ETP-KA/2021-6

Dynamic Provision of Heterogeneous Computing Resources for Computation- and Data-intensive Particle Physics Analyses

Zur Erlangung des akademischen Grades eines

DOKTORS DER NATURWISSENSCHAFTEN
(Dr. rer. nat.)

von der KIT-Fakultät für Physik des
Karlsruher Instituts für Technologie (KIT)

genehmigte

DISSERTATION

von

M.Sc. Matthias Jochen Schnepf

aus Bad Friedrichshall

Tag der mündlichen Prüfung:	16. April 2021
Referent:	Prof. Dr. Günter Quast
Korreferent:	Prof. Dr. Achim Streit

Contents

Chapter 1

Introduction

One of the most successful theories in physics is the Standard Model of particle physics (SM). The SM describes the characteristics of and interaction between the fundamental building blocks of the universe, so-called elementary particles. To test and improve the SM, the High Energy Physics (HEP) community performs experiments and compares the measurements with theory predictions. Although the SM successfully describes elementary particles in many aspects, some of them cannot currently be determined using first principles. One of these aspects is the composition of the proton that is described by so-called parton distribution functions (PDFs). The proton is often used in accelerator experiments, such as those at the Large Hadron Collider (LHC). Therefore, the quality of the analysis also depends on the quality of the PDFs. These PDFs are determined from data provided by multiple measurements. To further increase the precision of the PDFs, it is necessary to reduce the systematic and statistical uncertainty by further measurements.

This thesis describes one of the first triple differential cross-section measurements of the Z+Jet process using events recorded by the Compact Muon Solenoid (CMS) detector in 2017. This also includes corrections of detector effects via a specially designed unfolding method. This allows comparing the measured cross-section with theory predictions and measurements of other experiments. With the additional data provided by this analysis, the PDF statistical uncertainties will be reduced.

For HEP analyses and theory predictions, a massive amount of computing resources is needed. For example, several TB of data must be processed to perform the analysis described in this thesis. Additionally, for such an analysis, a similar amount of events has to be simulated, requiring a large amount of computing resources. With the planned upgrade of the LHC and further data taking periods, the amount of data will increase drastically in the following years. The financial investment in HEP specific computing centers to deal with this increasing demand is enormous and can not be easily increased. One contribution to mitigating the increasing demand is to integrate additional resources dynamically. There are various computing resource providers such as commercial cloud providers, high performance computing centers, or institute clusters. Some commercial cloud providers have periods with reduced costs, and institute clusters are usually not fully utilized at all times. The dynamic usage of resources not dedicated to HEP can help to cover peak loads. Furthermore, free resources at other clusters can be used, resulting in higher utilization of these clusters.

The dynamic integration of additional resources introduces some challenges. The first challenge is the provisioning of the needed software stack. HEP collaborations require a well-defined software stack for their experiment software. Another challenge is the integration of various resources of different types and hardware. In HEP, the resources used so far are almost homogeneous. With the integration of additional resources from different providers, the pool of resources gets more heterogeneous, resulting in more complex resource management. Another important point is that the resource management system has to be able to interact with different types of resource providers. A further challenge for data intensive tasks running on provisioned additional computing resources results from a limitation in the available network bandwidth. Computing tasks need to read their input data from storage systems via a network. If the network bandwidth between the storage system and computing resource is insufficient, the task runs inefficiently. To avoid such inefficiencies, a network aware resource scheduling is developed. The CPU-performance is another aspect of the usage of computing resources. In particular, dynamically integrated computing resources shared with multiple users may impact the CPU-performance. Especially for commercial cloud providers, this is important to estimate the provided performance per money. Therefore, CPU-performance benchmarks are performed on different systems to estimate the variation over time.

This thesis begins with the physical background that is introduced in chapter 2. Chapter 3 provides an overview of the experimental setup of the LHC and the CMS detector. The analysis to measure the triple differential cross-section including event selection and unfolding, is described in chapter 4.

In chapter 5, computing in scientific communities is described. This chapter includes the current situation in HEP computing and describes solutions to dynamically and transparently integrate additional resources from various resource providers. While this thesis is focused on the use-case of HEP, the described solutions can also be applied to other scientific communities. Chapter 6 illustrates the usage of these solutions based on two examples. The developed network aware resource scheduling is described in chapter 7. The CPU-performance benchmarks on several shared systems and their results are described in chapter 8.

At the end of this thesis, chapter 9 concludes this thesis with a recapitulation of the results. These are the measured triple differential Z+Jet cross-sections with data recorded in 2017 by the CMS detector and the comparison to theoretical predictions, as well as the dynamically and transparent integration of heterogeneous computing resources.

Chapter 2

Physics Background

The best theory to describe the fundamental structure of matter known to humankind is the SM. However, some parameters of the SM have to be measured and can not be predicted by theory. The Large Hadron Collider (LHC), as the currently most powerful particle accelerator, helps to improve our knowledge of particle physics by providing a window to the highest achievable energies. The LHC accelerates protons in two beams with opposite directions to almost the speed of light, further described in section 3.1. These proton beams cross each other at so-called interaction points. Thereby, two protons with opposite momentum can collide. Such a proton-proton collision is also called an *event*. If the momentum transfer between the two interacting protons is much higher than the mass of a proton, the constituents of the protons, so-called *partons*, interact with each other. This provides an insight into how the building blocks of the universe are fetched together and interact. To describe these interactions in detail, it is necessary to know the type of the initial partons and their momentum. This can be described by PDFs. A PDF represents the statistical distribution of the momentum fraction of the proton a parton carries. With the additional analyzed events, it is possible to include additional constraints on current PDFs. The PDFs will further be explained in section 2.3.

2.1 Proton-Proton Collisions

With current theoretical understanding, proton-proton collisions at the LHC can approximately be split into several convoluted parts due to the factorization theorem of QCD [1].

The first part in the factorization theorem describes how much the momentum fraction of a proton is carried by each parton. The probability of finding a specific parton with a certain energy fraction at a given momentum transfer at the collision is described by PDFs.

The second part is the hard interaction process. It describes the interaction of partons according to the perturbative description of the SM, which is further described in

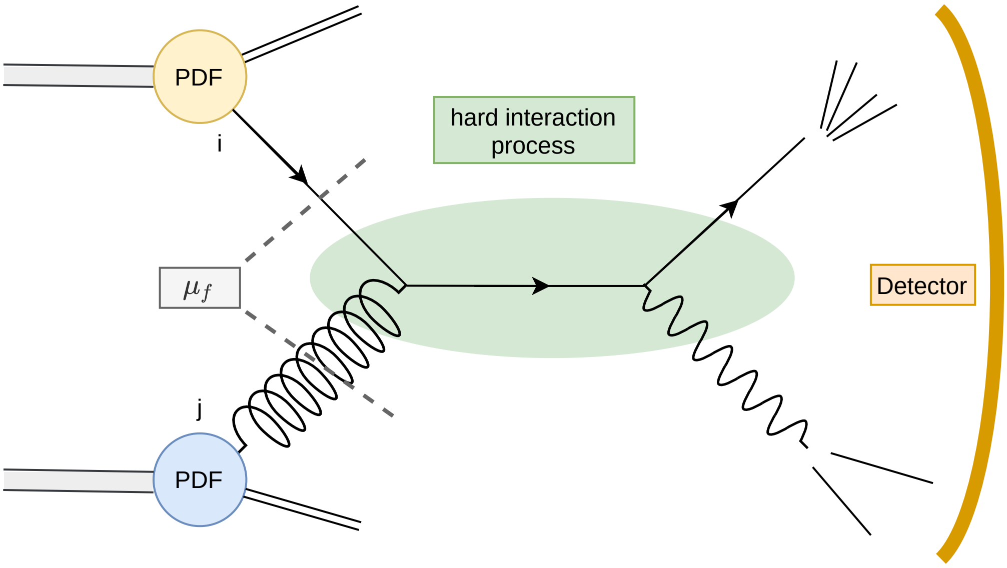


Figure 2.1: A proton-proton collision at the LHC. Two different types of partons interact with each other. These are the incoming particles for the hard interaction process. The remnants of the protons interact additionally and produce lower energetic particles. Within the hard interaction process, other particles are produced. The produced particles decay and create further particles. The detector can then measure the momentum and energy of the particles which live long enough to reach the detector.

section 2.2. For a full simulation of an event, leading order (LO) or next-to-leading order (NLO) calculations are used. However, for special analysis and processes, higher orders predictions are required for higher precision of the theoretical prediction.

To separate between the first and second part, it is necessary to decide at which energy the parton should be described by the hard interaction process or by the PDF. This scale is called the factorization scale (μ_F).

The third part is the generation of color-neutral particles from the color-charged partons. This so-called hadronization is simulated based on phenomenological models, described in section 2.2.

The fourth part is the detector interaction and detector response to the incoming particle. Also, the detector response and interaction are estimated by simulating the interactions of the color-neutral particles with the detector materials. All these steps together give a representative of an observed event by a detector at a collider. An example of a proton-proton collision can be seen in Figure 2.1.

