce crucial high-level concepts such as jets and the process of their reconstruction. This is followed by an introduction to event and detector simulation in Chapters 3 and 4 respectively. The methodology behind the event reconstruction is discussed in Chapter 5. Here, the different types of jets created at CMS are discussed with reference to their method of reconstruction. In Chapter 6 the discrepancies between the reconstructed and particle-level jets are discussed, along with the four-step process used to account for and correct these differences. The steps facilitate the correction of the effects of pileup, the non-linear detector response, the residual simulation-data jet energy scale differences and the

jet flavour biased differences. This is done with a particular focus on deriving corrections from simulation. Chapter 7 then outlines the methodology of automating the process of computing jet energy corrections. Analysis reproducibility in high energy physics is discussed, along with challenges facing the widespread adoption of automation. The results of the development of this novel automation methodology are presented in Chapter 8. This is followed by a discussion of the process in Chapter 9 and concluding remarks in Chapter 10.

Chapter 2

The Standard Model and jets

The *Standard Model* (SM) of particle physics is the most advanced theory currently used to describe elementary particles and their interactions. With the exception of gravity, the SM underpins all interactions. As the effect of gravity is negligible in the energy ranges of high energy physics, the SM is considered to be comprehensive in the vast majority of particle physics experiments. A detailed review of the SM is found in References [1, 2].

The SM is a quantum field theory based on the idea of local gauge symmetries. Local excitations in quantum fields are interpreted as fundamental particles. The three fundamental forces correspond to three symmetry groups of gauge field theory. Firstly, the Abelian Lie group $U(1)_\Upsilon$ describes the electromagnetic (EM) interactions. Secondly, the non-Abelian Lie group of $SU(2)_L$ describes the (electro)weak interactions, and finally the non-Abelian Lie group of $SU(3)_c$ describes strong interactions. The subscript terms Υ, L and c represent the weak hypercharge, the requirement for left-handed chirality and the colour charge respectively. Each Lie group has a corresponding gauge field. In the SM, all interactions are mediated by *gauge bosons*, quanta of the gauge fields.

This chapter will provide an overview of the elementary particles and their interactions of the SM, before going on to present high-level concepts such as jets and their reconstruction.

2.1 Elementary particles and their interactions

All matter is made up of electrically charged leptons and quarks, which are themselves spin-1/2 fermions and exist in three generations of different mass scales. The lightest particles make up the first generation, and constitute all stable matter in the universe. Heavier particles, found in the two other generations, decay into the lighter ones. The leptons are arranged in the three generations; the "electron" and the "electron neutrino" form the first generation, followed by the "muon" and the "muon neutrino", and then the "tau" and the "tau neutrino". Similarly, the quarks are paired in the three generations; the "up quark" and the "down quark", the "charm quark" and "strange quark", and finally the "top quark" and "bottom (or beauty) quark". The quarks are generally referred to by the first character of their name, e.g. the "u quark".

The interactions between elementary particles are mediated by the gauge bosons with spin-1. Of these, photons mediate EM interactions, $W^{\pm}$ and Z bosons are responsible for weak interactions, and the strong force is carried by gluons. Gauge bosons may also be referred to as *vector bosons*, due to the vector field they correspond to in quantum field theory. Although not a gauge boson, the Higgs boson also has integral spin (spin-0) and is a *scalar boson*, corresponding to the scalar Higgs field. The particles of the SM is presented in Figure 2.1. The figure additionally illustrates the coupling between the particles.

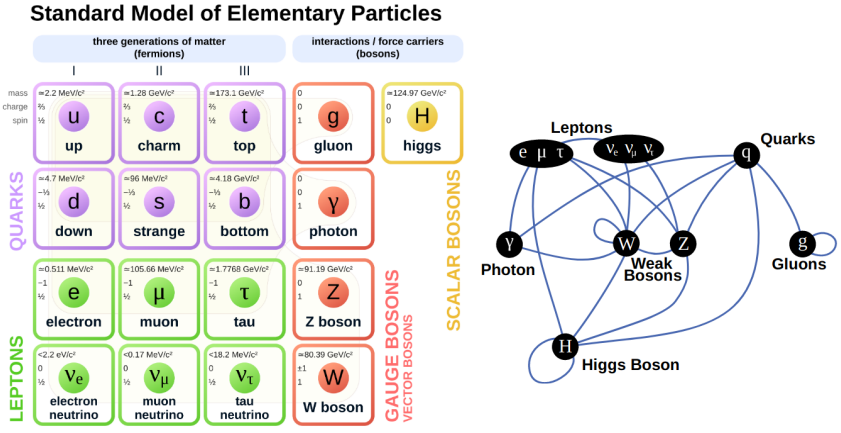


Figure 2.1: Table of the fundamental particles of the SM (left) and a graph illustrating the coupling between the particles (right) [3, 4].

Not all elementary particles interact with each force. Quarks experience EM, weak and strong interactions, while leptons only experience the two first. Unlike quarks, leptons do not carry the color charge related to the strong force and hence do not experience strong interactions. Strong interactions are at the core of proton-proton collisions and which are central to this thesis. Therefore, strong interactions will be described in more detail in Section 2.2.

The elementary particles obtain their masses via the *Higgs mechanism*, which manifests experimentally as the Higgs boson. With the exception of neutrinos, the Higgs boson couples with and generates the masses for all fundamental particles. The mechanism behind the generation of the neutrino mass remains unknown. However it is clear that a finite mass difference between the generations must exist to allow oscillations between flavour eigenstates.

2.2 Quantum chromodynamics

Quantum chromodynamics (QCD) provides an effective theoretical framework to describe strong interactions. Strong interactions merit consideration due to their importance in proton-proton collisions and in the subsequent formation of jets.

The strong force acts on the colour charge carried by quarks and gluons. Borrowing from the widely established red, blue and green colour system, the colour charge is described by red, green and blue "colours". Conversely, the antiquarks have corresponding "anti-colours".

QCD is based on the $SU(3)$ gauge symmetry group and has eight gauge bosons called gluons. The non-Abelian property of the symmetry group enables the interaction and colour exchange between the gluons themselves. The gluon self-interaction polarizes the vacuum which consequently increases the force linearly with distance greater than about a femtometer. This differs from the EM and weak forces, which become weaker as the distance increases. As a consequence, only at very high-energy momentum transfers, or equivalently at small distances, do quarks and gluons behave like free or weakly bound particles. This property is called *asymptotic freedom*. Another consequence is the phenomenon of *colour confinement*. The mutual interaction of gluons means that quarks can never exist in isolation.

At the separation of (anti)quarks, the potential energy necessary to overcome the colour field becomes so large that it is more energy efficient to break the bond by the emergence of a quark-antiquark pair out of vacuum.

2.3 Jets

Due to colour confinement, quarks and gluons cannot exist freely. Instead, they form colour-neutral hadrons in a process referred to as *hadronization*. No exact theory for hadronization is known, however there exist two successful models used by the event generators at CMS. Hadronization results in a collimated spray of particles, including hadrons as well as soft photons and leptons originating from secondary hadron decays.

Owing to colour confinement, detection of final-state quarks and gluons is not possible. Instead, event properties which have a close correspondence with their distributions are studied. These event properties are known as *jets*. To create a jet, the final-state particles are clustered according to best estimates of the initial-state quark or gluon from which they originated [5].

Jets are typically created from topologically-related energy deposits in the calorimeters, some which are associated with tracks of charged particles. However, jets may be defined during all stages of the event generation (presented in the top row of Figure 2.2). Using experimentally derived data, calorimeter jets are created from clusters of energy deposits in the detector's calorimeters. From simulated data, the calorimeter jets can be reconstructed based on the simulated detector response. These two are represented in the right-most jet definition of the figure. Using simulated data, jets can also be reconstructed directly from the stable particles of the collision (simulated events contain direct information of these particles). This is represented by the next-to-last jet definition in the figure. These jets are referred to as *particle-level* jets. In this thesis, particle-level jets are important when computing and correcting the jet response and as a tool to validate the overall corrections.

The clustering of final-state particles reduces the complexity of the event and facilitates analysis. In addition, these clusters are helpful when studying QCD processes. The importance of jets can be exemplified by considering the first ever detection of gluons at the TASSO detector at the DESY particle physics laboratory [6]. While gluons decay too quickly to be detected, their fragmentation trace could be observed as a jet. This jet

can be seen (in yellow) in the right-hand panel of Figure 2.3.

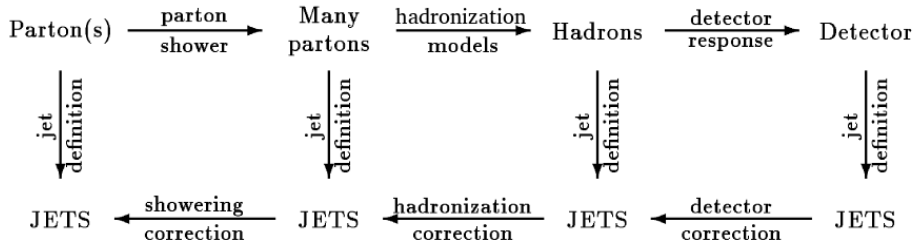


Figure 2.2: The typical stages of event generation as well as their corresponding jet definitions. Calorimeter jets are created from clusters of energy deposits in the calorimeters. Particle-level jets are reconstructed from the stable particles succeeding hadronization. Figure based on Reference [5].

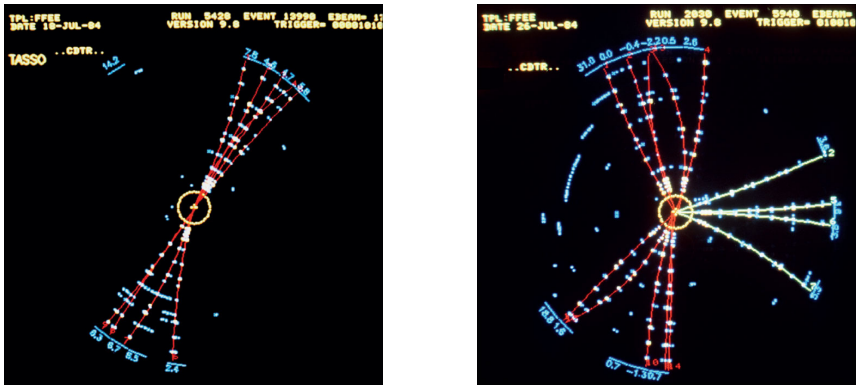


Figure 2.3: Computer display of two electron-positron interactions at the TASSO detector at the DESY particle physics laboratory. The yellow circle in the center represents the beam, where an electron and positron collides and annihilates, producing a virtual photon. The photon decays into a quark and an antiquark, which travel with high energy in opposite direction. As the pair travels they produce mesons due to colour confinement, creating two jets (in red). If the energy of the quarks are sufficiently high they may emit a gluon, creating a third jet (shown in yellow in the right-hand panel). The sum of the momenta of all the constituents of each jet corresponds to the energy of the original quark. The energy of the original quarks in turn, constituent to the energy of the original leptons. Image taken from the CERN Courier "The history of QCD", 27th September 2012. Originally from Oxford University PPU.

2.4 Jet clustering

The definition of a jet is dependent of the jet algorithm. Jet clustering algorithms function to reduce the complexity of the final state, by simplifying many hadrons to simpler objects. More specifically, the algorithm maps the momenta of the final state particles into the momenta of a certain number of jets.

Jet clustering algorithms may be classified as *cone algorithms* or *sequential recombination algorithms*. A study from 2015 on the performance of different jet algorithms found that the best algorithms for resolving jets as well as their substructure are found in the sequential recombination category [7]. Hence, those algorithms are favoured in the CMS experiment. The sequential recombination algorithm operates iteratively in the following steps.

$$d_{ij} = min\left(p_{T,i}^{2p}, p_{T,j}^{2p}\right) \frac{\Delta_{ij}^2}{R^2} \tag{2.1}$$

$$d_{ij} < p_{T,i}^{2p} \tag{2.2}$$

The algorithm starts by choosing a seed particle i at random. The distance between adjacent particle j and seed particle i is computed as $\Delta_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j)^2$, where y is the rapidity and ϕ the azimuthal angle. Next, a variable reflecting the computed distance and the transverse momentum (p_T) is calculated with the equation present in Equation 2.1. The distance parameter R defines the width of the jet. In the following step, the clustering algorithm combines particles iteratively by always choosing the particle j that minimises d_{ij}, in a manner that upholds Equation 2.2. The iteration stops when all particles satisfying Equation 2.2 have been clustered to the jet. Hence, the parameter R can also be regarded as a cutoff parameter. As the final step, the jet momentum is computed as the vectorial sum of all particle momenta in the jet. The algorithm then continues by choosing a different seed particle i for the next jet.

There are three clustering algorithms that differ in their choice of parameter p. The k_t algorithm has p equals 1, the Cambridge/Aachen algorithm p equals 0 and the anti-k_t algorithm p equals –1. The CMS software allows the reconstruction to be done with all three algorithms, however the most widely used clustering algorithm is the anti-k_t algorithm. The anti-k_t algorithm was also used in this thesis. It is insensitive to both infrared and

collinear divergences. The former means that the addition of soft infrared particles does not change the outcome of the clustering. The latter, that the addition of a collection of collinear particles with some total momentum is clustered identically to a single particle with the same momentum. This is necessary as to avoid bias from the threshold trigger of a calorimeter cell and the background noise. The anti-k_t algorithm is also largely favoured as it behaves like an idealised cone algorithm, shaping hard jets perfectly circular on the (y, ϕ)-plane [8].

The anti-k_T algorithm clusters all stable final-state particles (excluding neutrinos) with a decay length of $c\tau > 1$ cm [9]. The exclusion of neutrinos is a convention adopted by CMS, having only a small effect as the response is measured from samples with negligible neutrino content. However, additional systematic uncertainty should be considered for measurements including heavy hadrons fragmentation relative to the original b and c quarks.

Chapter 3

Event simulation

To aid in the extraction of useful information from data recorded by the detector, accurate models of the event kinematics at parton and particle level are simulated to a high degree of accuracy. Simulated events are produced by general-purpose Monte Carlo event generators, such as PYTHIA [10] and HERWIG [11]. Additional software (e.g. MADGRAPH [12] and POWHEG [13]) is commonly used to generate parton-level events. Their output is subsequently directed to the two previously mentioned event generators for further processing. At the CMS, simulated events are stored in the same format and reconstructed in the same way as experimentally-derived data. Simulated events are necessary in various parts of experimental particle physics. As presented in this thesis, simulated events are used in conjunction with detector simulation to estimate the signal and background of high-energy collisions.

The production of proton-proton collision simulated events at the LHC can be separated into four steps, which will be outlined in this chapter. The steps can be seen illustrated in Figure 3.1 and listed below:

1. The generation of the hard process, a few-body interaction which is modeled using pertubation theory.

2. The forward and backward evolution of parton showers, which systematically adjust the event towards a more realistic model.

3. The hadronization of the partons, where final-state partons are confined into colorless hadrons, and initial-state partons interact with

additional protons in the beam.

4. The modeling of secondary interactions between the partons.

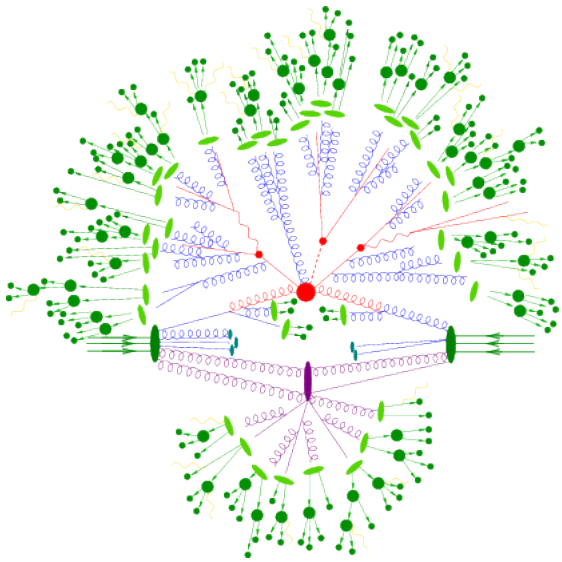


Figure 3.1: Schematic illustration of a hadron-hadron collision as simulated by a Monte-Carlo event generator. The two beams are illustrated with green arrows on each side of the figure. The red circle in the center signals the hard collision of the two partons originating from the beams. The hard scatter event is surrounded purple coils representing the bremsstrahlung from parton showers. The purple ellipse indicates a secondary hard scattering event. The hadronization is represented by light green ellipses, while dark green circles indicate hadron decays. Yellow lines represents the photon radiation [14].

3.1 Hard process

In proton-proton collisions, the center-of-mass energy of the partons participating in the hard collision varies from event to event according to their parton distribution function (PDF). The PDF $f(x, \mu)$ describes the probability that a quark or gluon is found carrying a fraction x of the total momentum of the incoming proton when it is probed at an energy scale μ. The PDF connects the initial-state protons to the parton-level hard

process. The PDF is measured experimentally, as it is determined by the non-perturbative physics inside the proton.

The probability amplitude of a process can be calculated by summing the terms corresponding to the Feynman diagrams up to a chosen order. Normally the leading-order is sufficient to describe the main features of the process, although some models use next-to-leading order. The square of the amplitude gives the matrix element of the process, which provides information about the strength of the transition between the initial- and final-state.

The generation of the hard process starts by selecting two partons from the colliding protons according to their PDFs. These are taken as the initial-state particles for which one can calculate the matrix element and subsequently the cross-section of the physical process. The distribution of the hard event is then predicted by theory based on the cross-section. Using the cross-section, the phase space is sampled and candidate events are chosen on random by a uniform distribution. As the quarks and gluons are present as free particles in the initial- and final-state, the energy of the collision must be high enough for the partons to acquire asymptotic freedom.

3.2 Parton shower

The event simulation increases in complexity as initial partons generate finite amounts of bremsstrahlung radiation, either before or after participating in the hard process. Coloured partons may emit gluons, and due to the non-Abelian nature of QCD, gluons may also themselves emit gluons. These emissions give rise to a cascade of partons which are continuously being scattered, annihilated and created until they reach the *infrared cutoff* at about 1 GeV [15]. The resulting cascade of partons is called a *parton shower*.

To model parton showers, an approximate method is formulated using probability distributions for branching partons. The process results to the formulation of a step-wise Markov chain.

In the parton showering model the emissions are separated into initial-state radiation (ISR) and final-state radiation (FSR). ISR is showered by backward evolution, from the hard process, in a time-reversed way towards

the incoming protons. Conversely, FSR evolves forward in time, creating new branches until reaching the infrared cutoff [16].

Event generators model parton showering in different ways. PYTHIA uses p_T-ordered dipole showers in which the hardest emissions come first. HERWIG employs angular-ordered parton showers, where the branching with largest angles are performed first to ensure correct treatment of soft gluons. A more detailed description of the parton showering models and the related calculations schemes can be found in Ref. [14].

3.3 Hadronization

When the momentum transfer reaches the previously described infrared cutoff, the perturbative QCD breaks down, and further handling of the event is done with a non-perturbative model. This is when hadronization happens. Hadronization describes the process by which coloured partons are transformed into colourless primary hadrons, which may decay in turn to secondary hadrons. Non-perturbative in nature, event generators describe the transition by fine tuning free parameters according to the results of carefully executed experiments.

At the hadronization stage, partons are no longer treated independently, but rather as a colour connected system. The implementation of parton hadronization differs between event generators. PYTHIA applies a hadronization model commonly referred to as the Lund model. This model is based on *linear confinement* (the linear growth of the potential energy between partons of opposite charge, when separated at a distance greater than about a femtometer). In the Lund model, quarks are connected with a "colour string". As the quarks become more distanced and the string grows, the potential energy exceeds the order of a hadron's mass. At this point it becomes more energy efficient to break the string by the emergence of a quark-antiquark pair out of a vacuum. HERWIG uses a cluster hadronization model based on *preconfinement*. At the end of the parton shower, all gluons are separated into quark-antiquark pairs. Next, colour-singlet combinations of partons, called clusters, are formed. These clusters subsequently form stable hadrons, ending the generation of new partons.

3.4 Underlying event

Due to the composite nature of the LHC's beam particles, the proton-proton collision is dominated by soft QCD events referred to as the *underlying event*. The underlying event represent all additional processes not directly associated with the hard interaction. Some ambiguity exists in how the hard interaction is described, but it is commonly defined as all processes that occur after the associated ISR and FSR.

The main cause of the underlying event is the additional colour exchanges between the remnants of the colliding protons. These exchanges are modeled as perturbative *multiple parton-parton interactions* (MPI), and are handled in a manner analogous to ISR and FSR [17]. MPI add further complexity due to subsequent parton showers which may combine with the showers from the hard interaction.

The beam contains more than a hundred billion protons per bunching, and each proton constitute of three valence quarks and an interactive sea of antiquarks, quarks and gluons. Not every parton participates in a hard interaction; most take part in soft secondary interactions. The soft interactions consist mostly of multiple-parton interactions and diffractive scatterings that produce low momentum particles. As the associated momentum transfer is low, perturbative QCD is not applicable and numerical models are used. The non-perturbative models use free parameters that are tuned according to experimental results. Eventually, particles of the underlying event are also subject to hadronization described in Section 3.3.

The simulated events used in this thesis are QCD multijet events, i.e. events with final-states consisting of a high multiplicity of jets. The events are generated using Pythia 8.230 with tune CP5. Modeling of the ISR, FSR, hard scattering, and MPI was done with PDFs at next-to-next-to-leading order (NNLO) [18]. The hard process were generated with collision energy 13 TeV and the events have transverse momentum ranging from 15 to 7000 GeV.

Chapter 4

Detector simulation

Following the previously described steps of event simulation, the simulated event is what an "ideal detector" would measure. In order to simulate a realistic response as the particles interact with the detector materials, the next step is to process the event with *radiation transportation software*. Such software simulates the propagation of particles and their energy deposits in the detector. The events for this thesis were processed through a CMS detector simulation based of GEANT4 [19].

The full simulation workflow, illustrated in Figure 4.1, includes the event generation, the detector simulation, the addition of pileup and the digitization of the signal. To understand the full extent of detector simulation software such as GEANT4, knowledge of the detector itself is needed. Therefore, before discussing the key concepts of detector simulation, a brief overview of the CMS detector will be given.

4.1 CMS detector overview

The CMS is a multi-purpose detector, operating at the Large Hadron Collider (LHC) at CERN. The detector is situated at one of the *interaction points*, where the two LHC beams collide. While the detector is designed to measure both proton-proton and heavy ion collisions, this thesis will focus on the former.

To describe the spatial coordinates of particles traversing the detector, the

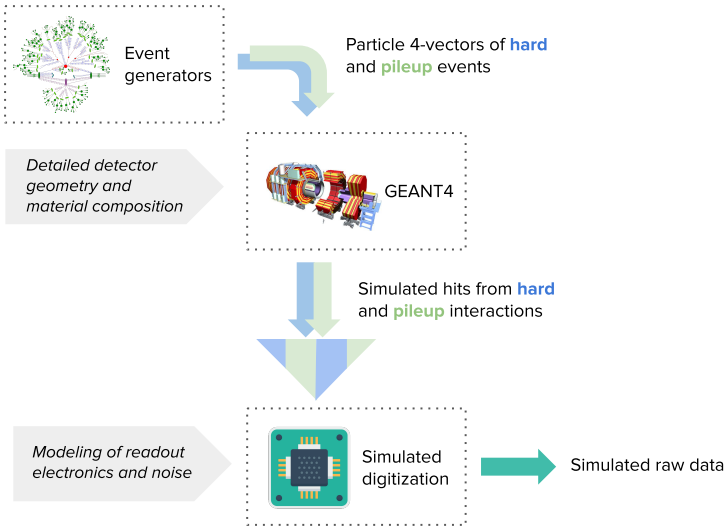


Figure 4.1: Illustration of the simulation workflow. After event generation, the detector response is simulated as the particles propagate through a detailed simulation of the CMS detector. The hard process and pileup events are generated separately. Both type of events are then combined before simulating the conversion of energy deposits to electric signals (referred to as digitization). The output of this step is known as "raw data".

CMS experiment has adopted a right-handed coordinate system with an origin at the beam interaction point. The plane of the x-axis points radially inwards towards the center of the LHC ring, while the y-axis points vertically upwards (perpendicular to the plane defined by the LHC ring). The x- and y-axes span the transverse plane, where the azimuthal angle (ϕ) is defined. The z-axis corresponds to the longitudinal axis of the CMS detector and points along the direction of the anticlockwise beam. The polar angle (θ) describes the angle of a particle with respect to the z-axis. *Pseudorapidity* (η) is then calculated from θ, as below.

$$\eta = -\ln\left(\tan\frac{\theta}{2}\right) \tag{4.1}$$

η is the term commonly used to describe the angle of the particle relative to the z-axis.

The CMS detector can be considered as comprising three regional segments

spanning different ranges of η. The *barrel* region describes $|\eta| < 1.3$, the *endcap* region $1.3 < |\eta| < 3.0$, and the *forward* region $3.0 < |\eta| < 5.0$. Additionally, the parts of the endcap region that are within and outside the tracker coverage are known as EC1 ($1.3 < |\eta| < 2.5$) and EC2 ($2.5 < |\eta| < 3$) respectively. The structure of the detector is illustrated in Figure 4.2. The range of η is defined on the top and the right margins.

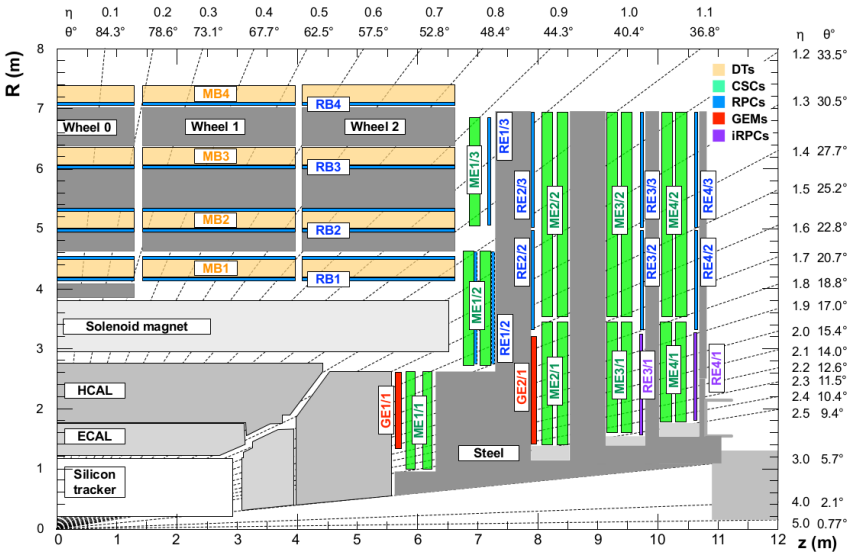


Figure 4.2: One quadrant of the CMS detector shown in the (z,y)-plane. The tracker, the ECAL and the HCAL (both defined in the following paragraph) are shown along with the muon chambers. The range of η is illustrated on the top and the right margins [20].

As the barrel and endcap particles produced from the beam interactions traverse through the detector, they pass through several layers of different subdetectors. The subdetectors measure the trajectories and energies of these particles. The subdetector closest to the interaction point is the silicon tracker, which measures trajectories of charged particles. As seen in Figure 4.2, the tracker is surrounded by the electromagnetic calorimeter (ECAL) which functions to absorb and measure the energy of electrons and photons. The next subdetector (the hadronic calorimeter - HCAL) absorbs and measures the energy remaining from showers initiated in ECAL, as well as any hadronic showers initiated in the HCAL itself. A superconducting solenoid magnet then encloses the tracker and calorimeters. The magnet provides a strong magnetic field which bends the trajectories of charged particles, allowing precise measurement of their momentum and charge.

Finally, multiple muon chambers alternate with layers of the return iron yoke.

In contrast to the barrel and endcap particles, particles in the forward region close to the beam interact with forward calorimeters.

4.2 GEANT4

All components, including active detector parts and passive materials such as cables and cooling systems, are modeled in GEANT4. The central detector alone (consisting of the tracker, the calorimeters and the muon subsystem) constitute of more than one million geometrical volumes [21]. Details about the detector geometry and material composition is provided to GEANT4 with a dedicated software package. Concepts such as magnetic field type and propagation parameters are also configurable.

GEANT4 simulates the particle interactions using stochastic methods. The software applies different stochastic processes depending on the probability of interaction. The software describes several interactions including ionization, bremsstrahlung and multiple scattering as well as detector properties including tracking, geometry description and material specifications. The modeling of interactions extend over a large range of energies, from elastic scattering at the MeV scale to hadron showers at the GeV and TeV scale.