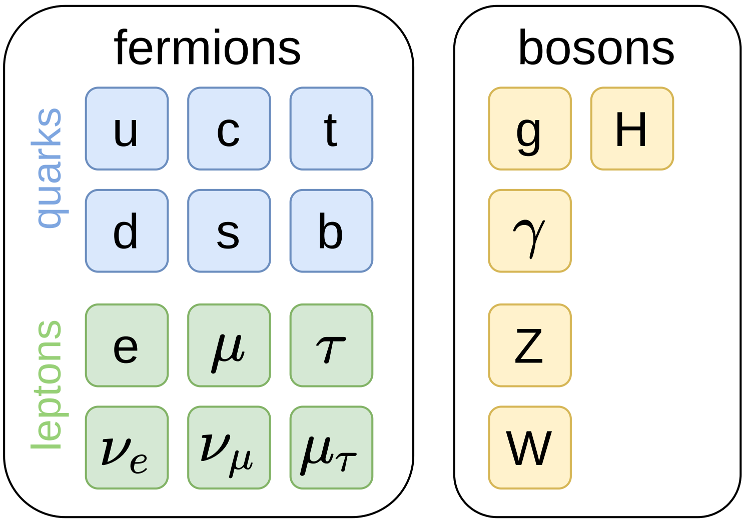


Figure 2.2: The particles of the SM are grouped into bosons and fermions. The fermions are further divided into quarks and leptons.

2.2 Standard Model

The SM describes the characteristics and interactions of the building blocks of matter. These building blocks can be subdivided into interaction *bosons* (gluon, photon, Z-, W-, and Higgs-boson) and *fermions*. Fermions can be grouped into types, the so-called flavor: *leptons* (electron, muon, tau, and corresponding neutrinos) and *quarks* (**u**p, **d**own, **c**harm, **s**trange, **t**op, **b**ottom); see Figure 2.2.

The most important bosons for the following analysis are the gluon (g) and the Z boson. The gluon is the interaction particle of the strong force, so-called quantum chromodynamics (QCD). The Z boson is one of the neutral interaction particles of the combined Electro-Weak force.

2.2.1 Quantum Chromodynamics

The QCD is the force that holds the nucleus together against the large electromagnetic repulsion of protons. The gluon is the interaction particle of the QCD. It is massless and can interact with all color-charged particles, namely quarks and gluons. As a result, a gluon can interact with itself. Due to the nature of this self-interaction, the force between two color-charged particles increases with distance. This results in one feature of the QCD, the *confinement*. If two color-charged particles move apart, their kinetic energy decreases with distance and gets converted into potential energy of the color field. New particles can be created in pairs if the potential energy of the color field is high enough. Therefore, quarks are confined to exist in bound states, so-called hadrons. Due to the threefold characteristic of color charge, the most common hadrons are quark-antiquark bound states (mesons) and three-quark bound states (baryons).

One such three-quark state is the proton. Its building blocks, partons, are valence quarks, sea quarks, and gluons. The valence quarks of a baryon are the quarks that mainly define the characteristics of the baryon, e.g., electrical charge. For a proton, the valence quarks are two up quarks and one down quark. Due to the gluon interaction between the valence quarks and vacuum fluctuations, additional quarks, so-called sea quarks, exist in a proton.

At the center-of-mass energy of the colliding protons achieved at the LHC, the energy of colliding partons is high enough to produce various hadrons in a cascade (hadronization). The cascade ends if the individual quarks and gluons do not have enough energy to produce further hadrons. These hadrons propagate in a similar direction as the original quark or gluon and form a shower of hardons and corresponding decay products, a so-called *jet*.

2.2.2 Electro-Weak Force

The Electro-Weak force is a unified description of the electromagnetic force (e.g., photoelectric effect [2]) and the weak force (e.g., nuclear decay [3]) [4]. It has four interaction particles: the photon (γ), two oppositely charged W-bosons, and the Z boson. Because the Z boson is essential for the following analysis, it will be further described.

The Z boson interacts with all fermions and has a mass of $m_\mathrm{Z} = 91.2\,\mathrm{GeV}$ [5]. Due to its short lifetime of $\tau_\mathrm{Z} = 2.6 \times 10^{-25}\,\mathrm{s}$, it is not directly detectable. However, it is possible to reconstruct it from its decay products, e.g., a $\mu^+\mu^-$-pair. According to the short lifetime of the Z boson and the energy-time uncertainty principle, the Z boson has a decay width of 2.5 GeV. Therefore, the invariant mass of the two decay particles from the Z boson is around the mass of the Z boson. The CMS detector is very suitable to measure charged muons. Therefore, Z bosons reconstructed from muons provide a clear signal to identify events that contain a Z boson.

At the LHC, Z bosons are directly produced via quark-antiquark annihilation at leading order, see Figure 2.3. With this process, it is possible to determine further constraints on quark PDFs. To also get information about the gluon PDF, Z+Jet events are more useful. The leading order production mechanism is shown in Figure 2.4, where one of the partons is a gluon.

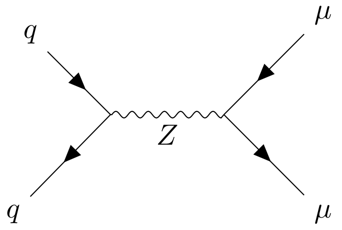
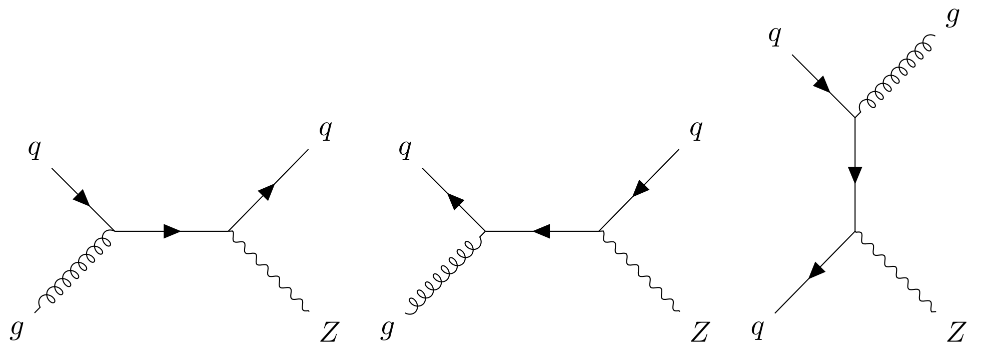


Figure 2.3: Quark-antiquark annihilation for Z boson production with Z to $\mu^+\mu^-$.



(a) quark-gluon

(b) antiquark-gluon

(c) quark-antiquark

Figure 2.4: Leading Order production of Z+Jet events.

2.3 Parton Density Function

At a collision of a proton and another particle, e.g., a proton or an electron at an energy of at least a few GeV, the partons of the proton interact with the other particle. Each parton carries a fraction of the energy of the whole proton and is represented by x. However, the probability of interacting with a parton of a given x is different for each parton. The probability distribution of each parton is described by a parton distribution function (PDF).

Currently, it is not possible to determine the PDFs by first principles. However, it is possible to estimate PDFs by fitting theory predictions to measured data. Therefore, the cross-section measurements of many different processes are used.

The cross-sections of proton-proton collisions to particles X can be described via the PDFs and the theoretical prediction of the hard interaction process. This is done by the sum over all parton combinations (i, j) integrated over the PDFs of the parton $(f_i(x_i, \mu_{\mathrm{F},Q^2})$ multiplied by the corresponding square amount of the matrix element of the hard interaction process ($\mathcal{M}$). This can be written as:

$$\sigma_{pp\to X} = \sum_{i,j} \int \int dx_i dx_j f_i(x_i, \mu_F^2, Q^2) f_j(x_j, \mu_F^2, Q^2) \mid \mathcal{M}(x_i, x_j, \mu_F^2, \mu_R^2) \mid^2 \quad (2.1)$$

The matrix element can be determined by perturbation theory with the parameter values of the SM as input.

The theoretical prediction of a cross-section includes perturbative QCD for the hard interaction process and parametrizations with some physical assumptions for the PDFs. Therefore, three parameters need to be defined. The first parameter is the renormalization scale (μ_{R}^2). The perturbative QCD contains some divergences which are handled via renormalization. The renormalization makes the coupling constant energy-dependent, the so-called running coupling constant. The renormalization scale defines at which energy the coupling constant is determined.

The second parameter is the factorization scale (μ_{F}^2). According to the QCD, a parton can emit a gluon with a fraction of its energy. The probability for that gets higher, the lower the energy of the emitted gluon is. The factorization scale defines which energy particle emissions of the parton are described by the PDF or by the hard interaction process.

The third parameter is the resolution scale Q^2. Usually, the factorization scale and the renormalization scale is set to the value of the resolution scale. The resolution scale is in the range of the momentum transfer during the collision. As a result, the resolution scale depends on the center-of-mass energy of the collider. The Dokshitzer-Gribov-Lipatov-Altarelli-Parisi (DGLAP) equation enables transferring PDFs measured by a given resolution scale to another [6–8]. Thereby, it is possible to combine measurements from different experiments.