4.3 Pileup overlay

During high luminosity ($\mathcal{L} = 15 \times 10^{33}\ \mathrm{cm}^{-2}\,\mathrm{s}^{-1}$) in 2018, LHC produces an average of about 32 inelastic proton-proton collisions per bunch crossing [22]. The distributions of the mean number of inelastic interactions per crossing for the years 2016, 2017 and 2018 are presented in Figure 4.3. These additional collisions "pile up" on on top of the signal (hard scattering). As the simulation of pileup depends on the LHC luminosity and run conditions, it is simulated separately from the signal. Pileup is added to the hard process events by mixing minimum-bias events (soft QCD interactions with a minimum number of tracks). The minimum-bias events are generated and processed through the simulation steps as described in Chapter 3 and Section 4.2. After being processed by the detector simu-

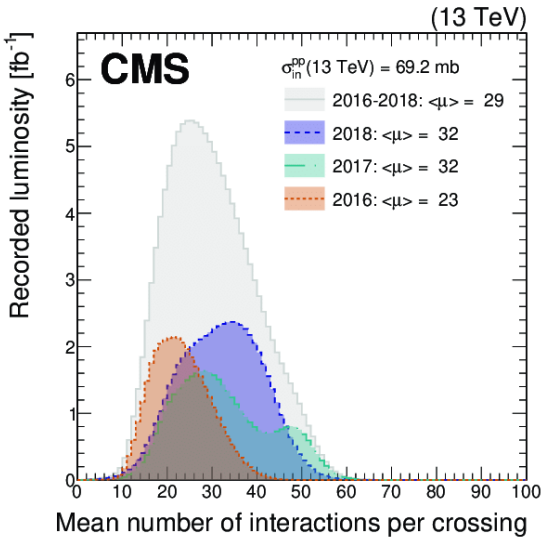


Figure 4.3: Distribution of the mean number of inelastic interactions per crossing (also known as pileup) in data for proton-proton collisions in 2016 (dotted orange line), 2017 (dotted dashed light blue line), 2018 (dashed navy blue line), and integrated over 2016–2018 (solid grey line) [22].

lation software, their energy deposits are mixed with the deposits of the original hard process events. The events of this thesis were overlaid with pileup generated with Pythia 8.230 tune CP5.

A pileup distribution is used to compute the number of minimum-bias events to include per hard process event. The pileup distribution describes the mean number of interactions per bunch crossing. It is created from experimental data or a simulated distribution which is later re-weighted. For each hard event, the mean number of interactions per bunch crossing is extracted from the pileup distribution. A Poisson distribution, using this value as the mean, then reports the amount of pileup. To avoid correlations between events with the same pileup event sequence, pileup samples are never reused in the same order.

Pileup that occurs within the same bunch crossing as the collision of interest is described as in-time, and pileup occurring in the previous and subsequent bunch crossings are referred to as out-of-time. The pileup distribution includes both. The correct modeling of out-of-time pileup requires the inclusion of several bunch crossings around the original bunch crossing in the simulation. Consequently, hundreds of minimum-bias events may be

needed to model pileup for one generated hard event. Simulating pileup is often the most time-consuming process.

In cases when the simulation campaign starts prior to completion of data collection, the pileup distribution for data is unknown when simulating pileup. To ensure event generation with small statistical uncertainty covering the full kinematic phase space at the LHC, the pileup events are generated with a flat p_{T} spectrum. Then, prior to analysis, the simulated events are re-weighted. The re-weighting matches the pileup distributions between data and simulation.

4.4 Digitization

After collection, all detector energy deposits are converted to electric signals during a step referred to as *digitization*. Digitization is performed by CMS software, simulating the behavior of the readout electronics used in experimental data acquisition. To obtain a realistic detector response the simulation includes electronic noise.

Each subdetector has dedicated software for simulating the electronic readout response. Charged particles traversing the active elements of the tracker, distributes energy loss along their trajectory. Therefore, the signal modeling of the tracker readouts include Landau fluctuations as well as Lorentz drift and diffusion of charges to the detector surface. The modeling also accounts for noise and coupling between channels. Since the process is similar in all the calorimeter subsystems, digitization of calorimeter signals use a unified framework. Modeling of ECAL and HCAL takes particular care in simulating the efficiency and non-uniformity in the photon collection of the crystals and modules. To correctly simulate overlay of out-of-time pileup with the signal, the model simulates signal pulses as a function of time per each hit. In the muon drift tubes, focus is put on the particle direction and impact position with respect to the sense wire. Due to air gaps where detectors are placed within the iron yoke, modeling of the muon detector accounts for the residual magnetic field effects this creates [23].

At this stage, the simulation of Level-1 trigger electronics is added. The simulated events are stored in a format identical to that used for experimental data. Similar to data, each simulated event contains the information of the Level-1 trigger it would have passed if it were a real event. This allows for the use of the same reconstruction algorithms and tools in simulated

and experimentally-derived data.

Chapter 5

Event reconstruction

Following event and detector simulation, each event is reconstructed using the *particle-flow* (PF) algorithm [24]. The purpose is to bring the recorded picture of the collision event back to particle-level. Combining information from all subdetectors, the goal is to reconstruct each type of particle separately. This requires the matching of tracks and energy clusters between the tracker and calorimeters. This combinatorial approach, illustrated in Figure 5.1, provides a more accurate event reconstruction than if relying on a single detector. The PF algorithm aims for a complete global event description. This includes a list of all particles present in the event and their properties such as charges, trajectories, and momenta.

The output of the PF algorithm is a set of PF candidates, classified into electrons, muons, photons, and charged and neutral hadrons. Hadrons and photons are further clustered into jets. Some jets may additionally be identified as originating from b quark hadronization or from hadronically decaying tau leptons. The presence of weakly interacting neutral particles is identified by calculating the missing transverse momentum of the event.

The PF approach was originally designed in the ALEPH experiment, which recorded the positron-electron collisions at the LEP [27]. The CMS now widely uses the PF method when reconstructing events. The PF method benefits from the CMS experiment due to the increased precision in charged hadron measurements, arising from the highly segmented ECAL. In addition, the muon tracking system provides an excellent muon identification. The stages of event reconstruction using the PF method will now be discussed.

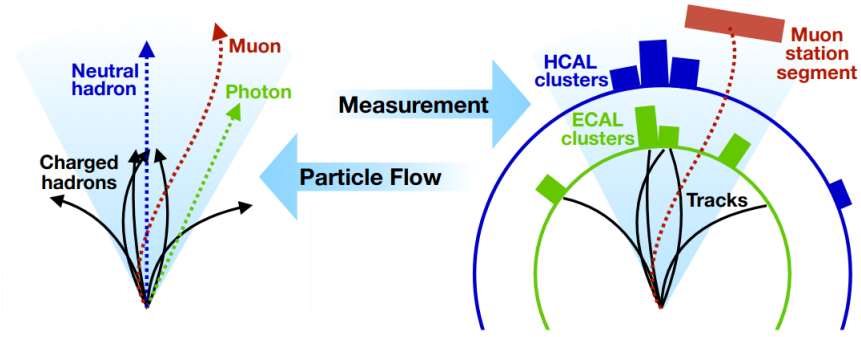


Figure 5.1: Schematic overview of the PF method. Information from all subdetectors are combined to produce a list of candidates ideally matching the particles present in the real event. Figure taken from Reference [25], originally adapted from Reference [26].

5.1 Tracking

The reconstruction of tracks from charged particles is achieved using an iterative method called *combinatorial track finder* (CTF). The CTF is based on *combinatorial Kalman filtering* and operates in three steps.

First, a few recorded hits compatible with a trajectory, called tracker seeds, are identified. Next, hits from the trajectory are gathered using pattern recognition software. Thirdly, a selected function is fitted to all hits to determine the full track and its momentum.

This process is repeated ten times, using different seeds and quality criteria each time in order to find the most optimal track [28]. The first few iterations target the highest-quality tracks. Hits associated with a reconstructed track are removed from subsequent iterations, gradually reducing the combinatorial complexity. This reduced complexity increases the spare capacity of the computing system, allowing more intricate seeding, filtering and fitting methods to be used, maintaining output quality. This improves the precision with which the low-quality track input can be reconstructed. Some iterations are designed to focus on specific classes of tracks, such as those with two hits missing in the pixel detector or those originating from very displaced vertices.

5.2 Calorimeter clustering

The electromagnetic showers in the ECAL and the hadron showers in the HCAL are wider than a single ECAL crystal or HCAL module. To correctly determine the energy deposits in these crystals and modules the deposits must be clustered.

The clustering algorithm starts by identifying clusters which exceed a given energy threshold. These clusters then function as seeds for the next step. Adjacent cells (eight in the ECAL and four in the HCAL) are mapped to the seeds. Next, topological clustering is performed, further mapping nearby cells (as long as the energy in the candidate cells are at least twice as large as the noise level). Finally, an iterative maximum-likelihood fit based on a Gaussian-mixture model is used to reconstruct the energy deposit inside each cluster. Different parameters for seeding, clustering and fitting are used in each calorimeter.

Clustering of energy deposits is essential for the detection of the momentum of photons and neutral hadrons. The clusters are also needed when reconstructing electrons and bremsstrahlung photons. To account for the bremsstrahlung photons emitted by electrons, the pair production of photons into positron-electron pairs, and the bent trajectories of the electrons, electromagnetic showers in ECAL identify *superclusters* with small width in η but large coverage in ϕ. Moreover, preshowers detected in the sampling calorimeters are extrapolated to the superclusters and have their energy deposits added into the superclusters.

5.3 Particle Flow linking

Once all the *PF elements* (tracks, calorimeter clusters and muon chamber signals) have been reconstructed, a *linking algorithm* is used to match them together. Taking into account the bent trajectories from the magnetic field, a link is created by extrapolating between the different elements.

The ECAL and HCAL clusters as well as the ECAL superclusters are linked by geometrical matching. Tracks originating from the same secondary interaction vertex are linked, and finally, tracks from the tracker and signals from the muon detectors are linked together. In the case of multiple possible trajectories, the trajectory with the smallest distance between two

elements is kept.

Sudden emission of bremsstrahlung in the tracker can change the direction of the electron trajectories. To improve the quality of reconstruction, the electron candidates (tracks linked to ECAL superclusters) are refitted with a Gaussians-sum filter algorithm, taking into account the possible energy losses due to bremsstrahlung. In addition, a dedicated algorithm takes care of tracking photons to potential positron-electron pairs arising from pair production.

Once the links have been made, *PF blocks* are created. Each PF block is a collection of elements linked together by association of the same single particle or group of particles. The identification and reconstruction of PF candidates is carried out in each PF block separately.

5.4 Particle Flow candidate reconstruction

Muons propagates through the calorimeters with minimal interaction. As they leave the clearest signal, muons are reconstructed first, using information from the tracker and muon chambers. The high separation of muon signals from the rest has lead to an impressive 99% efficiency in muon reconstruction [29].

Next electrons and isolated photons are reconstructed. This is done with tracks and ECAL clusters as well as isolated ECAL clusters, respectively. Electrons passing through the detector are likely to emit bremsstrahlung. As prompt and bremsstrahlung photons tend to convert through pair production, which subsequently decays by emitting photons, it is convenient to use a common approach for reconstructing electrons and isolated photons.

Hadrons originating from the hadronization of jets are then reconstructed with the information of ECAL and HCAL. Due to the fine granularity of ECAL (25 times finer than HCAL), neutral hadrons can be spatially separated from charged ones by combining the information from both calorimeters. Neutral hadrons deposit an excess of energy in HCAL, compared to what is expected based on information from ECAL. They can thus be reconstructed separately, even when they have deposited their energy in the same cells as charged hadrons.

Jets are reconstructed by combining existing PF objects with the use of jet

algorithms (such as the anti-k_T algorithm explained in Section 2.4). The PF algorithm is very useful as it allows jets to be presented as a collection of individual constituents, compared to a single object. Finally, weakly interacting neutral particles are determined by calculating the missing transverse momentum, which first relies on the reconstruction of all PF objects in the event.

5.5 Jet reconstruction

Different types of jets are created in the CMS experiment. The type of jet depends on the method of reconstruction, which differ by the use of different kinds of detector inputs and reconstruction algorithms. The different jets are the following:

- PF jets are reconstructed from particle-flow candidates.
- PFchs jets are reconstructed from particle-flow candidates where charged-hadron substraction (CHS) has been performed. CHS is explained further in Section 6.1.1.
- PFPuppi jets are reconstructed from particle-flow candidates using the pileup per particle identification (PUPPI) algorithm. PUPPI is explained further in Section 6.1.1.
- Calo jets are reconstructed by solely utilising the energy deposits from the calorimeters.
- JPT jets are reconstructed by combining the charged particle tracks to the spatially associated Calo jets.

An important parameter governing any algorithm is the *jet distance parameter*, which specifies the size of a cone-shaped jet in the (η, ϕ)-plane. The parameter is specified as $R = \sqrt{(\Delta\eta^2 + \Delta\phi^2)}$. The default jet distance parameters for Run 2 are 0.4 and 0.8. At the CMS, jet distance parameter 0.4 is mainly used for reconstruction of showers from light quarks. Conversely, 0.8 is used for heavy particles such as W, Z, Higgs bosons and beyond SM particles created with a large Lorentz boost. The particles decays of heavy boosted particles are often very collinear. This makes jet reconstruction difficult when using a small jet distance parameter and hence it is better to reconstruct all decay products within one jet with a larger size of 0.8 [22].

The reconstructed jets typically have the shorthand names [AA][CS][TY], where [AA] represents the clustering algorithm, [CS] represents the jet distance parameter and the [TY] represents the jet type. Resulting names include AK4PFchs or AK8PFPuppi.

Chapter 6

Jet Energy Correction

As the jets traverse through the CMS detector, signals are left in the detector components. The individual signals are combined to PF candidates, which jet algorithms use to produce reconstructed jets. However, due to an array of effects, the energy of the reconstructed jets do not precisely equal the particle-level jet energies.

The discrepancies between PF and particle-level jets arise from various effects, including energy deposits from pileup interactions, non-linear calorimeter response to hadrons, minimum energy thresholds in calorimeters and nuclear interactions in the tracker material. To account for and correct for these differences, jets are calibrated via a sequential multi-step process. The steps facilitate the correction of the effects of pileup, the non-linear detector response, the residual simulation-data jet energy scale (JES) differences and the flavour biased differences. The two first steps are applied to both simulated events and experimentally-derived data. The third step is only applied to data, in an attempt to bring it closer to the simulation. The fourth step is optional in both cases. As this thesis focuses on deriving corrections from simulation, only the two first steps are further discussed in Chapter 8 and 9.

The computed corrections are applied as scale factors for the jet four-momentum, factors which depend on various jet related quantities such as jet p_{T}, η and area A_{jet}, as well as the pileup p_{T} offset density ρ in the event. Finally, closure tests contribute to the validation of the corrections. Within simulated jets, corrections are derived separately for the PF, PFchs, Puppi, Calo and JPT jets. Conversely, the residual corrections derived from

experimental data are computed specifically for PFchs jets (with $R = 0.4$) and assumed to be the same for all other kind of jets.

The JES is critical to many physics analyses, and its precision makes an important contribution to their systematic uncertainties. In addition to comparing the jet cross-section to QCD predictions, accurate jet measurements are important for a variety of processes such as calibrating high-mass particles decaying to quarks and gluons, removing soft contamination from hard jets, tagging heavy objects originating from jets and for background subtraction. A better understanding of the JES and its uncertainties allows for more precise analysis measurements.

6.1 Pileup offset corrections

The additional contribution of jet energy and momentum due to pileup is referred to as pileup offset. The pileup offset correction adjusts the energy by estimating and then removing the energy corresponding to pileup inside the jet. Pileup removal is performed with both experimentally-derived data and simulated events. This chapter will consider two different approaches of pileup mitigation; that performed at per-particle level before jet clustering (CHS, PUPPI [22]), and at per-jet level after jet clustering (area-median [9]).

6.1.1 Pileup mitigation before jet clustering

The two most widely used methods within the CMS experiment operating at the PF candidate level are the CHS and the PUPPI method. Unlike the CHS method, which rejects only charged particles associated with pileup vertices, PUPPI applies a more rigorous selection to charged particles as well as rescales the four-momenta of neutral particles according to their probability of originating from the primary collision of interest, the *leading vertex* (LV). Both methods act on the particles, complementing the pileup subtraction algorithms performed after jet clustering (see Section 6.1.2) which work at the event or jet level. The pileup-corrected particles emerging from both CHS and PUPPI are used as input to jet clustering algorithms. CHS clustered jets are further corrected for residual neutral offset by the event or jet level pileup mitigation algorithms, while no such correction is necessary for PUPPI jets.

In CHS, charged particles originating from pileup vertices are identified and removed from the event. Charged hadrons are identified in the PF algorithm as tracks, which are associated with calorimeter hits. The LV is chosen based on the largest sum of squares of the track transverse momenta [22]. Subleading vertices are classified as pileup vertices, and are required to pass further quality criteria based on a minimum number of degrees of freedom in the vertex fit. If the track of a charged hadron is associated with a pileup vertex satisfying the above criteria, it is considered as pileup and removed.

The PUPPI algorithm builds on the CHS algorithm. The addition is based on the observation that neutral particles from a parton shower are typically aligned with charged particles from the same shower, while the particles from pileup vertices are more uniformly distributed in all directions.

The PUPPI algorithm operates as follows. Firstly, charged particles are assigned weights based on their available tracking information. Particles used in the fit of the LV are assigned a weight of 1, while those associated with a pileup vertex are assigned a weight of 0. Next, the algorithm defines a local metric α which differs between leading and pileup vertices. The variable α is constructed to be large for particles close to the LV, and for particles with $|\eta| > 2.5$, to be close to highly energetic particles [22]. At CMS the variable α for a given particle i is defined in Equation 6.1. Here, variable j are all other particles at a distance R_{ij} less or equal to $R_0 = 0.4$, $p_{T,j}$ their transverse momentum in GeV and $\Delta R_{ij} = \sqrt{(\Delta\eta_{ij}^2 + \Delta\phi_{ij}^2)}$. In $|\eta| < 2.5$ where tracking information is available, only charged particles associated with the LV are considered. Finally, the unique α-distributions for leading and pileup vertices are computed.

$$\alpha_i = \log \sum_{j \neq i, \Delta R_{ij} < R_0} \left(\frac{p_{T,j}}{\Delta R_{ij}} \right)^2 \tag{6.1}$$

$$\begin{cases} \text{for } |\eta_i| < 2.5,\ j \text{ are charged particles originating from the LV,} \\ \text{for } |\eta_i| > 2.5,\ j \text{ are other reconstructed particles,} \end{cases}$$

The algorithm uses the α-distributions for charged particles considered as pileup, to generate expected event-level pileup distributions. The weights of neutral particles are assigned by comparing their α values to the median and root-mean-squared (RMS) of the charged pileup distribution. Small weights encode the probability that the particle originated from pileup.

Finally, the weights are used to rescale the neutral particles four-momenta to correct for pileup at particle-level. Particles with very small rescaled p_{T} are discarded.

Figure 6.1 shows the jet energy resolution (JER) as a function of jet p_{T} for PF, PFchs and PFPuppi jets. PF jets perform worse at low p_{T} as the jets in that region are greatly affected by pileup. In the forward region ($3.2 < |\eta| < 4.7$), where no tracking is available, PFchs jets perform the same as PF jets. PFPuppi jets are superior when $R = 0.8$, as neutral particles from pileup contribute greatly to such jets. The preferred algorithm depends on the analysis. Better pileup mitigation is achieved with PUPPI compared to CHS, especially for events with more than 30 interactions. While the PUPPI algorithm provides a better jet p_{T}-resolution for jets with $p_T < 100$ GeV, CHS is superior when p_{T} exceeds 100 GeV [22].

6.1.2 Pileup mitigation after jet clustering

Following the optional per-particle pileup mitigation, particles are clustered into jets and pileup removal on event and jet-level is performed. The pileup offset correction is taken from the true offset in simulation, and for experimental data the correction is further scaled by the ratio of random cone (RC) offsets for experimental data and simulation. This will now be considered in more detail.

Firstly, QCD multijet events are simulated both with and without pileup overlay. Generally, the latter sample is completely without pileup. However, due to misconfiguration for samples with $\langle\mu\rangle$ equal to 0, in this thesis as sample with $\langle\mu\rangle$ about 10^{-8} was used. Particle-level jets in both samples are matched, required to be within a distance less than $R/2$ [9]. Here, R is the jet distance parameter. Next, the particle-level pileup offset is calculated as the average difference in p_{T} between the matched jets. To parametrize the pileup dependence, the diffuse offset energy density of the event is estimated as the median of energy deposits in η/ϕ bins encompassing the whole detector. Finally, the diffuse pileup offset energy per event is estimated. The corrections are applied as a function of the uncorrected jet p_{T}, η, ρ and jet area defined as the (η, ϕ)-plane where the particles are clustered.

For experimental data, the corrections are further scaled by offset scale factors. These factors are determined with the RC method in simulated and

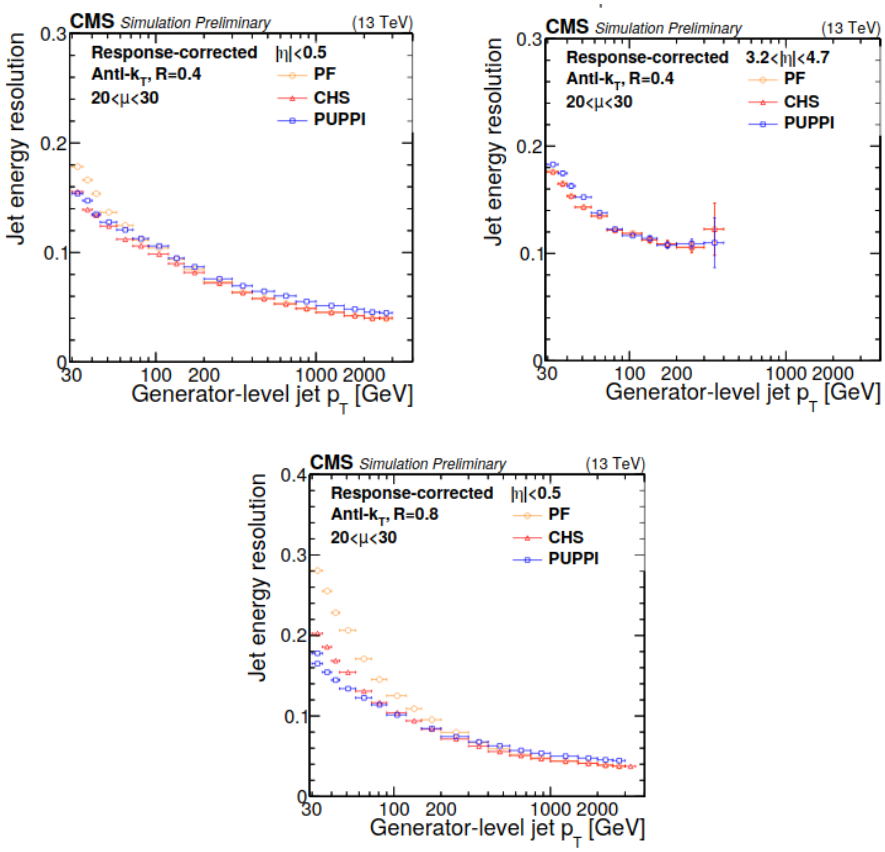


Figure 6.1: Jet energy resolution as a function of generator-level jet p_{T}. PF jets are shown with orange circles, PFchs jets with red triangles and PFPuppi jets with blue squares. The two top figures used AK4 jets with $|\eta| < 0.5$ (left) and $3.2 < |\eta| < 4.7$ (right), while the bottom figure used AK8 with $|\eta| < 0.5$ The number of pileup interactions was on average between 20 and 30 [22].

zero-bias data. Zero-bias events are triggered randomly, and consequently do not usually contain any hard scattering; only pileup (the zero-bias sample is further weighted according to the luminosity measurement in order to represent the average pileup conditions of the dataset used in the analysis). Firstly, the energy from PF candidates are added in cones placed at random in the (η, ϕ)-plane. The average p_{T} of the jets measured in each event is an estimate of the average energy density clustered into a jet. When the method is applied to zero-bias events the main contribution to the jet energy arises from noise and pileup. Assuming noise is negligible, the average p_{T} of the jets is an indication of the average energy offset due to pileup. Finally, the offset scale factor are derived by fitting the RC measurement with a quadratic function and the ratio is computed between the scale factors for zero-bias data and simulation. The offset in data and simulation for 2018, separated by PF candidate, is presented in Figure 6.2.

6.2 Response corrections

Following pileup offset corrections, the jet simulated response corrections are computed and applied. This involves the determination and correction of the non-linear response of the calorimeters as a function of p_{T}, and several other detector effects leading to variations of the response in η. The detailed simulation of the CMS detector as described in Section 4.2 is used together with particle-level dijet events to provide accurate descriptions of the jet response (particle-level jets do not include energy from neutrino contributions). By using simulated samples as opposed to experimental data, one can ignore the bias inherent from data-based methods. Another advantage is that simulated data can access parts of phase space which are considered inaccessible, due to the low statistics, for experimental data. This includes low ($p_T < 30$ GeV) and high ($p_T > 1$ TeV) momenta jets, as well as jets with small ($\mu < 5$) and large ($\mu > 40$) average number of pileup interactions [9].

$$R_{ptcl}\left(\langle p_T \rangle, \eta\right) = \frac{\langle p_T^{reco} \rangle}{\langle p_T^{ptcl} \rangle} \tag{6.2}$$

The generated and reconstructed jets are matched spatially, on the basis that their distance is $R/2$ from each other. Here, R is the tunable jet distance parameter. This criteria results in 100% matching efficiency for jets

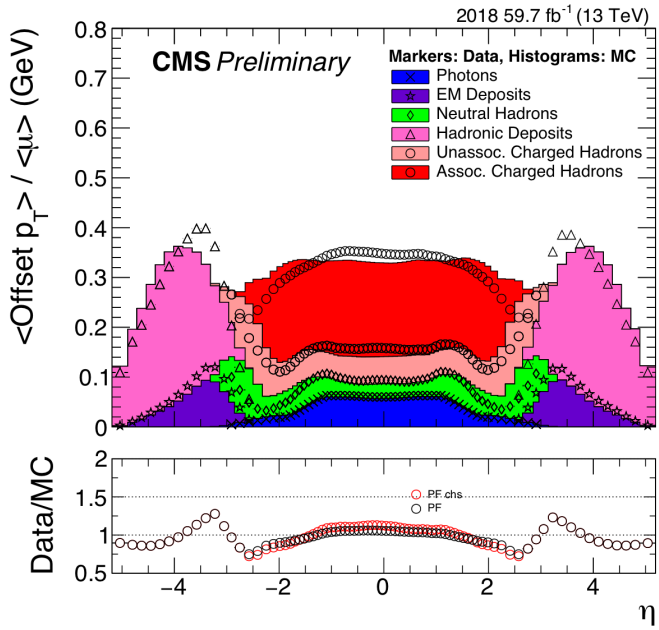


Figure 6.2: The RC offset measured in data (markers) and simulation (histograms) versus η is shown on the top. The offset is normalized by the average number of pileup interactions $\langle\mu\rangle$. Different types of PF candidates are represented. Associated charged hadrons are associated with reconstructed pileup vertices and thus removed from the list of PF candidates in the jet clustering by the CHS algorithm. In contrast, unassociated charged hadrons are not mitigated by CHS. At the bottom, the ratio of data over simulation, representing the offset scale factor applied for pileup in data, is also shown for PF and PFchs [30].

with p_{T} above around 30 GeV [9]. After matching the jets, the momentum response is computed with Equation 6.2, where $\langle p_T^{reco}\rangle$ is the mean p_{T} of the reconstructed jets in a given p_T^{ptcl} bin, and $\langle p_T^{ptcl}\rangle$ is the mean of the particle-level jet p_{T} in the same bin. Finally, the correction is determined by fitting a function to the inverse of the mean response as a function of $\langle p_T^{reco}\rangle$ in fine bins of η.

6.3 Residual corrections

For experimental data, after pileup offset and simulated response corrections, the remaining jet response discrepancy between experimental data and simulation is mitigated with a series of residual corrections. Residual corrections are determined by comparing the p_{T} of a jet to the p_{T} of a reference object, and computing the response. By utilising the principle of momentum conservation in the transverse plane, one can estimate the energy scale of a recoiling jet by using a reference object with a energy scale measured at high precision. Such an object can be a Z boson, a photon or another jet.

Residual corrections are divided into relative and absolute corrections. For the former, the correction is determined by extrapolating energy measurements from the barrel region ($|\eta| < 1.3$) to the endcap region ($1.3 < |\eta| < 3.0$). This is achieved by exploiting dijet events, which contain two jets with similar p_{T}, one in the barrel (tag jet) and one outside it (probe jet). Figure 6.3 shows an illustration of the dijet topology. The corrections include a p_{T} dependence of the JES relative to the JES of the barrel jet.

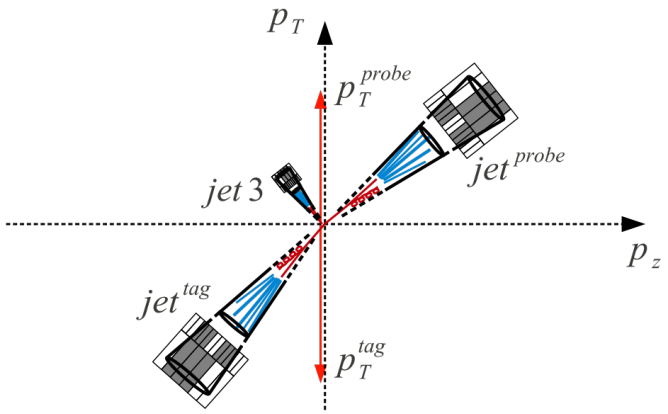


Figure 6.3: Illustration of dijet topology [30].

To correct the jet momenta for ranges of 30–700 GeV, the absolute residual corrections use three jet events ($Z \to e^+e^- + \text{jet}, Z \to \mu^-\mu^+ + \text{jet}$ and $\gamma + \text{jet}$). The precisely measured p_{T} of the Z boson or photon is used to derive a correction for the recoiling jet. The absolute jet p_{T} scale is determined with a χ^2 minimization.

By combining relative and absolute residual corrections, jets can be cali-

brated over a wide range of p_T and η values. As a result the precision of the JES is about 1% for central jets of 200 GeV [9].