Different parameterizations exist for the description of PDFs. Usually, PDFs are published in sets that contain one PDF per parton based on several datasets from various experiments. Most of these sets use for the parametrization of analytical functions, such as the CT14 PDFs [9]. It is also possible to describe PDFs via neural networks [10], where the parametrization gets defined by the neural network. These PDFs have to be fitted to measured event distributions. One of these PDF sets is shown in Figure 2.5.

The usual calculation of the cross-section via equation 2.1 needs more than 100.000 CPU hours for jet production at next-to-next-leading order (NNLO) precision for one PDF set. The calculations of that with different PDF parameters for a fit would result in a huge amount of needed CPU time. However, the factorization theorem enables splitting the PDFs from the hard interaction process [12, 13]. Therefore, it is possible to calculate tables with multiple approximated matrix elements for different parameters. These tables enable to calculate the matrix element for a given PDF parameter set much faster than a complete calculation of the cross-section. The

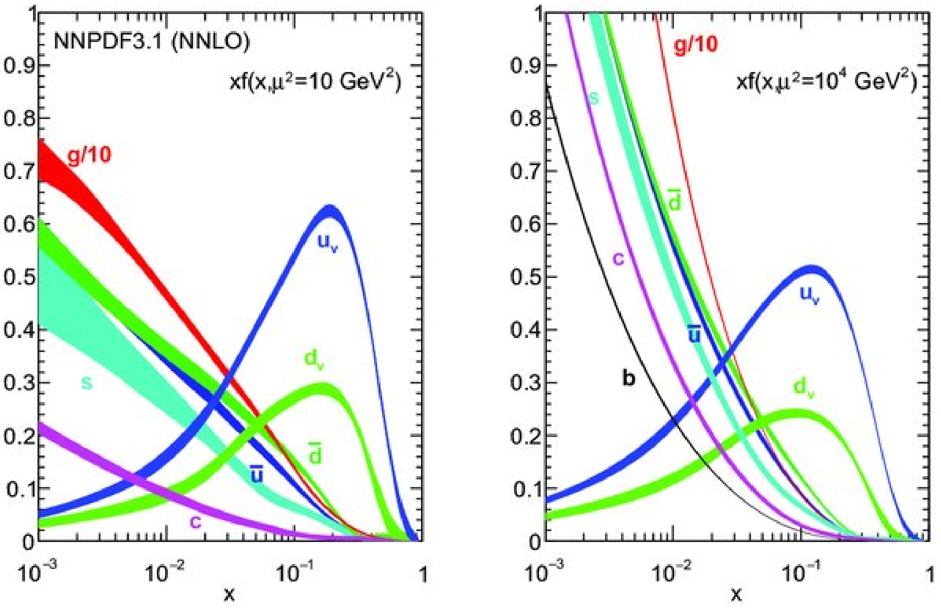


Figure 2.5: PDF set NNPDF 3.1 with the studied partons[11]. The left plot shows the PDFs at $\mu^2 = 10\,\mathrm{GeV}^2$ and the right plot at $Q^2 = 10^4\,\mathrm{GeV}^2$. In these plots, Q^2 is represented by μ^2. [11]

typical production of such tables takes about twice as long as just the calculation of the cross-section. For tables at NNLO order precision, about 250.000 CPU h are needed. [14]

The quality of the fitted PDF set depends profoundly on the quality of the used cross-section datasets. The LHC enables some improvements to the PDFs description. Firstly, at the LHC, massive amounts of data are produced, which can be used to reduce statistical uncertainties. Secondly, proton-proton collisions have a high fraction of events where one of the interacting particles is a gluon. This enables a closer look at the gluon PDF. The goal of the following analysis is to provide a further measurement to enable additional constraints to the PDFs.

Chapter 3

Experimental Setup

The CMS detector and several other High Energy Physics (HEP) experiments are located at Conseil européen pour la recherche nucléaire (CERN) near Geneva. Some HEP experiments, such as CMS, measure and study high energy particle collisions. Therefore, particles are accelerated into packages, so-called bunches, by alternating electric fields. Since these bunches follow each other only with a short time interval, one speaks of a beam. Several such particle accelerators are necessary to bring particles on the highest energy currently possible. An overview of the located accelerators and experiments at CERN is shown in Figure 3.1. The accelerator with the highest center-of-mass energy is currently the LHC.

3.1 Large Hadron Collider

The LHC is a circular hadron collider with a ring circumference of 26.6 km. During its "Run II" phase between 2015 and 2018, the LHC mainly accelerates protons up to 6.5 TeV, which results in a center-of-mass energy of $\sqrt{s} = 13\,\text{TeV}$ for the collision of two protons. Those are the most energetic particle collisions humankind ever produced. [16, 17]

The LHC also provides a huge amount of particle collisions. In HEP, the number of recorded collision events in a dataset or run period is commonly represented by the integrated luminosity (L_{int}). The integrated luminosity is the number of events multiplied by the cross-section of these events:

$$L_{\text{int}} = \sigma \times N$$

With further upgrades of the LHC, the luminosity, the number of events multiplied with their cross-section per time interval, will increase.

The most extensive LHC upgrade planned is the upgrade to the High Luminosity LHC (HL-LHC) [18]. It is planned to increase the center-of-mass energy to $\sqrt{s} = 14\,\text{TeV}$ and further increase the luminosity, see Figure 3.2. The increased luminosity

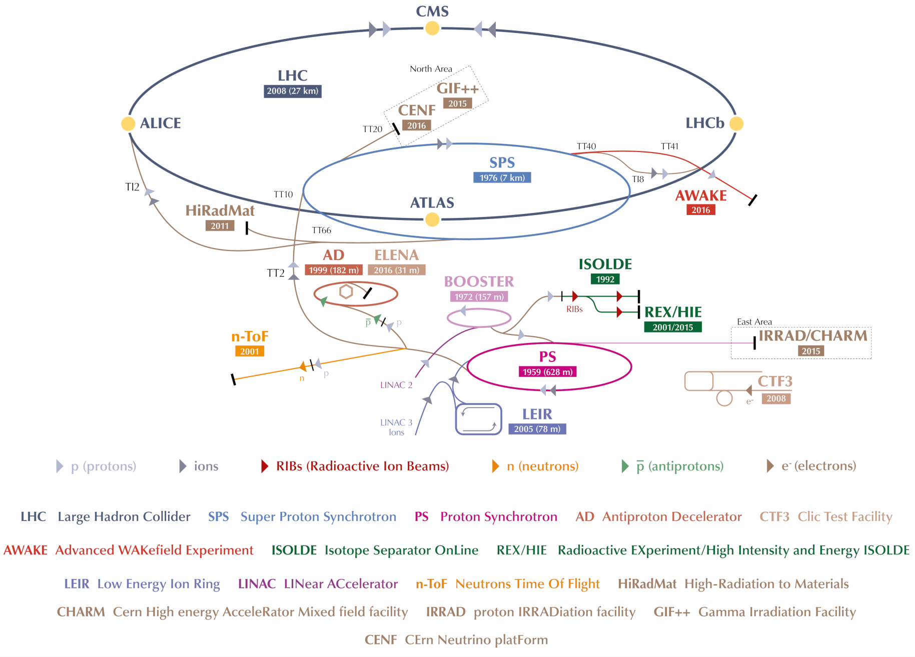

CMS
LHC
2008 (27 km)
North Area
GIF++
2015
CENF
2016
ALICE
LHCb
TT20
TT40
TT41
SPS
1976 (7 km)
TI8
TI2
TT10
ATLAS
HiRadMat
2011
AWAKE
2016
TT66
AD
1999 (182 m)
ELENA
2016 (31 m)
TT2
ISOLDE
1992
BOOSTER
1972 (157 m)
RIBs
REX/HIE
2001/2015
East Area
IRRAD/CHARM
2015
p̄
p
n-ToF
2001
n
p
PS
1959 (628 m)
CTF3
2008
e⁻
LINAC 2
LINAC 3
Ions
LEIR
2005 (78 m)
p (protons)
ions
RIBs (Radioactive Ion Beams)
n (neutrons)
p̄ (antiprotons)
e⁻ (electrons)
LHC Large Hadron Collider
SPS Super Proton Synchrotron
PS Proton Synchrotron
AD Antiproton Decelerator
CTF3 Clic Test Facility
AWAKE Advanced WAKefield Experiment
ISOLDE Isotope Separator OnLine
REX/HIE Radioactive EXperiment/High Intensity and Energy ISOLDE
LEIR Low Energy Ion Ring
LINAC LINear ACcelerator
n-ToF Neutrons Time Of Flight
HiRadMat High-Radiation to Materials
CHARM Cern High energy AcceleRator Mixed field facility
IRRAD proton IRRADiation facility
GIF++ Gamma Irradiation Facility
CENF CErn Neutrino platForm

Figure 3.1: Particle accelerators and experiments located at CERN. The currently largest particle accelerator is the LHC. The protons which enters the LHC are pre-accelerated via the Linac3, PS, and SPS particle accelerators. Figure is based on [15].