6.4 Flavour corrections

Jets may be characterised on the basis of the flavour of the quark initiating the jet. Jets are assigned a flavour in a process known as *tagging*. In order to tag a jet, particle jets are matched to the hardest nearby b or c hadron for heavy flavours or to the hardest nearby generator-level parton for light flavours (u, d or s quark or gluon). This parton must be within $\Delta R < 0.25$, and be part of the hard scattering matrix element process [9]. Jets without a matching parton have undefined flavour.

The downstream detector response is dependent on the jet fragmentation. Therefore, additional corrections are derived from simulations by comparing the transverse momenta of jets of different flavours. Jets from u and d quarks have the highest response, while jets arising from gluons have the lowest due to high fragmentation into soft particles (which tend to fall below tracker and calorimeter p_T thresholds). c and b quarks are found between the two, as a result of the decay of heavy hadrons (e.g. B, Λ_b, D, Λ_c) into softer particles. QCD dijet samples are usually enriched in gluon jets, while Z + jet and γ+ jet are enriched in quark jets [9].

To determine the flavour corrections, the CMS uses an extension of the technique described in Section 6.3 for absolute residual corrections. Here, the jet events are Z+jet(b-tagged), photon+jet(quark and gluon-tagged) or Z+jet(quark and gluon-tagged). The reference object is easily identified and reconstructed with high precision. In addition, it is ideally back-to-back with the parton initiating the jet.

6.5 Systematic uncertainties

At CMS, any measurement of a physical quantity which includes the use of jets has to include an estimation of the jet energy calibration-induced uncertainty. In general, this is achieved by evaluating the measured quantity when the jet energy is fluctuated up and down according to the total jet energy uncertainty.

The JES uncertainties are provided as systematic sources that include correlations across p_T and η. At CMS the total uncertainty of the jet energy correction is computed as a quadratic sum of the uncertainty of each different source (of which there are currently 15). The uncertainties arise from the modeling of physics phenomena such as showers and underlying events, the modeling of the detector properties such as response and noise, and as potential biases in the methodologies used to estimate the corrections. Several of these uncertainties are related and can be combined into the following groups:

- Pileup offset.
- η-relative calibration of JES.
- p_T-relative calibration of JES.
- Jet flavour response.
- Time dependence.

Each of these are discussed in the following.

As seen in Figure 6.4, the pileup offset uncertainty is dominant at low p_T. The uncertainty in scale factors for the η dependence in data is corrected by a source from this group. However, the main systematic uncertainty lies in the offset jet p_T dependence, derived from simulation. It is about 1% for jet momenta equal to 30 GeV [9]. The η-dependent source is evaluated by varying the offset energy density within one standard deviation. The p_T-dependent source is created by estimating the difference between the particle-level offset and the RC offset. An additional uncertainty estimates the bias introduced by the data-based calibration for no-pileup conditions (this is optional and therefore not included by default in the quadratic sum).

For η-relative corrections the main uncertainties arise from the JER and the ISR+FSR bias corrections. The JER uncertainty is evaluated by varying the JER for each detector region independently, while applying *smearing* to the simulation. Smearing is used to introduce measurements errors. The uncertainty arising from the ISR+FSR, are estimates from the differences obtained by comparing Pythia and Herwig++ simulations.

The uncertainty in p_T-relative corrections is extracted from a global fit to Z/γ + jet and multijet events. The uncertainty is related to the lepton

momentum scale for muons in $Z(\rightarrow \mu\mu) + \text{jet}$ and the single-pion response in ECAL and HCAL. Additional uncertainty is added to correct for the bias arising from the exclusion of neutrinos and ISR outside of the detector acceptance. Shape uncertainties for the p_{T} dependence are additionally computed as the jet response differences due to the different fragmentation models implemented in Pythia and Herwig++.

Following pileup offset uncertainties, the flavour uncertainty is the highest at low p_{T} (about 1.5%). The flavor uncertainties are assigned based on Pythia and Herwig++ differences, which are propagated through a mockup of the data-based calibration chain with dijet and Z/γ+jet events. The uncertainty is computed from $uds/c/b$-quark and gluon responses [30]. The flavor differences in Pythia and Herwig++ are largest for gluon jets, while the two simulations agree well on both light-quark and heavy-flavour jets.

The JES in the endcaps shows some residual difference between the luminosity-weighted average of corrections per data-taking period per year. Prescaled triggers samples from different run periods with different weights leads to slight scale differences between different data sets. This time dependence uncertainty is estimated as the root-mean-squared variation of the η-dependent correction factors determined with dijet events, from approximately ten different data-taking periods [9].

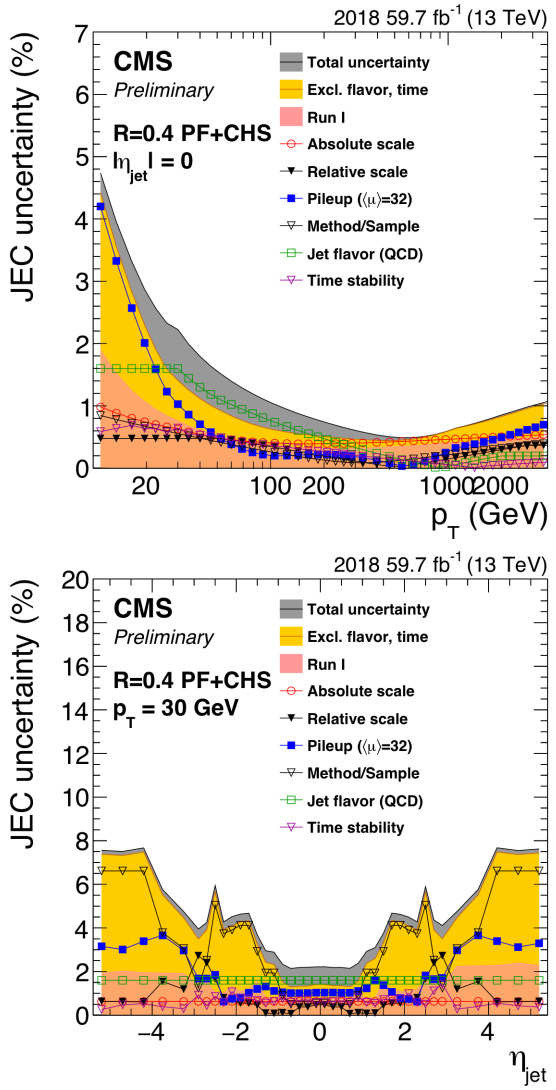


Figure 6.4: Jet energy uncertainty sources and total uncertainty (quadratic sum of individual uncertainties) as a function of jet p_{T} (top) and jet η (bottom). Run I uncertainty without flavour and time sources are shown for comparison [30].

Chapter 7

Analysis reproducibility and software practices

Previous chapters discussed how high energy physicists rely heavily on large data sets and as a result require a lot of computational power. Typically, each analysis group builds their analytical pipeline of computational steps by using scripting languages such as Bash [31] and Python [32]. These pipelines are often hard to maintain, lack portability and may be difficult to compare to other pipelines. A further problem is that the pipelines lack a description of either the environment or the software packages used. This reduces, or completely prevents, reproducibility of the analysis in a different context.

This approach uses the *imperative* paradigm. Here, the analyst is responsible for changing the state of the program by using explicit statements. In this thesis the benefits of an approach in which the description of the control flow (i.e. the order and execution of the individual statements) is removed from the analyst responsibility will be presented. This is referred to as the *declarative* paradigm.

Declarative rather than imperative programming enables the analyst to focus on the solving the task without having to spend time implementing details such as job orchestration. The analyst's focus is therefore on expressing *what* should be executed, without focusing on *how* to execute it. As a result, the computational tasks of the analysis are automated and the productivity of the analyst is improved. A further benefit of declarative programming is the separation of physics knowledge from specific program-

ming skills; high-level programming skills are no longer required to perform physics analysis, making it more accessible to a wider range of people [33].

This chapter will discuss the meaning of analysis reproducibility in high energy physics. The inherent challenges will be addressed, as will the steps steps that can be taken towards achieving reproducibility, such as embracing declarative programming. In addition, the software practices of the novel methodology described in this thesis will be considered. To successfully automate the computation of jet energy corrections, detailed information concerning both the computational *steps* and the computational *environment* is needed. The former is achieved with a computational workflow, and the latter using container technology. In this chapter, both will be discussed in detail.

7.1 Analysis reproducibility

The meaning of the term *reproducibility* is not constant across all scientific disciplines. In this thesis, the term is equivalent with "computational reproducibility". This means that the computational results should be achieved with a high degree of agreement when the study is reproduced with the same input data, computational steps, methods, code and conditions of analysis [34].

The typical stages in a high energy physics analysis are described in Figure 7.1. The stages include online data collection, offline computing processes and finally the statistical analysis. The online and offline data processing usually takes place in global grid computing infrastructures using automated scripts. In contrast, the physics analysis is performed by a range of analysts working individually using a variety of computational approaches. However, this step lacks standardisation, providing a barrier to effective reproducibility.

Challenges with reproducibility have been reported across many scientific disciplines [35], including high energy physics. These challenges are often attributable to a lack of a reliable communication strategy. For example, physicists at CERN have reported that communication practices often rely on engaging in personal communication with colleagues in order to find resources [36]. CERN physicists also report that meetings and presentations are an important means of disseminating knowledge. However, these forms of communication often do not leave a meaningful digital record of

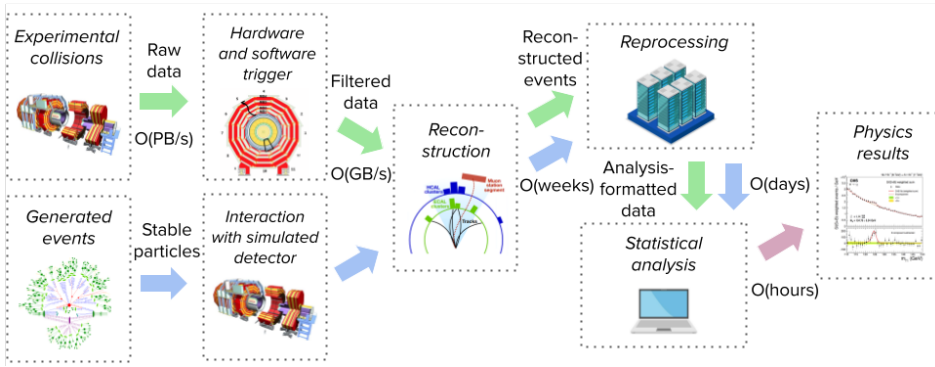


Figure 7.1: Typical stages in a high energy physics analysis. Following online data collection, the experimentally derived data is further processed offline (green arrows). To generate simulated events, the steps are mirrored by event generators and detector simulations (blue arrows). The analyst then compares the experimental and the simulated data against theoretical models using statistical analysis.

the methodological information conveyed. This makes it very difficult to physicists not physically present at the meeting to access this information. A second example stems from the inherent complexity (due to the sophisticated algorithms and the large amounts of data involved) of physics analyses. Although efforts to comprehensively document the precise steps of an analysis are sometimes made, the complexity of the analysis often hides important details. A failure to repeat the steps accurately will lead to different results being obtained. In both examples reproducibility is poor.

CERN is currently making efforts towards combating the issues surrounding reproducability. The CMS experiment is improving the preservation of analyses by using a reproducible analysis platform called REANA [37]. This platform allows researchers to express the computational data analysis steps using a declarative approach. The REANA platform reads the structured analysis description provided by the researcher and initiates analysis steps on containerised compute clouds. In this thesis, the computation of the jet energy corrections is designed to be executed on REANA.

7.2 Computational workflows

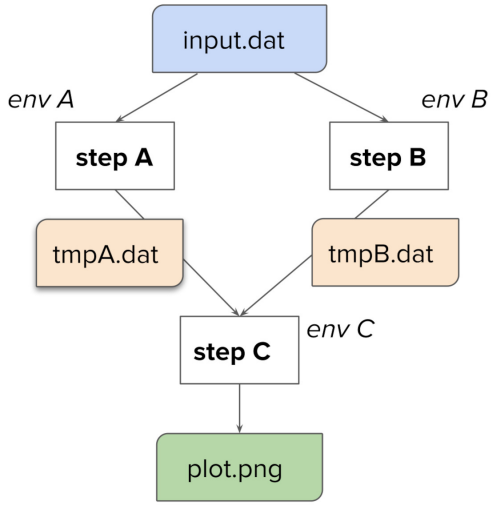
In recent years, a number of efforts have been made to address the challenges associated with sharing and reproducing computational analyses. One of the solutions has been to introduce *computational workflows*. A computational workflow uses a declarative approach to specify the steps of a process (e.g. a physics analysis) and their inter-connectivity. An independent "workflow system" then orchestrates the step execution to various deployment architectures. Computational workflows are highly portable and may be reliably executed in different settings, ranging from single host to high performance computing (HPC) clusters. In addition, computational workflows allow large-scale datasets to be processed, and can easily be amended to exploit parallel data processing across several nodes. The main benefits are portability and scalability, as well as ease of maintenance and comparison. In addition, software technologies such as Docker [38] may be leveraged, and most workflow systems monitor the executional progress and exit gracefully in the event of a failure at any given step.

The computational steps of an analysis workflow can be represented as a Directed Acyclic Graph (DAG). Each node represents a computational step, with input and outputs, and the edges describe the interconnection of the steps. A computational workflow allows you to express the DAG in a structured manner. An illustration of this is presented in Figure 7.2. The workflow is generally constructed using abstract formal language such as JSON [39] or YAML [40] formats. The order of job execution is determined automatically by the workflow engine.

In this thesis the workflow language Yadage [41] was used to capture the computational steps for computing the jet energy corrections. A schematic chart of the workflow is presented in Figure 7.3.

7.3 Parallelisation

A typical high energy physics data set is several terabytes in size and may be composed of several thousands of files. To speed up processing of the data and to avoid memory issues, the data is typically divided into independent jobs which run in parallel and do not communicate with each other. This type of data processing is referred to as being "embarrassingly parallel". A parameter known as "batchsize" describes the maximum number of data



step A:
- environment: env A
- inputs: input.dat
- outputs: tmpA.dat

step B:
- environment: env B
- inputs: input.dat
- outputs:tmpB.dat

step C:
- depends on: step A, step B
- environment: env C
- inputs: tmpA.dat, tmpB.dat
- output: plot.png

Figure 7.2: A simple computational workflow example expressed as a Directed Acyclic Graph (DAG). The left-hand panel shows a graph consisting of three nodes representing the computational steps A, B and C. The graph edges represent the analysis flow. The right-hand panel expresses the computational graph in an abstract formal language.

files in each job. The workflow language Yadage is particularly well-suited for running workflows which are highly parallelisable.

In general, parallel programming increases the complexity of the control flow. For example, coding errors may introduce a situation in which two jobs are dependent on each other and therefore are never executed. Another example is two jobs attempting to modify a shared resource at the same time. As a result, software development processes such as debugging and code maintenance increase in complexity.

A significant advantage of a declarative approach is the avoidance of these problems. Computational workflows remove the majority of the burden of code maintenance from the analyst by introducing implicit parallelism [33]. In the case of a physics analysis, the analyst expresses (in a declarative manner) *what* the analysis should be doing, without having to consider *how* the jobs are to be executed. The workflow engine automatically manages the dispatch and orchestration of the parallel jobs. If the program runs sequentially it will work equally well running in parallel. In either case, all that is required of the analyst is the correct input of the job description.

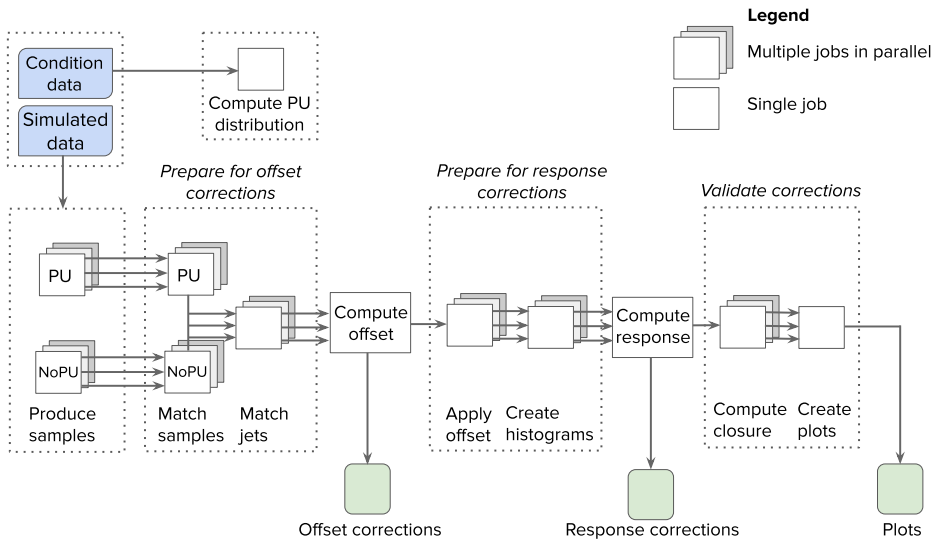


Figure 7.3: Schematic chart of the CMS offset and simulated response jet energy corrections workflow. The abbreviations "PU" and "NoPU" refers to with and without pileup respectively. The dotted lined boxes represents abstract high-level concepts, while the white squares represents jobs of the workflow. Three overlaid squares paired with three arrows signals a job which is scattered and ran in parallel. In contrast, as single square is a single job. Blue boxes signal input data while the green boxes represents final outputs.

A parallelisation scheme created by the Jet and Missing Energy (JME) group at CMS, was used to overcome the challenge of the large event sample when computing the jet energy corrections. First, events with and without pileup overlay are separated into files according to lumisections. A list containing the event, run number and lumi section for each file is then created. Next, a subset of with pileup overlay files are matched to the subset of without pileup overlay which contains the same information. This enables running over pairs of with and without pileup files in parallel, which is done up until computing the offset correction. Since no matching between with and without pileup events are necessary after this step, here after, separating the with pileup files simply means dividing the files into bunches of desired size. Parallelisation is used as much as possible, only merging the files when requiring all the statistics; before computing a correction or validating the result.

7.4 Container technology

The capture of the computational steps of an analysis is not in itself adequate for effective reproducibility. Computational analyses are designed and run in a specific computing environment. Reproducibility problems may arise as documentation of an environment is often insufficient to accurately replicate the environment for future analyses. As a solution, container technology may be used to capture the computational environment as well as necessary software packages.

Virtual machines (VM) have been used to simulate computing environments since the 1960s [42]. VMs are software which allows functional emulation of a specific physical computer on a different computer. Containers are lightweight counterparts to VMs, offering virtualization at the operating system level as opposed to the hardware level. Containers allow the packaging of an application, with its dependencies, so that it can be run in isolation from other processes. Container technology enables the analyst to work in a computing environment that has been frozen with respect to the libraries and conditions of the analysis. As a result, the analyst can use the same software that analysts were using several years ago without having to download all the relevant years-old libraries.

Two widely adopted open source container technologies are Docker and Singularity [43]. The former is used for development, testing, and deployment across various software enterprises, while the latter was specifically designed for HPC clusters.

Containers are powerful additions to computational workflows as they package the precise computational components required to execute each step. For instance, containers ensure that the step performing data selection has the necessary data selection libraries installed, while the step requiring specific CMS SoftWare (CMSSW) has access to the necessary release. Further information regarding containerizing CMSSW is provided in Appendix A. Another major benefit is the portability that containers provide. Analysts can download containers and recreate the analysis environment on their own system, data center or cloud provider.

In this thesis a lightweight Docker image only containing the base operating system (Centos7) was used. Cern VM File System (CVMFS) was mounted during run time to access CMSSW for compiling the code and for execution.

7.5 Continous integration and delivery

Computational workflows and containerization are practises which facilitates portable, shareable and reproducible analyses. Together with automated systems they allow the tracking of changes to the source code, input data and computing environment. As a result, such changes can be tested and if needed reversed automatically.

Analysts typically use a source code management system (also known as version control) such as GitHub [44] or GitLab [45] for developing their analysis code. Traditional sequential development methods are often used, where deployment follows the planning, developing and testing stages. Software engineering practises such as continuous integration (CI) and continuous delivery (CD) allow a different approach of code management, where all four stages cycle continuously with the software always being in a deliverable state. An illustrative comparison between traditional sequential development methods and CI/CD is presented in Figure 7.4.

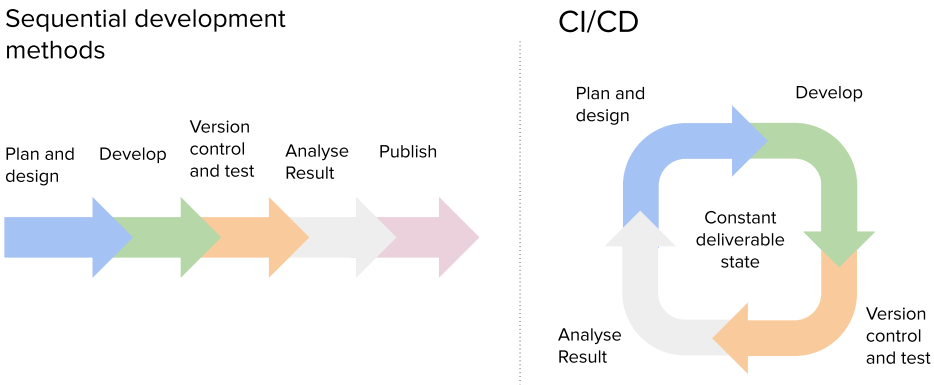


Figure 7.4: Schematic illustration comparing traditional sequential development methods and CI/CD. The main distinction is that CI/CD cycles all four stages (planning, developing, testing and deploying) continuously, allowing the software to be in a constant deliverable state.

CI commonly means that the developer continuously integrates their changes

with the main codebase. This is achieved by automating the compilation and test phases of the workflow. The system is configured to automatically perform the tasks after the integration of code changes [46]. In so doing, one may avoid challenges that can occur when when integration is delayed. CD is an extension of CI. CD ensures that the software is always ready to be released [47].

The main benefits of CI/CD are avoiding errors as the coding mistakes are identified early by the automated tests. This allows the developer to spend less time testing and solving issues, leaving more time for code improvement. In addition, CI allows the analyst to often and automatically examine how changes in the source code, input data and computational environment affects the output.

For these reasons, CI/CD have gained widespread popularity in the software industry. However, despite these benefits it has not yet been adopted by most physicists. The CMS experiment has a lifespan of decades, includes coding contributions from a large number of developers and is constantly being upgraded to achieve higher precision. Due to the lack of standardisation of the individual analyses within the wider CMS experiment, introducing CI/CD would greatly increase coding efficiencies, minimise the need for manual testing and maintain code quality.

REANA has developed a GitLab-REANA bridge that allows researchers to use GitLab as the source code management platform to develop their workflows, and REANA as a CI service to run them. The jet calibration workflow of this thesis was developed in hand with this integration.

Chapter 8

Results

In this chapter further details are presented about the simulated samples that were used when computing the jet energy corrections discussed in this thesis. The average value of the ratio of measured jet p_{T} to particle-level jet p_T^{ptcl} is referred to as the *response*. As a ratio close to one indicates a well calibrated jet, the response is used to validate the computed energy correction. The response of the computed corrections are presented, as are the benefits of automating the computation. The corrections are computed for jets with two different jet distance parameters (R=0.4 and R=0.8). There is also a consideration of the benefits of using a declarative, rather than imperative, paradigm.

8.1 Sample description

The main input to the workflow were two simulated QCD multijet samples (with and without pileup overlaid), each consisting of tens of millions of events. Both samples had a size of a few terabytes. The first step of the preparatory phase consists of matching particle-level jets to PF jets. In this step, a preliminary filter is applied to the data in order to extract only information which is useful for downstream analysis. This speeds up the analysis. This step is typically split into several hundred parallel computing jobs, each lasting several hours, and is performed with specific tools implemented in CMSSW. The output of this preparatory phase is an analysis formatted sample which has a reduced size of a few hundred gigabytes. This step is summarised in the two first rows of Table 8.1.

Sample	Byte size	Production time
Simulated QCD samples	$\mathcal{O}$(10 TB)	$\mathcal{O}$(1 week)
Analysis formatted samples	$\mathcal{O}$(100 GB)	$\mathcal{O}$(1 day)
Combined with/without pileup	$\mathcal{O}$(1 GB)	$\mathcal{O}$(1 hour)
Control distributions	$\mathcal{O}$(1 MB)	$\mathcal{O}$(1 min)

Table 8.1: Sequential summary of the byte size and production time of the analysis samples.

The samples then enter the jet energy calibration workflow described in this thesis. The corresponding jets from the samples with and without pileup are combined to calculate the offset and resulting corrections. If we exclude the creation of the QCD samples and the subsequent analysis formatted sample (they are typically only run once in the beginning and are not part of the jet energy calibration workflow captured in this thesis), this creation of the combined sample is the most time-consuming part of the workflow. The combined data set is reduced to a few gigabytes in size and the files containing the subsequent corrections and control distributions are of the order of megabytes. The data set sizes for the derivation of the simulated response corrections are similar.

8.2 Correction computation with R=0.4

Here, the result from computing the pileup offset and response corrections for jets, with jet distance parameter R=0.4, is illustrated. In Figure 8.1, the offset-corrected response (as a function of jet $|\eta|$) before and after applying response corrections are presented. The simulated jet response for jets ranging from low to high p_{T} is shown in Figure 8.1 (left). The response is stable for the majority of the barrel region, maintaining a value between 0.90 and 0.95. At $|\eta| > 1$ the response decreases, with a noticeable dip in the transition between the barrel and endcap regions. After a slight upturn, the response decreases throughout the endcap region, reaching a low point when entering EC2 (with the exception of jets with $p_{\mathrm{T}} \geq 400$ GeV).

In the endcap and forward regions, the responses for jets of different p_{T} diverge. The general response then increases in the forward region, but a distinction between the jets with different p_{T} remains. The response is lower for low p_{T} jets, in particular in the region around $|\eta| = 3.0$ where it decreases to 0.59 and 0.65 for jets with p_{T} equal to 30 GeV and 60 GeV respectively.

The response, after applying the pileup offset and response corrections, as a function of reconstructed η, is shown in Figure 8.1 (right). The response has improved accuracy, and is within ± 0.01 in the barrel region. The response decreases around $|\eta| = 2$, but then improves and returns to ± 0.01 for jets with $3.5 < |\eta| < 5$.

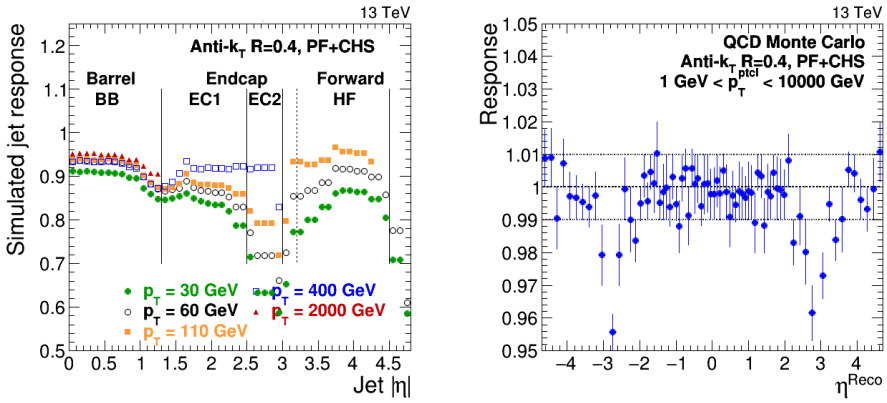


Figure 8.1: The left-hand panel shows the simulated jet response versus jet $|\eta|$. The simulated response is shown for different values of jet p_{T}. On the right is the response after applying the pileup offset and response corrections, versus reconstructed η. The response is shown for particle-level jets with energies between 1-10000 GeV. The jets are clustered from simulated QCD samples using the anti-k_t algorithm. The jets are reconstructed with jet distance parameter R=0.4 from PF candidates having been subjected to CHS.

In Figure 8.2, the response as a function of the particle-level jet p_T^{ptcl} is shown (1) before any corrections, (2) after correcting for pileup offset and (3) after correcting for pileup offset and the non-linear detector response. To illustrate the dependence of the response on jet η, distributions corresponding to different values of η are shown separately. Before any corrections are applied, the response is in general higher than 1 for jets with $p_{\mathrm{T}} < 100$ GeV excluding jets with $1.3 < |\eta| < 2.5$ in which the limit is set at

about 30 GeV. After correcting for pileup offset, jets with $2.5 < |\eta| < 3.0$ have a noticeably lower response. The response for these jets reaches 0.56 as the p_T decreases. After correcting for both pileup offset and the non-linear detector response, the jet response is stable for jets with p_T in the ranges from 30 to 2000 GeV. For jets with $p_T < 30$ GeV, excluding jets with $2.5 < |\eta| < 3.0$, the response stays within ± 0.05. For jets with $2.5 < |\eta| < 3.0$ the response decreases as the jet p_T decreases, finally reaching 0.79.

8.3 Correction computation with R=0.8

Here, the results from computing the pileup offset and response corrections, for jets with jet distance parameter R=0.8, are illustrated. The simulated jet response for jets ranging from low to high p_T (Figure 8.3, left) will be considered first. As with R=0.4, the detector response is stable for the majority of the barrel region, remaining between 0.90 and 0.95 for jets with $p_T \geq 60$ GeV. Similarly to R=0.4, in the endcap and forward regions the response diverges for jets with different p_T. However, for R=0.8, there is already a noticeable divergence in the barrel region for jets with low p_T. Again, there is a slight dip in the transition between the barrel and endcap regions. Similarly to R=0.4, the response in generally lower for low p_T jets. The response in the forward region follows a similar curve as for R=0.4.