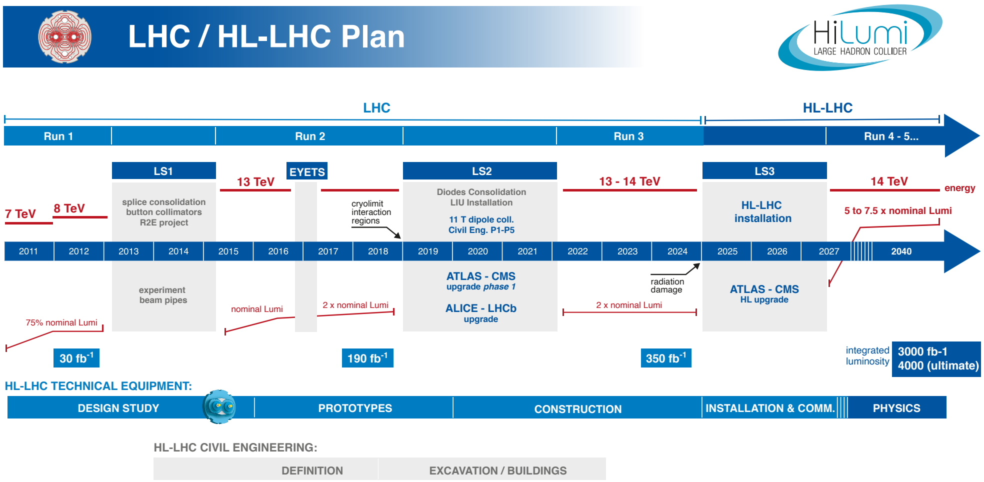


Figure 3.2: LHC upgrade plan with detector and LHC upgrades. [19]

results in a much higher data rate that challenges the detectors and the computing infrastructure to provide enough resources for analyzing these huge amounts of data.

The LHC accelerates protons in two beams in opposite directions. At several points, so-called interaction points, the two beams cross each other, and particles of the two beams collide. At four of these interaction points where the protons collide, the events are recorded by one particle detector. These are designed for a different purpose:

- The A Large Ion Collider Experiment (ALICE) detector is designed to record heavy-ion collisions. These collisions are done at the LHC instead of the usual proton-proton collisions for short times during a run period. During these collisions, a strongly interacting matter, so-called quark-gluon plasma, exists. This state of matter is similar to the matter in the universe shortly after the big bang. [20]

- The Large Hadron Collider beauty (LHCb) detector is designed for studying the b quark. Processes that involve b-quarks can give a hint of matter-antimatter asymmetry. This asymmetry is responsible that our universe is full of matter, where antimatter is almost gone. [21]

- The A Toroidal LHC ApparatuS (ATLAS) detector is a multi-purpose detector. It allows to make precision measurements of the SM and enables the search for undetected particles. [22]

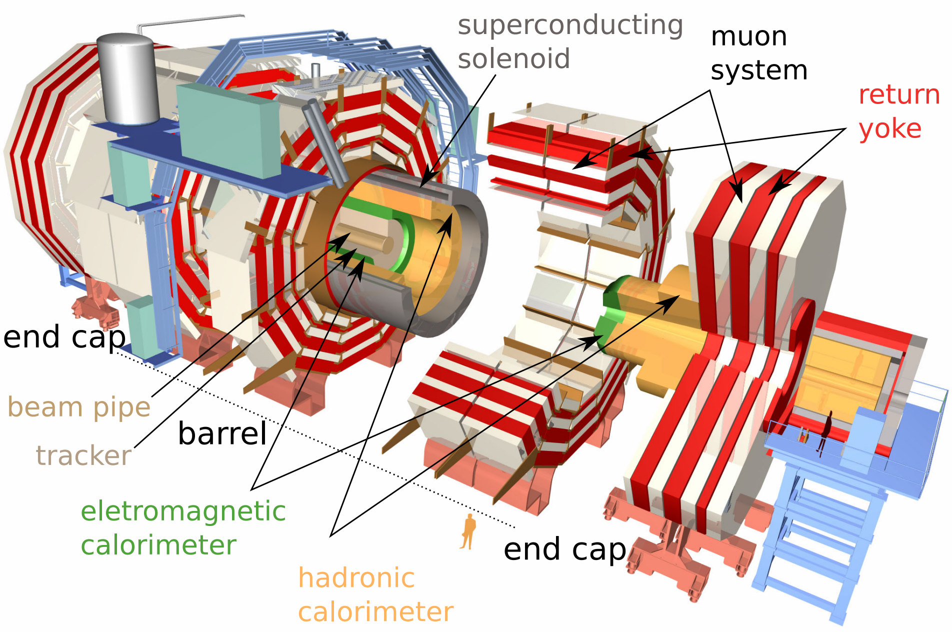


Figure 3.3: Sketch of the CMS detector with its subsystems. The detector has a central region, a so-called barrel, and two end caps. The tracker and the superconducting solenoid are only within the barrel. The calorimeters, muon system, and return yoke are in the barrel and in the two end caps. Based on [24]

- The CMS detector is, as the ATLAS detector, a multi-purpose detector. However, it has some differences in used material and construction. These differences enable us to cross-check the results of each other.

3.2 CMS Detector

The CMS detector was built to record data for precision measurements of the SM and search for new particles. The detector includes several subsystems and is built of a cylindrical section (also called barrel), with two end caps around the beam pipe see Figure 3.3. The following is a short introduction to the subsystems of the CMS detector and its coordinate system that are important for the analysis. A more detailed description of the CMS detector can be found in [23].

3.2.1 Coordinate System

The CMS detector is radially symmetrical around the beam pipe. The center of its coordinate system is positioned at the interaction point and is shown in Figure 3.4.

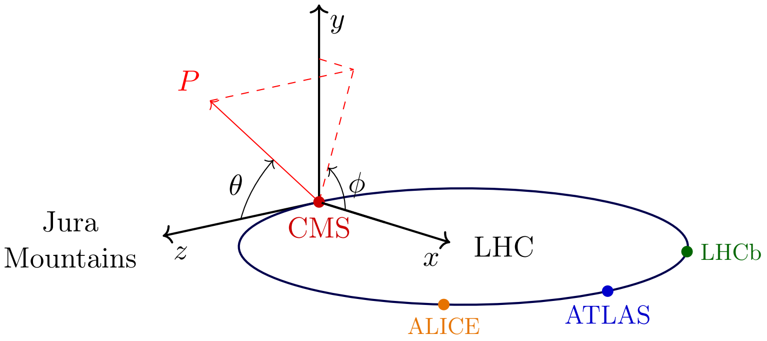


Figure 3.4: CMS coordinate system based on [26]

The x-axis of the coordinate system points to the center of the LHC ring. The y-axis points upwards, and the z-axis points west along the beam pipe in the direction of the Jura mountains.

Events are usually described in spherical coordinates. The angle between the x-axis and the y-z plane is the azimuthal angle ϕ, and the polar angle θ is measured from the z-axis. [25]

3.2.2 Subsystems

With the information of several subsystems, it is possible to identify and measure the momentum of most of the particles flying through the detector. Figure 3.5 shows a transverse slice in the center of the CMS detector with its subsystems and their interaction with particles. The subsystems will be described in the order the particles move through the detector.

In the center of the detector is the particle collision point. Due to the fact that the particles are accelerated in bunches, there are several collisions per bunch cross. During the year 2017, on average, 37 proton-proton collisions per bunch cross happened [28].

Around the interaction point are layers of the tracker. The tracker is built of semiconducting material that sends an electric signal when a charged particle moves through. Thereby it is possible to determine the trajectory by the hits in the layers. This enables us to determine its original proton-proton collision out of the others during a bunch cross.

Inside the CMS detector is a magnetic field that is induced by a superconducting solenoid; see later. Thus the charged particles have a curved flight path, which additionally enables to determine the momentum of these particles.

To improve the performance as well as the resolution of the tracker, it was updated between the 2016 and 2017 run period of the LHC. The updates include smaller and

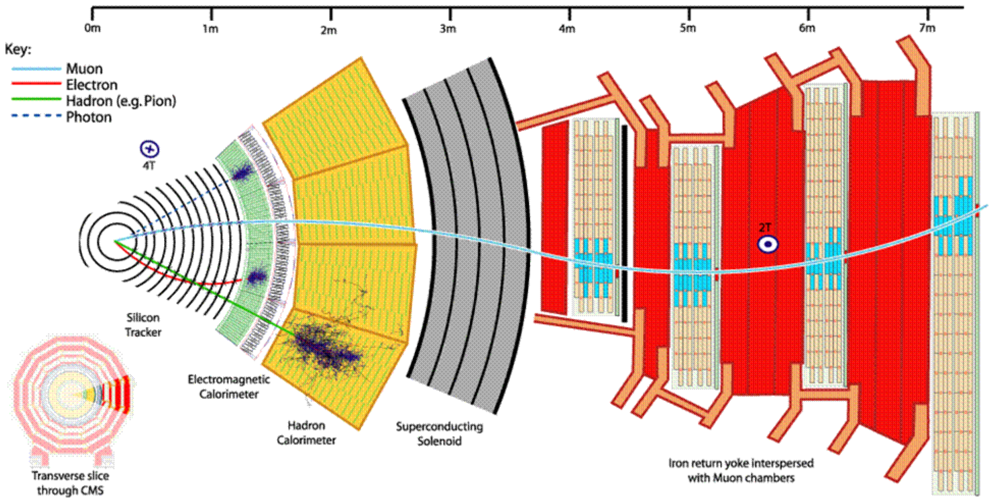


Figure 3.5: A part of a transverse slice of the CMS with its subsystems. Each type of the long life particles which reaches the detector (muons, electrons, photons, and hadrons) interact with other subsystem. [27]

more pixels, one additional layer in the inner tracker, and a closer position of the first layer to the bean pipe. [29, 30]

The electromagnetic calorimeter (ECAL) of the CMS detector is a homogeneous calorimeter. That means that the absorption and scintillation material is the same, here lead tungstate crystals. Thereby, photons, electrons, and electrically charged hadrons interact with the material of the ECAL and create an electromagnetic shower. The charged particles of the shower create light through the crystal scintillation. That light is converted into electrical signals.