The response, after applying the pileup offset and response corrections (Figure 8.3, right), is also similar to that of R=0.4. The response is within ± 0.01 in the barrel region. However in contrast to R=0.4, the response remains within that limit for longer, up to $|\eta| < 4$. Additionally, the barrel and endcap regions are characterised by reduced response variation.

The response at different stages of the workflow (Figure 8.4) are now considered. Before applying the corrections the response is higher than 1 for jets with $p_T < 100$ GeV. This is similar to the results for R=0.4, but also includes jets with $1.3 < |\eta| < 2.5$). Similarly to R=0.4, after computing and correcting for pileup offset, jets within $2.5 < |\eta| < 3.0$ have the lowest response (reaching 0.60 at p_T equal to 0 GeV). After correcting for both pileup offset and the non-linear detector response, the jet response is stable for jets with p_T in the ranges $50 - 2000$ GeV. For jets with $p_T < 50$ GeV the response decreases with p_T, decreasing to 0.85 for jets in the barrel and endcap region, and 0.57 for jets with $2.5 < |\eta| < 3.0$.

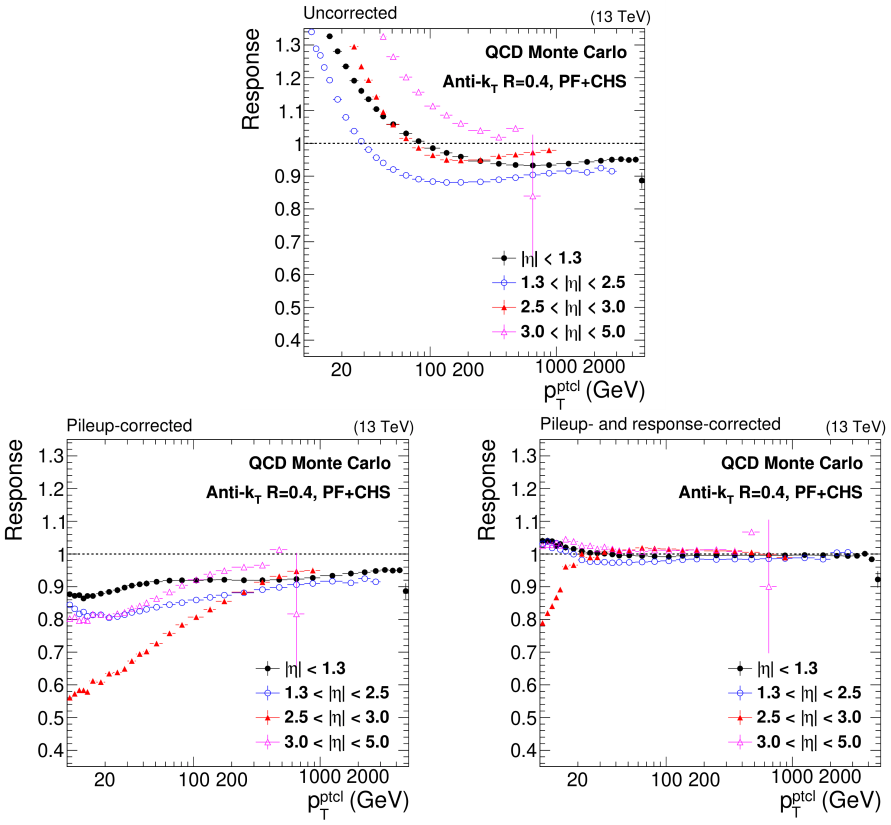


Figure 8.2: These graphs show the response, with statistical uncertainties, at various stages of computing the jet energy corrections. The response is measured in bins of p_T^{ptcl} before any corrections (top), after correcting for pileup offset (bottom left) and after correcting for pileup offset and the non-linear detector response (bottom right). The response is shown for different values of $|\eta|$. The jets are clustered from simulated QCD samples using the anti-k_t algorithm. The jets are reconstructed with jet distance parameter 0.4 from PF candidates having been subjected to CHS.

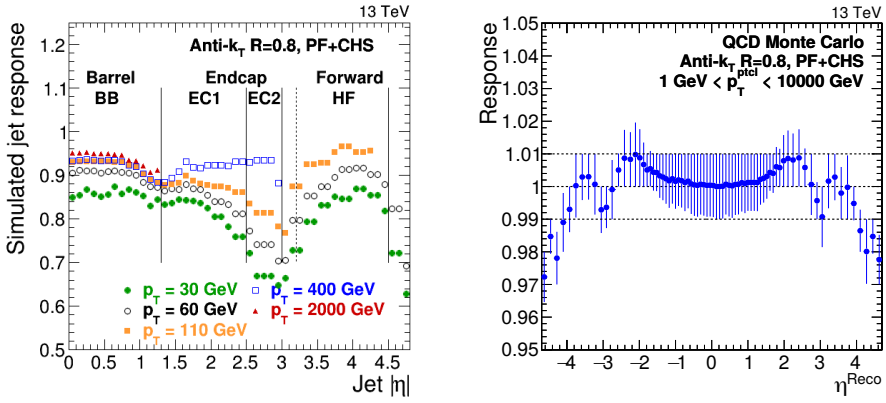


Figure 8.3: The left-hand panel shows the simulated jet response versus jet $|\eta|$. The simulated response is shown for different values of jet p_{T}. On the right is the response after applying the pileup offset and response corrections, versus reconstructed η. The response is shown for particle-level jets with energies between 1-10000 GeV. The jets are clustered from simulated QCD samples using the anti-k_t algorithm. The jets are reconstructed with jet distance parameter 0.8 from PF candidates having been subjected to CHS.

8.4 Automation and the declarative approach

It was found that the jet energy calibration workflow could be successfully automated using the methodology described in this thesis. The benefits associated with this automation are now considered.

Workflow automation allows the monitoring of simulated sample generation, as well as automatic computation of the corrections (independent of analyst input). The workflow can be started as soon as the input data sets are available, and runs to completion without the need for further user input. In addition, the application of the declarative paradigm to the workflow greatly reduces the time needed to obtain the results. As a result the running time of the workflow decreases from two days to approximately two hours. This is achieved in part by removing responsibility for code execution from the analyst. This enhanced autonomy, coupled with a reduced running time, means that first pass calibration results are available almost immediately.

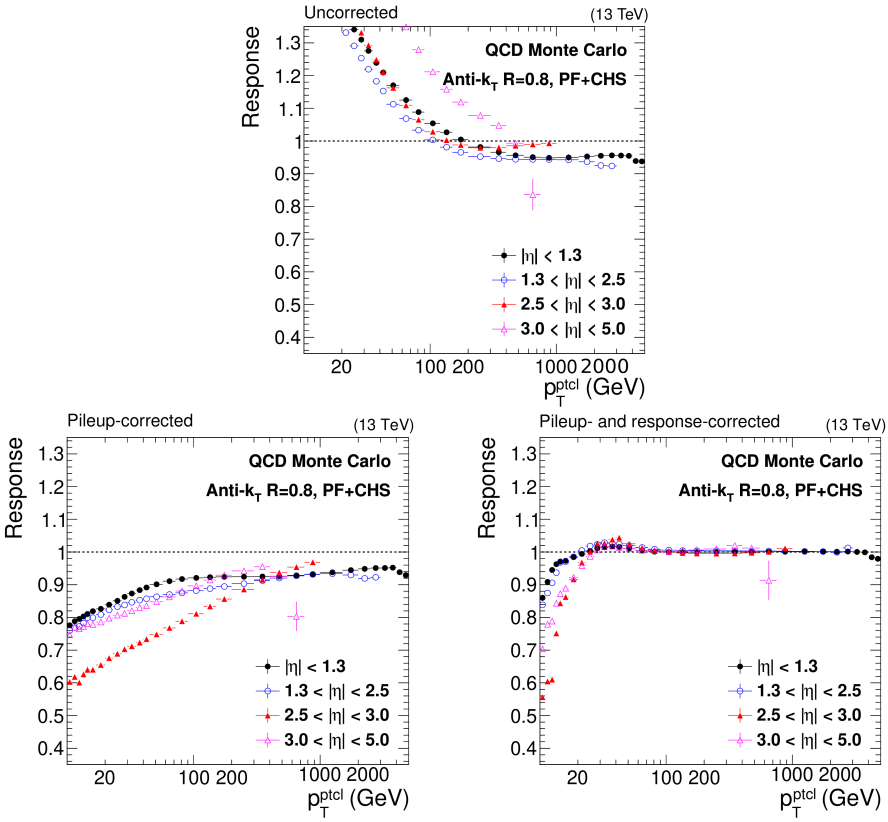


Figure 8.4: The response (average value of the ratio of measured jet p_{T} to particle-level jet p_T^{ptcl}), with statistical uncertainties, at various stages of computing the jet energy corrections. The response is measured in bins of p_T^{ptcl} before any corrections (top), after correcting for pileup offset (bottom left) and after correcting for pileup offset and the non-linear detector response (bottom right). The response is shown for different values of $|\eta|$. The jets are clustered from simulated QCD samples using the anti-k_t algorithm. The jets are reconstructed with jet distance parameter 0.8 from PF candidates having been subjected to CHS.

Another result is the increased ease of code maintenance. The computational steps are saved with the environment in one place under version control. Source code changes, changes to the job description and changes to the computational environment are all automatically saved and versioned between runs. This brings improved reproducibility. With the addition of CI, the analyst is able to test changes at any stage. This allows, for example, the analyst to take into account an improved parameterisation. The restarted workflows are versioned and do not overwrite previous iterations, thereby conserving the full provenance of results and facilitating comparisons between previous runs.

However, it is worth noting that this method is not without its challenges. As with all physics analyses, the analyst typically rerun workflows many times with altered conditions, either to fix a problem or to study the results with altered parameters. With this new automated approach, manual intervention is still required to do this.

Chapter 9

Discussion

The results of the thesis are discussed in detail. Firstly, the similarities and differences between the computed jet response for jet distance parameter R=0.4 and R=0.8 will be considered. Secondly, the advantages to automating the jet energy calibration workflow will be presented. This will include the benefits of shifting from the imperative to the declarative paradigm. Finally, potential sociological and technological challenges to the widespread adoption of automated declarative analyses will be discussed.

9.1 Comparing corrections with R=0.4 and R=0.8

Examining the response figures for R=0.4 and R=0.8, it is clear that the general structure for both cases follows the same curve. As seen in Figure 8.1 (left) and 8.3 (left), in the barrel and endcap regions, there is a general trend of decreasing jet response as $|\eta|$ increases. With increasing $|\eta|$, the distance which the particles have to travel through the tracker, and correspondingly the amount of material traversed, increases. As a result, at higher $|\eta|$, there is a greater chance of particles decaying before depositing their energy in the calorimeters, leading to a decrease in the response. The detector performance is particularly poor in the transition regions. The reason is again the larger amount of material in front of the calorimeters, and the correspondingly high rates of nuclear interactions in the tracker. As the EC2 region ($2.5 < |\eta| < 3.0$) is outside tracker coverage, the response decreases due to a lack of tracks to be used by event reconstruction.

Although the pattern for the jet response when R=0.8 is broadly similar to that of R=0.4, there are a few differences suggesting a dependence between the jet response and the jet distance parameter R. Jets with larger distance parameter tend to contain a larger fraction of low-energy particles. Low-energy particles have a lower response compared to particles from hard scatter, lowering the response for low p_{T} jets with large R. This can be seen in Figure 8.1 (left) and 8.3 (left). Comparing jets of 30 GeV in both figures, the response for jets with R=0.8 is about 0.85 throughout the barrel region, while jets with R=0.4 is at 0.9. A similar, but less severe, effect can be seen for jets of 60 GeV. In contrast, the high p_{T} jets are not affected by the jet distance parameter, staying at 0.95 in both cases.

Before correcting, the measured jet p_{T} is greater than the particle-level jet p_T^{ptcl} for low p_{T} jets with both R=0.4 and R=0.8. This can be seen in the top of Figure 8.2 and 8.4. This is largely due to low energy pileup. When comparing the top and bottom left figures of Figure 8.2 and 8.4 the response after applying pileup-offset corrections give the biggest improvements in accuracy for jets with $p_{\mathrm{T}} < 100$ GeV (R=0.4) and $p_{\mathrm{T}} < 400$ GeV (R=0.8). The observed difference between jets of different R is again explained by the increased amount of particles from pile-up covered by the larger jet area.

The larger the R, the larger the area in (η, ϕ)-plane that is covered by the jet. As an effect, larger jet distance parameters average the jet response over a larger area. As a result, sharp features in the detector response versus η are smeared. This can be seen in Figure 8.1 (right) and 8.3 (right). Comparing the two figures, the response for jets with R=0.4 appear more spread out, while the response for jets with R=0.8 follow a smooth curve.

9.2 Benefits of automation and the declarative approach

The results obtained from the automation of the jet energy calibration workflow strongly suggests that capturing complex experimental particle physics analyses with declarative workflow languages is both achievable and advantageous. The main benefits are that the workflow is quick, portable, scalable, and easy to maintain and compare to previous versions.

The reduced running time was expected, and may be explain by two fac-

tors. Firstly, by automating the code execution, the workflow can be performed independently of the analyst. This speeds the analysis as it can progress without the need to wait for analyst input, which may be delayed if the analyst is working on another task or it is out of working hours. The running time is also enhanced because of improved resource availability. In the established approach, the analyst dispatches jobs to computing servers through HTCondor. The HTCondor system is widely used, leading to queueing times which are often significant, slowing the analysis. In contrast, the jet energy calibration workflow described in the thesis runs in the REANA cluster. At the time of writing (February 2021) the REANA cluster has 10 nodes and 80 cores available. The traffic on REANA is relatively low, greatly reducing the waiting time before the analysis can begin and increasing the analysis speed. On REANA, the workflow runs with an average of two hours. Secondly, it is also possible to introduce a further automated step to initiate the analysis pipeline. By monitoring the generation of the simulated samples (e.g. with a cron job) the correction computation may be automatically started and ran to completion without analyst input. This removes any lag-time between simulated sample generation and beginning of the analysis.

The resulting reduced run time provides quick feedback on the data set quality to the CMS collaboration, enhancing the efficiency of the experiment.

The declarative approach further reduces workload of the analyst. Changes to the job orchestration is eased by the help of the computational workflow. To give one example, with the computational workflow of this thesis, changing the input only requires changing the path at one place in the script. This is true even when the number of files changes. In contrast, with an imperative approach, changing the number of files might affect the interconnection between downstream jobs, requiring more code handling. The reduced workload frees up time for the analyst. Instead of repeatedly running the largely technical and tedious parts of work, the analyst's time can be allocated to improving the analysis.

As discussed in Section 7.1, to help ensure the reproducibility of computational results, clear and complete information about any computational methods, data and conditions that support the published results should be shared. In our case, reproducibility is facilitated by keeping the full description of these in one place. The computational steps and associated parameters are stored and versioned by storing the computational workflow

under version control. The computational environment where the analysis was originally executed, such as operating system and library dependencies are saved with the use of container technology. In addition to reproducing the final result, any intermediate results and outputs are guaranteed with the complete analysis pipeline captured by the computational workflow.