Around the ECAL is the hardon calorimeter (HCAL). The HCAL measures the energy of electrically neutral hadrons. Similar to the ECAL, the hadrons form hadronic showers. This shower also has electrically charged particles that create scintillation light that also gets converted into electrical signals to measure the energy of the hadrons.

As mentioned before, the momentum of charged particles can be determined by the curvature of their trajectory within a magnetic field. The more the trajectory gets curved, the more accurate is the momentum resolution. For that, a strong magnetic field is required and is produced by a superconducting solenoid. The magnetic field inside the solenoid has a strength of about 4 T. Outside of the solenoid is the iron return yoke that gives the magnetic field an ordered structure. There, the magnetic field is about 2 T strong.

Inside the iron return yoke is the muon system. Due to its characteristics, the muon is the only charged particle that can reach the muon system from the collision point in the CMS detector. The muon system is built of gaseous particle detectors. Via ionization in the gaseous detectors, the track of the muons can be detected. Together with the reconstructed track in the tracker, it is possible to measure the muon momentum.

All these components together produce about 1 MB of data per bunch cross. In combination with the collision rate of 40 MHz, an immense data rate has to be handled. However, it is not possible to store the data with that high data rate. Furthermore, it is not possible to analyze such big data sets. Therefore, it is necessary to reduce the data rate. The reduction of the data rate of events, which should be stored and get analyzed later, is done by a fast and simple event selection.

The CMS detector has a Level-1 (L1) Trigger and a High-Level-Trigger (HLT). The L1 trigger is integrated into the detector and selects events based on the muon system and the calorimeters. The event rate after the L1 trigger is around 100 kHz. The HLT is a software trigger that runs on a cluster of about 1000 CPU cores. To select events, the HLT uses the information of all detector systems. After the HLT, the event rate is in the magnitude of 100 Hz. The events that passed the HLT get stored as data sets and reconstructed at CERN. For further analysis and improved event reconstruction, a copy of the data sets are distributed to data centers across the world, see chapter 5.

Chapter 4

Measurement of Triple Differential Z+Jet Cross-Section

Triple-differential Z+Jet cross-section measurements open up a window to a better understanding of the PDFs. The first triple differential Z+Jet cross-section measurement and comparison with predictions at NNLO accuracy were performed in a former analysis with the data recorded by the CMS experiment [31]. In that analysis, only data taken in 2016 by the CMS detector was used.

The following analysis is an improved version of the former and is performed on data recorded in 2017 with the CMS detector. As a cross-check, the data recorded in 2016 is also analyzed and compared with the former analysis.

First, the characteristics of Z+Jet events are discussed, followed by a discussion of the observables. After that, the event selection and the differences between the former analysis [31] and the current analysis are shown. The measurements are compared with simulations and afterward unfolded to consider detector effects. This makes it possible to compare the data measured by the CMS detector with measurements from other detectors and theory predictions.

4.1 Characteristics of Z+Jet Events and Observables

In this analysis, Z+Jet events with a Z boson and at least one jet are studied. Further jets can exist in events due to QCD radiation and pileup. The jet with the highest transverse momentum (hardest jet) is used for further analysis because it has the highest probability to originate from the hard interaction.

Z bosons can be measured more precisely than jets. This is because Z bosons can be reconstructed from muons, which provide a clear signal in the detector and can be precisely measured. Therefore, the transverse momentum of the reconstructed Z boson ($p_{\mathrm{T}}^{\mathrm{Z}}$) is chosen to be one of the main observables to describe Z+Jet events.

Z+Jet events are produced in a proton-proton collision where two partons of the protons interact. By studying the topology of Z+Jet events, it is possible to get in-

Table 4.1: This table shows the $p_{\mathrm{T}}^{\mathrm{Z}}$ binning for the three y_b-y^* regions (central, edge, and extra).

binning	**bin edges** (GeV)
central (C)	25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 110, 130, 150, 170, 190, 220, 250, 400, 1000
edge (E)	25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 110, 130, 150, 170, 190, 250, 1000
extra (X)	25, 30, 40, 50, 70, 90, 110, 150, 250

formation about the properties of the interacting partons. In particular, the rapidity sum of the Z boson and the hardest jet

$$\mathrm{y_b} = \frac{1}{2}|\mathrm{y}^{\mathrm{Z}} + \mathrm{y}^{\mathrm{jet1}}| \tag{4.1}$$

is linked to the boost of the Z+Jet center-of-mass system. Based on that, the momentum fraction of the interacting partons (x_1, x_2) can be estimated.

Another variable is the difference between the rapidity of the Z boson and the hardest jet,

$$\mathrm{y}^* = \frac{1}{2}|\mathrm{y}^{\mathrm{Z}} - \mathrm{y}^{\mathrm{jet1}}|\,, \tag{4.2}$$

which is correlated to the scattering angle in the center-of-mass system. The former analysis showed that the rapidity difference (y^*) of the two analysis objects is sensitive to the composition of the different production processes in the scattering theory. Since, different production processes include different partons, such measurements can provide information about the PDFs.

The three observables, $p_{\mathrm{T}}^{\mathrm{Z}}$, $\mathrm{y_b}$ and y^*, are used to categorize events in the former analysis [31]. For this, the cross-section in bins of these observables is measured. To compare this analysis with the former analysis, the same binning is used. For $\mathrm{y_b}$ and y^*, an equal bin width of 0.5 for both observables has been chosen. Furthermore, no events with $\mathrm{y_b}$ or y^* greater than 2.5 are considered. Due to the limited amount of events in some $\mathrm{y^b}$-y^* regions, three different $p_{\mathrm{T}}^{\mathrm{Z}}$-binnings are defined.

The $\mathrm{y_b}$-y^* binning is depicted in Figure 4.1 together with the corresponding $p_{\mathrm{T}}^{\mathrm{Z}}$-binning categories **C**entral, **E**dge, and e**X**treme. The $p_{\mathrm{T}}^{\mathrm{Z}}$ binning used for each category is shown in Table 4.1.

These three-dimensional bins are arranged in a linear sequence for the unfolding procedure discussed in section 4.4. Therefore, this three-dimensional binning is converted into a one-dimensional binning that is shown in Figure 4.2.

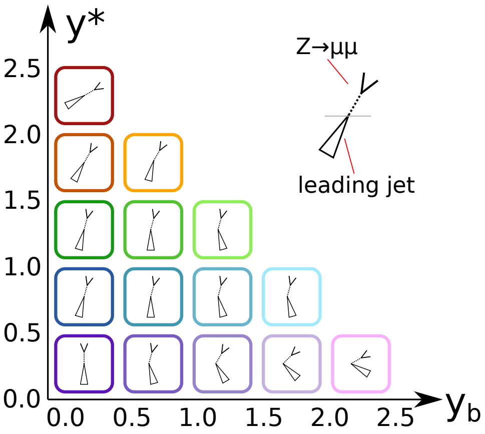


Figure 4.1: Two overview plots with the y_b-y^* binning are shown. On the left hand side, the y_b-y^*-binning with a symbolic representation of the orientation of the hardest jet and the Z boson in the detector frame is depicted. On the right hand side, the p_T^Z-binning for each y_b-y^*-bin is shown.

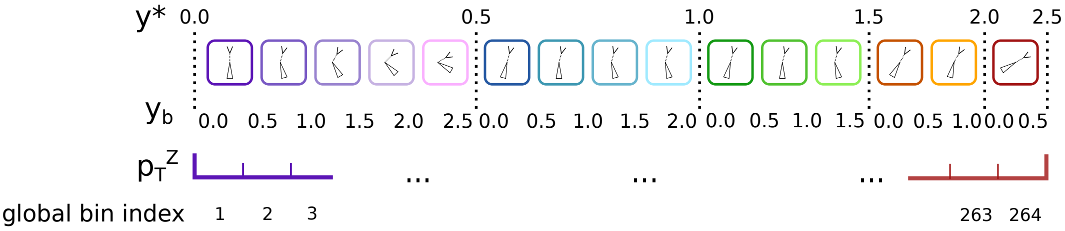

y* 0.0 0.5 1.0 1.5 2.0 2.5
y_b 0.0 0.5 1.0 1.5 2.0 2.5 0.0 0.5 1.0 1.5 2.0 0.0 0.5 1.0 1.5 0.0 0.5 1.0 0.0 0.5
p_T^Z
...
global bin index 1 2 3 263 264

Figure 4.2: Arrangement of the 3-dimensional bins into a linear sequence. All bins are sorted by y^*, y_b, p_T^Z bins. In this process, each y_b-y^*-p_T^Z bin is assigned to a global bin index from 1 to 264.

4.2 Event Selections and Corrections

An event selection is chosen to discard background events and thus increase the fraction of Z+Jets events in the sample. Only events within the defined phase space pass the selection. The selections are usually based on observables related to the kinematic properties of objects in an event, such as muons or jets, and are applied to measured and simulated events. In the following, reconstruction, corrections, and selections used by the triple-differential Z+Jet cross-section measurements, both in the former analysis and the current analysis are presented.

The first event selection is performed by the trigger system. The trigger system filters events based on detector information with a simple object reconstruction. Most triggers make their decision based on the transverse momentum of simple reconstructed particles. If the transverse momentum of such a particle is above a threshold, the trigger fires and the event is marked for readout and further reconstruction. For the dataset recorded in 2016 and 2017, a trigger is used that selects events with at least one muon. The threshold of the transverse muon momentum is different for both years. In the former analysis of the 2016 data, the threshold was chosen to be $\mathrm{p}_\mathrm{T}^\mu \geq 24\,\mathrm{GeV}$ [31]. For the 2017 data taking period, the trigger threshold was set to $\mathrm{p}_\mathrm{T}^\mu \geq 27\,\mathrm{GeV}$. However, the trigger is not 100% efficient. Therefore, the muon physics object group (MuonPOG) measures the trigger efficiency ($\epsilon_\mathrm{trigger}$) via a tag and probe method [32, 33]. To take this efficiency into account, data events are weighted with a factor

$$w_\mathrm{trigger} = \frac{1}{\epsilon_\mathrm{trigger}} \tag{4.3}$$

This gives an estimate for a 100% efficient trigger, which is assumed in the simulations used for this analysis.

Under certain conditions, the L1 trigger system can veto consecutive events. Such an event veto happens when a significant amount of energy is detected within a region of the ECAL. However, due to a bug in the L1 trigger system, a gradual timing shift was not correctly propagated from the ECAL to L1 trigger primitives [34]. As a result, an event can mistakenly veto itself. Such a veto appeared with a probability depending on η and p_T of all jets and photons, with overlaps being taken into account. This probability has been derived in [35]. It is used to correct the veto by applying weight on data events.

$$w = 1 - P(\mathrm{prefire}) = \prod_{i=\mathrm{photons,jets}} \Big(1 - \epsilon_i^\mathrm{prefire}(\eta, p_\mathrm{T})\Big) \quad . \tag{4.4}$$

This bug affects events recorded in the years 2016 and 2017.[35]

An improved object reconstruction is the particle-flow (PF) algorithm. The PF algorithm creates particle candidates by using information from all available detector subsystems in the CMS detector, which improves the precision compared to a measurement of particles with a single sub-detector system. These particle candidates are used to reconstruct and correct objects measured in this analysis. [36]

First, muons are reconstructed from the PF candidates. A muon candidate has to fulfill several criteria to be identified as a muon object. The first criterion is the muon identification (muon ID) provided by the PF algorithm. The second criterion is muon isolation. Thereby, the energy deposition in the detector inside a cone around the reconstructed muon track is used to calculate a discriminating variable. These two criteria reduce the number of hadrons misidentified as muons and muons from subsequent and preceding events or from hadron decays. To compare the criteria between different data taking periods, efficiency working points are defined. These working points define the threshold value of criteria as a function of the identification efficiency. The transverse momentum of the reconstructed muons is corrected for detector effects and losses due to bremsstrahlung. [32]

Jets are the most common object in events measured at LHC detectors. In the following analysis, jets are reconstructed via the anti-k_t jet algorithm [37] with the distance parameter of $R = 0.4$. To remove the charged hadron contribution from pileup, the *charged hardon subtraction* (CHS) algorithm is used [36]. Due to detector effects, the measured transverse momentum of jets is different from the real transverse momentum. This effect is corrected via *jet energy correction* (JEC). Additionally, the *jet energy resolution* (JER) is different between measurement and simulation. The simulation assumes a better JER than expected in the measured data. With the JER determined in data, the jet energy in simulations is smeared according to the JER of the data. [38]

After the corrections, the jets are required to pass two identification steps to reduce objects wrongly identified as jets and jets produced by pileup. The first is the particle flow jet ID [39] which includes observables such eas fraction of neutral hadrons ECAL and HCAL. The second is the pileup jet ID [40] which includes observables such as the multiplicity of charged and neutral particles and the jet profile.

Based on the corrected and selected objects, the event selection in phase space is performed. First, events are selected with a Z boson that decays into two muons ($Z \to \mu^+\mu^-$). Therefore, the events must have at least two muons with an opposite charge with a transverse momentum of at least p_T^μ> 28 GeV. Furthermore, only muons within the pseudorapidity of $|\eta| < 2.4$ are selected. The invariant mass of the two muons must be in a range of 20 GeV around the Z boson mass ($m_{\mathrm{Z}^\mathrm{PDG}} =$ 91.1876 GeV [5]). If more than two muon pairs exist, the oppositely charged muon pair is chosen with an invariant mass closest to m_Z. For the further analysis, only this chosen muon pair is used. The transverse momentum of the Z boson reconstructed from the selected muon pair is required to be bigger than $p_\mathrm{T}^\mathrm{Z} > 30$ GeV. Secondly, events are selected that have a jet produced in the hard process. Only jet objects are taken into account that are not in a cone of $R = 0.3$ around the two selected muons. This reduces the probability of getting a muon mistakenly identified as a jet. For the event to pass the selection, at least one jet with transverse momentum greater than $p_\mathrm{T}^\mathrm{jet} > 20$ GeV and pseudorapidity in the range of $|\eta| < 2.4$ is required.

Unfortunately, the detector simulation is an idealized model of the CMS detector and thus results in residual differences in selection efficiencies between recorded data and simulation. This is the case for muon identification and muon isolation. The difference between data and simulation is corrected by applying scaling factors on data based on the selected muons in the event. The corresponding scaling factors are derived and provided by the CMS Muon POG [33]. Furthermore, the pileup jet ID selection efficiency is different in data compared to the one in simulation. Therefore, an event-based scaling factor is applied on the hardest jet in data. The scaling factors are derived and provided by the CMS JetMET group [40].

4.3 Measurements and Simulations

With the selections of events, it is possible to create an event collection enriched with Z+jet events. Such a signal process enriched dataset still contains events originating from other processes but passing the selection criteria, so-called background processes. Some processes have a similar event final state as the Z+jet production process. Events of these processes are indistinguishable from events originating from the Z+jet production process. Additionally, due to misidentification of physical objects, some events of background processes look similar to signal events on reconstruction level. However, with the simulation of signal and background processes, it is possible to estimate the background contribution in recorded data. The background processes in this analysis are: top-antitop-quark pair production with associated jets, single top-quark and antitop-quark production in the t-channel and tW-channel, double Z-boson production, and double vector boson production. More detailed information about the backgrounds can be found in the former analysis [31]. Additionally to the background estimation, a comparison between data and Monte Carlo simulation is useful to check that the detector simulation performs as expected.

4.3.1 Comparison to Former Analysis

Since the triple differential Z+jet cross-section measurement of data recorded by the CMS detector in the year 2016 by T. Berger [31], the understanding of the detector has improved. This improved understanding is directly included in the analysis of the data recorded in 2017. As a cross-check, the improved understanding is also included into the data recorded in 2016 and is compared to the data recorded in 2016 and analysed by T. Berger.

The following improvements and updates are performed on the former analysis:

- the transverse momentum muon corrections are updated
- update to most recent JEC
- update to most recent JER
- to compare the data recorded 2016 with the data recorded in 2017, the selection on the transverse momentum of the muon has to be $p_{\mathrm{T}}^{\mu}> 28\,\mathrm{GeV}$ instead of $p_{\mathrm{T}}^{\mu}> 25\,\mathrm{GeV}$, due to the changed trigger threshold
- updated separation in η-ϕ space used to determine the lepton isolation
- updated jet ID to the recommended working point
- applied pileup-jet ID scaling factors

- applied event weights to correct efficiencies caused by the L1 trigger bug, see chapter 4.2
- updated cross-section used to scale the simulated events of the WZ process to the number of expected events
- add single top- and single antitop-quark tW-channel to the considered background processes
- MadGraph5 [41] Z+0,1,2 jets at NLO accuracy multijet merged with FXFX interfaced to Pythia 8 [42] is used for the simulation of the signal process instead of MadGraph5 Z+0,1,2,3,4 jets at LO accuracy multijet merged with MLM method interfaced to Pythia 8
- used newer Monte Carlo simulations to consider the improved understanding of the detector

A list of the datasets used in the former analysis is shown in Appendix A.1, and the datasets used for the current analysis of the data recorded in 2016 are shown in Appendix A.2. The simulated datasets are scaled with the measured integrated luminosity of $L = 35.9$ [43, 44] to get the expected amount of events. In the following, some distributions of observables are shown for the former and the current analysis for comparison.