9.3 Challenges of automation and the declarative approach

The current common practice of experimental particle physicists is centered around the imperative approach rather than declarative. While there are clear advantages to the new declarative approach, there may be a few obstacles to its widespread adoption. Despite requiring less extensive programming skills compared to the established approach, declarative programming does require the analyst to think differently about the problem at hand. The use of declarative programming necessitates more than simply learning a new software tool, it requires a new way to structure the whole analysis. This may lead to some initial resistance from analysts. However, once the extensive benefits of declarative programming are understood, we anticipate the analysts to buy in to this approach.

Additional challenges foreseen by us are more technological in nature. As described in Chapter 8, manual intervention is still necessary when rerunning workflows. This is common practice when considering the effects of different conditions. The non-automated and imperative nature of the established approach allows the process to be restarted partway through, without the need to return to the beginning. This is desirable as the initial steps are often the most time-consuming. This is because the data sets must first be filtered to only contain information relevant for the downstream analysis and converted to a simplified data format, which enables faster analysis. In contrast, the automated declarative approach requires the entire process to be repeated to obtain new results. The duplicative and redundant rerunning of analyses represents an inefficiency of the current state of the automated declarative approach. It is possible to manually intervene to avoid the need to return to the start of the analysis. However, this process is more difficult than it is in the imperative approach. Future development addressing this issue will facilitate wide-spread adoption of this methodology.

Chapter 10

Conclusion

This thesis describes the successful automation of the CMS jet energy corrections (derived from simulation) computations. This was achieved using the declarative paradigm, as opposed to the more commonly used imperative paradigm. While the widespread adoption of this new approach must overcome anticipated challenges of a sociological and technological nature, these are considered surmountable in light of the extensive benefits.

The workflow description and containerisation allowed preservation of the computational tasks of the analysis together with the full description of the job orchestration and the execution environment. Accompanying source code management facilitated easier code maintenance, enhanced analyst productivity and simple comparisons between runs. Additionally, the wider declarative approach brought the benefits of reduced analysis running time and increased analysis portability and scalability. Importantly, higher levels of reproducibility were facilitated. In conclusion, automation of the jet calibration has the potential to enhance productivity and free the analyst to spend more time improving the corrections. This can lead to a better understanding of the JES and its uncertainties, facilitating more precise downstream analysis measurements and improving the CMS experiment as a whole.

Appendices

Appendix A

CMSSW Docker image building service

The CERN Open Data portal uses virtual machines (VMs) to provide access to CMS-specific software and conditions. VMs with CMS SoftWare (CMSSW) facilitate execution reproducibility on machines without access to the framework. VMs are often used for educational purposes, for example by teachers and students at workshop events. As container technology has become more popular, there has been a movement towards system-agnostic containerized versions of CMSSW. Containers are more light-weight than VMs and therefore require less allocated space and download time. The main advantage of containerized versions of CMSSW mirrors that of VMs; they provide an interface facilitating access to computing tools for working with CMS data. Additional benefits include allowing:

- Teachers and students to avoid the need to spend hours setting up the necessary software, often for use in a short lesson.
- Easy management of the environment and dependencies when running on the batch system.
- Developing of analysis in a self-contained environment.
- Offline development.
- Access to CERN VM File System (CVMFS).
- Shared common setup between developers of larger projects.

- Analysis preservation and reproducibility.

We created a platform for building Docker containers, with installed CMSSW releases, on demand. The *CMSSW Docker image building service* (`cmssw-docker.web.cern.ch/`) is a part of the HEPContainers project. The goal of this project is to converge all HEP containers into one place, making it easy for physicists in the HEP community to use and distribute containers created specifically for their kind of work.

The platform has a frontend created in REACT and backend coded in Golang. The platform avoids the need for a database by utilising GitLab. When a request is made in the frontend by the user, a HTTP request to the backend is made. The backend utilises the GitLab API to create an Gitlab *issue*, storing the necessary information. This include, the CMSSW release, SCRAM ARCH version, reason for request and email of the requester (if entered). The issue (request) stays open until approved. The administrators of the site approves the request by interacting with the issue (more exactly by adding the label "Approved"). A webhook, triggered by the event, then provides the backend with the information of the requested image. The information is parsed and the GitLab API is used to start a pipeline on the GitLab CI. A tool known as kaniko is responsible for building the container images. Building of the image may take few hours until completion. When the status of the pipeline changes, a HTTP request is made to the backend via another webhook. If the build is successful the requester is provided with an email for how to access the requested Docker image. In case of issues, the administrators are warned.

Users navigate through requests via the web page created by the administrators. On the other hand, the administrators manages all requests through the exploitation of GitLab.

References

[1] M. D. Schwartz, "Quantum field theory and the standard model". Cambridge University Press, 2014.

[2] S. Weinberg, "The making of the Standard Model", *The European Physical Journal C-Particles and Fields* **34** (2004), no. 1, 5–13.

[3] W. Commons, "File:standard model of elementary particles.svg — wikimedia commons, the free media repository", 2021. Accessed: 02-02-2021.

[4] W. Commons, "File:elementary particle interactions.svg — wikimedia commons, the free media repository", 2020. Accessed 02-02-2021.

[5] M. H. Seymour, "Jets in QCD", *AIP Conf. Proc.* **357** (1995) 568–587. 20 p, doi:10.1063/1.49625, https://cds.cern.ch/record/283896.

[6] P. Soding, B. Wiik, G. Wolf, and S. L. Wu, "The First evidence for three jet events in e^+ e^- collisions at PETRA: First direct observation of the gluon", in *International Europhysics Conference on High-energy Physics (HEP 95)*, pp. 3–14. 1996. doi:10.1016/0370-2693(80)90639-5, https://doi.org/10.1016/0370-2693(80)90639-5.

[7] R. Atkin, "Review of jet reconstruction algorithms", *Journal of Physics: Conference Series* **645** (2015) 012008, doi:10.1088/1742-6596/645/1/012008, https://doi.org/10.1088/1742-6596/645/1/012008.

[8] M. Cacciari, G. P. Salam, and G. Soyez, "The anti-kt jet clustering algorithm", *Journal of High Energy Physics* **2008** (2008) 063–063, doi:10.1088/1126-6708/2008/04/063, http://dx.doi.org/10.1088/1126-6708/2008/04/063.

[9] CMS Collaboration, "Jet energy scale and resolution in the CMS experiment in pp collisions at 8 TeV", *Journal of Instrumentation* **12** (2017) P02014–P02014, doi:10.1088/1748-0221/12/02/p02014, http://dx.doi.org/10.1088/1748-0221/12/02/P02014.

[10] T. Sjöstrand et al., "An introduction to Pythia 8.2", *Computer Physics Communications* **191** (2015) 159 – 177, doi:https://doi.org/10.1016/j.cpc.2015.01.024, http://www.sciencedirect.com/science/article/pii/S0010465515000442.

[11] M. Bähr et al., "Herwig++ physics and manual", *The European Physical Journal C* **58** (2008) 639–707, doi:10.1140/epjc/s10052-008-0798-9, http://dx.doi.org/10.1140/epjc/s10052-008-0798-9.

[12] P. Nason, "A new method for combining NLO QCD with shower Monte Carlo algorithms", *Journal of High Energy Physics* **2004** (2004) 040–040, doi:10.1088/1126-6708/2004/11/040, http://dx.doi.org/10.1088/1126-6708/2004/11/040.

[13] S. Alioli, P. Nason, C. Oleari, and E. Re, "A general framework for implementing NLO calculations in shower Monte Carlo programs: the POWHEG BOX", *Journal of High Energy Physics* **2010** (2010) doi:10.1007/jhep06(2010)043, http://dx.doi.org/10.1007/JHEP06(2010)043.

[14] S. Höche, "Introduction to parton-shower event generators", in *Journeys Through the Precision Frontier: Amplitudes for Colliders: TASI 2014 Proceedings of the 2014 Theoretical Advanced Study Institute in Elementary Particle Physics*, pp. 235–295. World Scientific, 2016.

[15] Z. Nagy and D. E. Soper, "What is a parton shower?", *Physics Review D* **98** (2018) 014034, doi:10.1103/PhysRevD.98.014034, https://link.aps.org/doi/10.1103/PhysRevD.98.014034.

[16] T. D. Gottschalk, "Backwards evolved initial state parton showers", *Nuclear Physics B* **277** (1986) 700 – 738, doi:https://doi.org/10.1016/0550-3213(86)90465-7, http://www.sciencedirect.com/science/article/pii/0550321386904657.

[17] Particle Data Group Collaboration, "Review of particle physics", *Physics Review D* **98** (2018) 030001,

doi:10.1103/PhysRevD.98.030001,
https://link.aps.org/doi/10.1103/PhysRevD.98.030001.

[18] A. M. Sirunyan et al., "Extraction and validation of a new set of CMS Pythia 8 tunes from underlying-event measurements", *The European Physical Journal C* **80** (2020) doi:10.1140/epjc/s10052-019-7499-4, http://dx.doi.org/10.1140/epjc/s10052-019-7499-4.

[19] S. Agostinelli et al., "Geant4—a simulation toolkit", *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **506** (2003), no. 3, 250 – 303, doi:https://doi.org/10.1016/S0168-9002(03)01368-8, http://www.sciencedirect.com/science/article/pii/S0168900203013688.

[20] CMS Collaboration, "The performance of the CMS Muon system with data at $\sqrt{s} = 13$ TeV", Technical Report CMS-CR-2018-167, CERN, Geneva, 2018. doi:10.1063/1.5091214, http://cds.cern.ch/record/2668218.

[21] CMS Collaboration, "The CMS simulation software", in *2006 IEEE Nuclear Science Symposium Conference Record*, volume 3, pp. 1655–1659, IEEE. 2006.

[22] CMS Collaboration, "Pileup mitigation at CMS in 13 TeV data", *Journal of Instrumentation* **15** (2020) P09018–P09018, doi:10.1088/1748-0221/15/09/p09018, http://dx.doi.org/10.1088/1748-0221/15/09/P09018.

[23] CMS Collaboration, "CMS Physics: Technical Design Report Volume 1: Detector Performance and Software", CERN-LHCC-2006-001, 2006 https://cds.cern.ch/record/922757.

[24] CMS Collaboration, "Particle-flow reconstruction and global event description with the CMS detector", *Journal of Instrumentation* **12** (2017) P10003–P10003, doi:10.1088/1748-0221/12/10/p10003, https://doi.org/10.1088/1748-0221/12/10/p10003.

[25] S. Laurila, "Search for Charged Higgs Bosons Decaying to a Tau Lepton and a Neutrino with the CMS Experiment". PhD thesis, University of Helsinki, 2019. https://helda.helsinki.fi/handle/10138/305652.

[26] G. Petrucciani, A. Rizzi, and C. Vuosalo, "Mini-AOD: A new analysis data format for CMS", *Journal of Physics: Conference Series* **664** (2015) 072052, doi:10.1088/1742-6596/664/7/072052, https://doi.org/10.1088/1742-6596/664/7/072052.

[27] ALEPH Collaboration, "Performance of the ALEPH detector at LEP", *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **360** (1995), no. 3, 481 – 506, doi:https://doi.org/10.1016/0168-9002(95)00138-7, http://www.sciencedirect.com/science/article/pii/0168900295001387.

[28] CMS Collaboration, "Description and performance of track and primary-vertex reconstruction with the CMS tracker", *Journal of Instrumentation* **9** (2014) P10009–P10009, doi:10.1088/1748-0221/9/10/p10009, https://doi.org/10.1088/1748-0221/9/10/p10009.

[29] CMS Collaboration, "Performance of the CMS muon detector and muon reconstruction with proton-proton collisions at $\sqrt{s} = 13$ TeV", *Journal of Instrumentation* **13** (2018) P06015–P06015, doi:10.1088/1748-0221/13/06/p06015, http://dx.doi.org/10.1088/1748-0221/13/06/P06015.

[30] CMS Collaboration, "Jet energy scale and resolution performance with 13 TeV data collected by CMS in 2016-2018", Detector Performance Summary CMS-DP-2020-019, CERN, 2020. https://cds.cern.ch/record/2715872.

[31] P. GNU, "Free software foundation. bash (3.2. 48)[unix shell program]", 2007.

[32] G. Van Rossum and F. L. Drake Jr, "Python reference manual". Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[33] J. W. Lloyd, "Practical advantages of declarative programming", in *Joint Conference on Declarative Programming*, pp. 3–17. 1994.

[34] N. A. of Sciences Engineering and Medicine, "Reproducibility and Replicability in Science". The National Academies Press, Washington, DC, 2019. doi:10.17226/25303, ISBN 978-0-309-48616-3.

[35] M. Baker, "1,500 scientists lift the lid on reproducibility", *Nature News* **533** (2016), no. 7604, 452.

[36] S. S. Feger, S. Dallmeier-Tiessen, A. Schmidt, and P. W. Woundefinedniak, "Designing for reproducibility: A qualitative study of challenges and opportunities in high energy physics", in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, p. 1–14. Association for Computing Machinery, New York, NY, USA, 2019. doi:10.1145/3290605.3300685, https://doi.org/10.1145/3290605.3300685.

[37] T. Šimko et al., "REANA: A System for Reusable Research Data Analyses", *EPJ Web of Conferences* **214** (2019) 06034, doi:10.1051/epjconf/201921406034.

[38] D. Merkel, "Docker: lightweight Linux containers for consistent development and deployment", *Linux journal* **2014** (2014), no. 239, 2.

[39] F. Pezoa et al., "Foundations of JSON schema", in *Proceedings of the 25th International Conference on World Wide Web*, pp. 263–273, International World Wide Web Conferences Steering Committee. 2016.

[40] O. Ben-Kiki, C. Evans, and I. döt Net, "Yaml ain't markup language version 1.2", 2009.

[41] K. Cranmer and L. Heinrich, "Yadage and Packtivity – analysis preservation using parametrized workflows", *Journal of Physics: Conference Series* **898** (2017) 102019, doi:10.1088/1742-6596/898/10/102019, http://dx.doi.org/10.1088/1742-6596/898/10/102019.

[42] J. Smith and R. Nair, "The architecture of Virtual Machines", *Computer* **38** (2005) 32–38.

[43] G. M. Kurtzer, V. Sochat, and M. W. Bauer, "Singularity: Scientific containers for mobility of compute", *PLOS One* **12** (2017), no. 5, e0177459.

[44] Github incorporated, "GitHub". https://github.com/. Accessed: 23-02-2021.

[45] Gitlab incorporated, "GitLab". https://gitlab.com/. Accessed: 23-02-2021.

[46] M. Fowler, "Continuous Integration". https://martinfowler.com/articles/continuousIntegration.html. Accessed: 28-01-2021.

[47] M. Fowler, "Continuous Delivery". https://martinfowler.com/bliki/ContinuousDelivery.html. Accessed: 28-01-2021.