Figure 4.3 shows the distribution of the mass of the reconstructed Z boson in data and simulation on reconstruction level. The ratio between the distribution of the measured data and simulation shows a better modeling of the Z boson mass resolution in simulation for the former analysis, which points to a better understanding of the detector than the current analysis. However, this could also be due to the mismodeled WZ-background contribution in the former analysis.

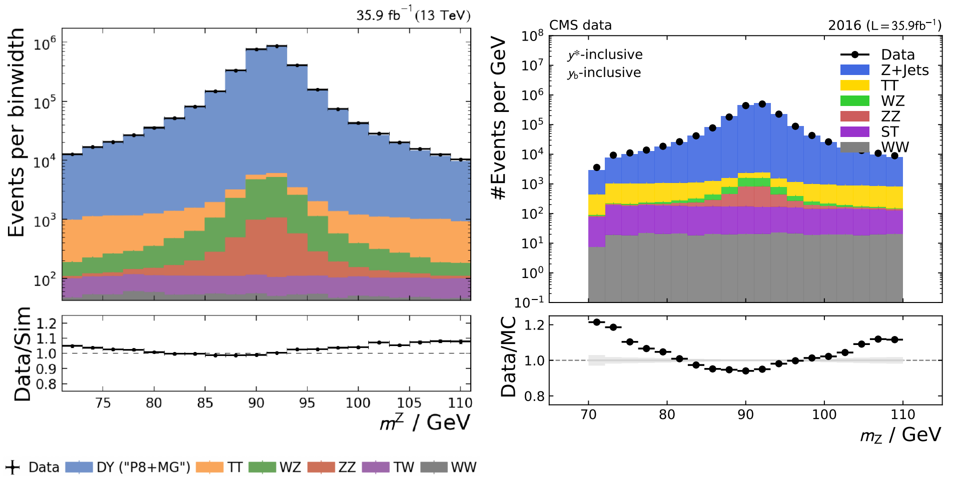
Figure 4.4 shows the distribution of $p_{\mathrm{T}}^{\mathrm{jet1}}$ in recorded data and simulation. In the ratio of the $p_{\mathrm{T}}^{\mathrm{jet1}}$ distribution in recorded data and simulation, the current analysis shows a better correspondence of simulation to data than in the former analysis. This indicates a better understanding of the detector response of jets.

Figures 4.5 and 4.6 show the y^{Z} and y^{jet1} distributions, respectively, on reconstruction level for data and simulation.

The $p_{\mathrm{T}}^{\mathrm{Z}}$ distribution on reconstruction level for data and simulation is shown in Figure 4.7. The ratio between the distribution of data and simulation shows a similar level of understanding in both analyses.

4.3.2 Analysis of 2017 data

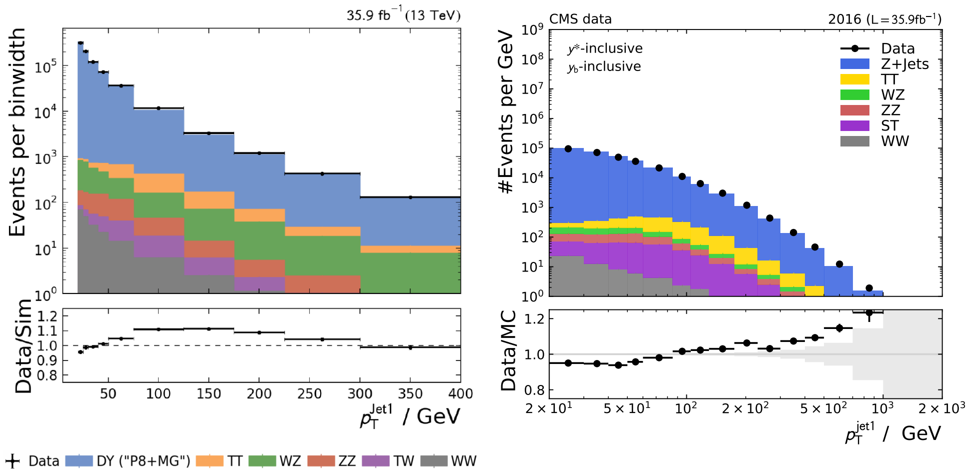
The measurement of the triple-differential Z+jet cross-section with data recorded in 2017 enables a further reduction of the uncertainty compared to a measurement



(a) former analysis by T. Berger [31]

(b) current analysis

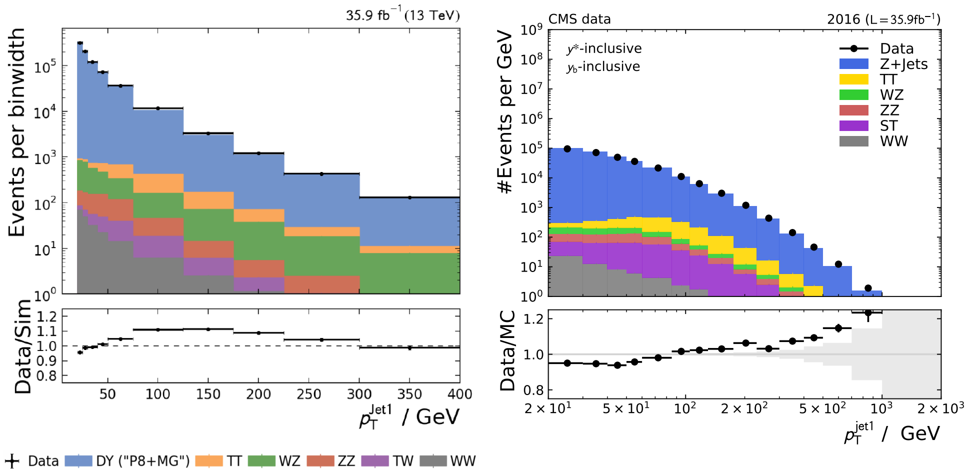
Figure 4.3: Shown is the m_Z distribution in y^*-y_b inclusive phase-space of the measured and simulated events on reconstruction level.



(a) former analysis by T. Berger [31]

(b) current analysis

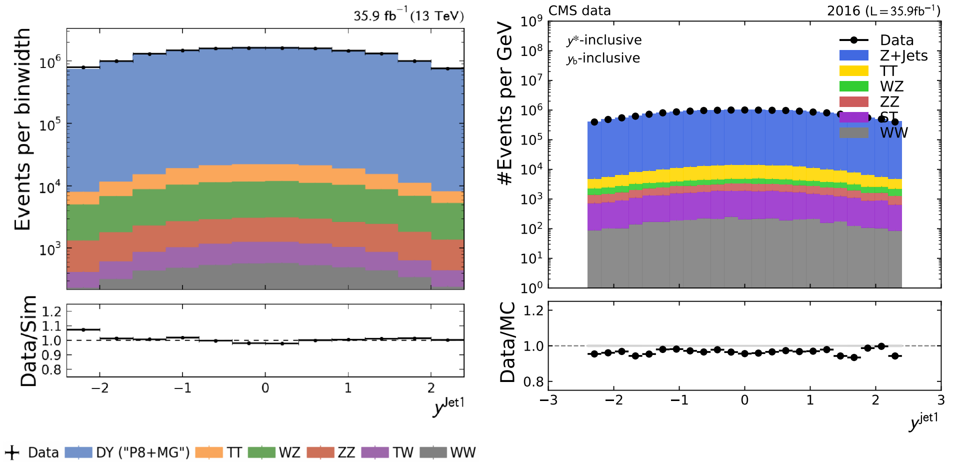
Figure 4.4: Shown is the p_T^{jet1} distribution in y^*-y_b inclusive phase space of the measured and simulated events on reconstruction level.



(a) former analysis by T. Berger [31]

(b) current analysis

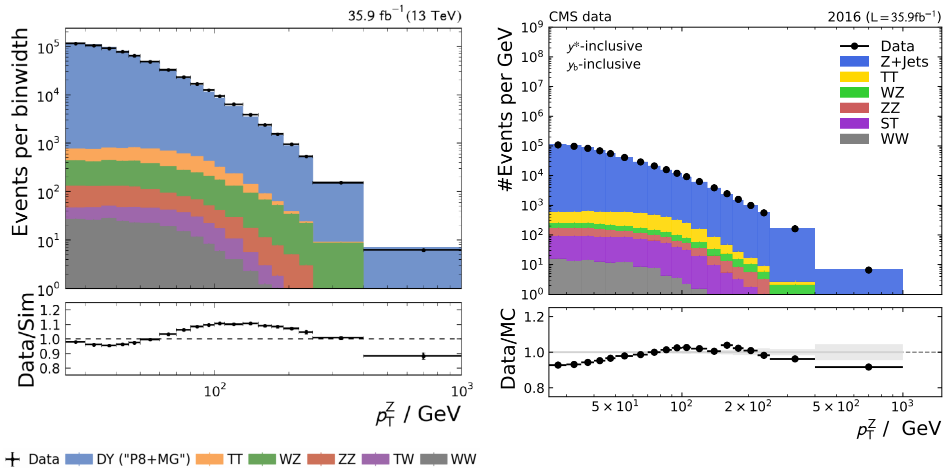
Figure 4.5: Shown is the $y^{\rm Z}$ distribution in y^*-$y_{\rm b}$ inclusive phase space of the measured and simulated events on reconstruction level.



(a) former analysis by T. Berger [31]

(b) current analysis

Figure 4.6: Shown is the $y^{\rm jet1}$ distribution in y^*-$y_{\rm b}$ inclusive phase space of the measured and simulated events on reconstruction level.



(a) former analysis by T. Berger [31]

(b) current analysis

Figure 4.7: Shown is the p_T^Z distribution in y^*-y_b inclusive phase space of the measured and simulated events on reconstruction level.

only relying on data recorded in 2016. The analysis of the data recorded in 2017 by the CMS detector is performed on the single muon datasets. A list of simulated processes and their accuracy is shown in table 4.2 and the used datasets are shown in table A.3. By comparing the Monte Carlo simulations of the signal and background processes with data, it is possible to estimate the background fraction in data.

In addition, it has to be checked whether the detector simulation provides a valid representation of the detector for the data recorded in 2017. For this, the distributions of the simulated events are scaled to the integrated luminosity of the recorded data in 2017 ($L = 41.5$) [44, 45].

Table 4.2: This table shows the Monte Carlo datasets used for signal and background processes.

Process	Event Produced	Accuracy
signal dataset		
Z+Jets	MadGraph5 + Pythia 8	NLO
background datasets		
TTJets	MadGraph5 + Pythia 8	LO
WZ	Pythia 8	LO
ZZ	Pythia 8	LO
WW	Pythia 8	LO
ST t- & tW-channel	Powheg + Pythia 8	NLO

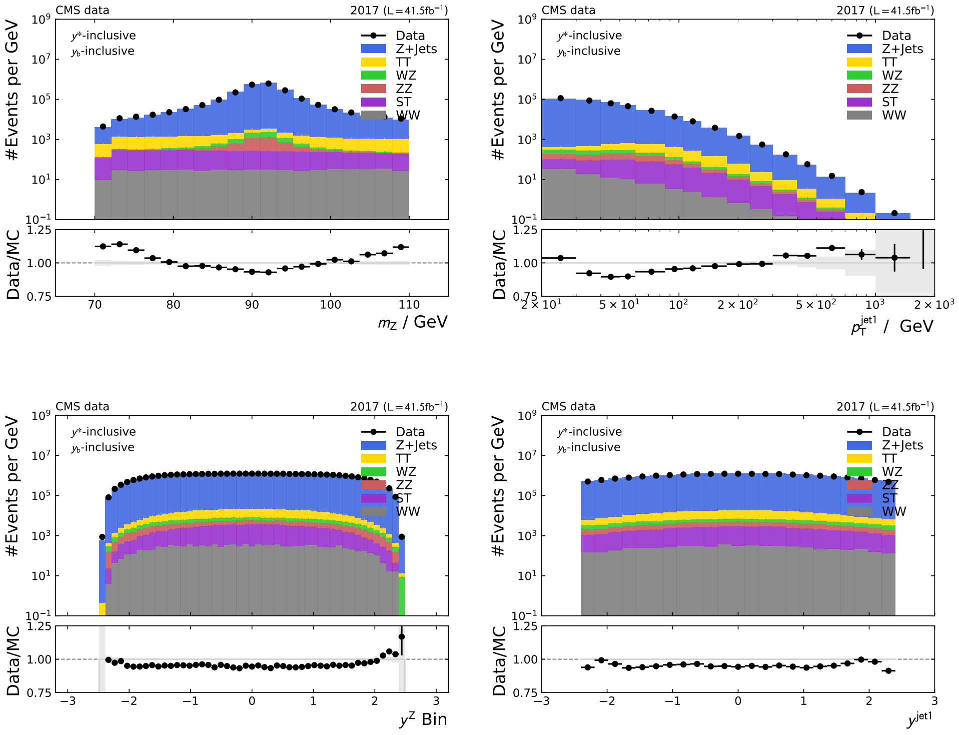


Figure 4.8: From top left to bottom right: m_Z (left) and p_T^Z (right) distribution of data recorded in 2017 with the corresponding simulation on reconstruction level.

The m_Z, p_T^{jet1}, y^Z, and y^{jet1} distributions are shown in Figure 4.8. The ratios between data and simulations in these plots show a higher level of understanding of the detector than in the current analysis of the data recorded in 2016.

Moreover, the p_T^Z distribution shows a good understanding of the detector, see Figure 4.9. The integrated luminosity of the dataset recorded in 2017 is 15% higher than that of the dataset recorded in 2016. The higher integrated luminosity leads to a reduced statistical uncertainty in 2017 compared to 2016, especially in the high-p_T^Z region.

It was shown that the data recorded in 2016 and the data recorded in 2017, as well as the detector are sufficiently well understood. As mentioned before, for the updated analysis of the data recorded in 2016, the muon selection is changed to be comparable with the former analysis. The same muon selection for both years makes it possible to compare the updated analysis of the data recorded in 2016 with the data recorded in 2017. The following results are based on the datasets recorded in 2016 and 2017 with the same selection. Therefore, it should be possible to assume

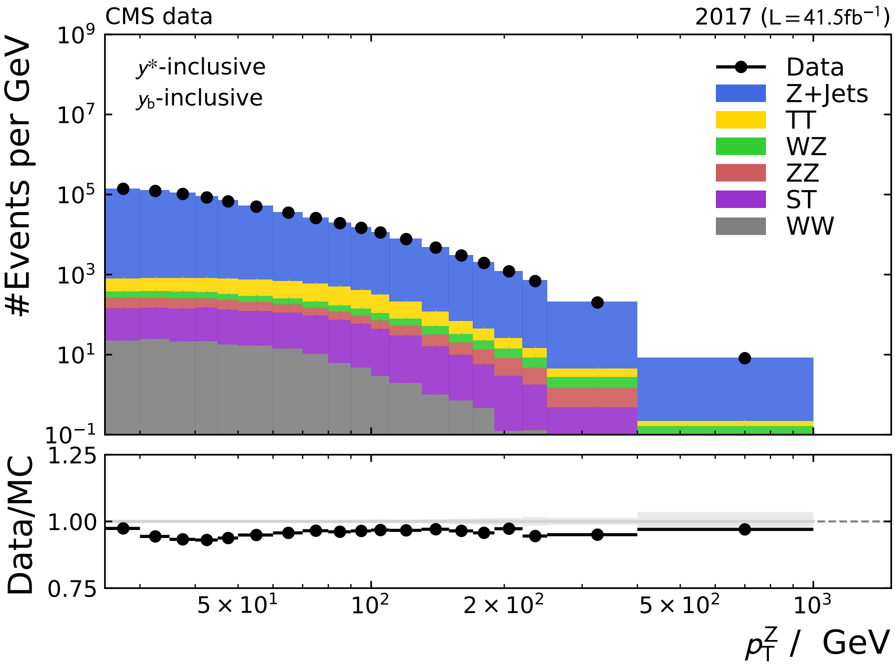


Figure 4.9: Shown is the $p_{\mathrm{T}}^{\mathrm{Z}}$ distribution on reconstruction level in data and simulation for the year 2017.

that both measurements can be combined.

Figure 4.10 shows the ratio between the data recorded in 2017 and 2016 divided by the integrated luminosity of the corresponding datasets. The dataset recorded in 2016 has an uncertainty on the integrated luminosity of 2.5%, while the dataset recorded in 2017 has an uncertainty of 2.3%. It is expected that, both years has the same number of events divided by the integrated luminosity. Under the assumption that the uncertainty of the integrated luminosity is not correlated between the years, a difference of about 4% in absolute numbers divided by the integrated luminosity is estimated. However, it is observed that the overall number of events, after applying all event weights and accounting for the luminosity of each sample, is about 7% higher in the 2017 dataset compared to 2016. This difference of 7% cannot be explained by the other systematic uncertainties (trigger efficiency, L1 trigger bug correction, lepton ID and isolation selection, JEC), as can be seen in Figure 4.10 These uncertainties are described in detail in section 4.6. Such a difference between the datasets recorded in 2016 and 2017 is also observed by another CMS analysis that studies Z+b-jets events for the data recorded in 2016, 2017, and 2018 before the event selection for b-jets. [46] This result will be sent to the CMS collaboration as it is a hint to a residual uncertainty that has to be further studied. Since the difference between the two years is yet to be understood, further analysis is only performed on the data recorded in 2017.

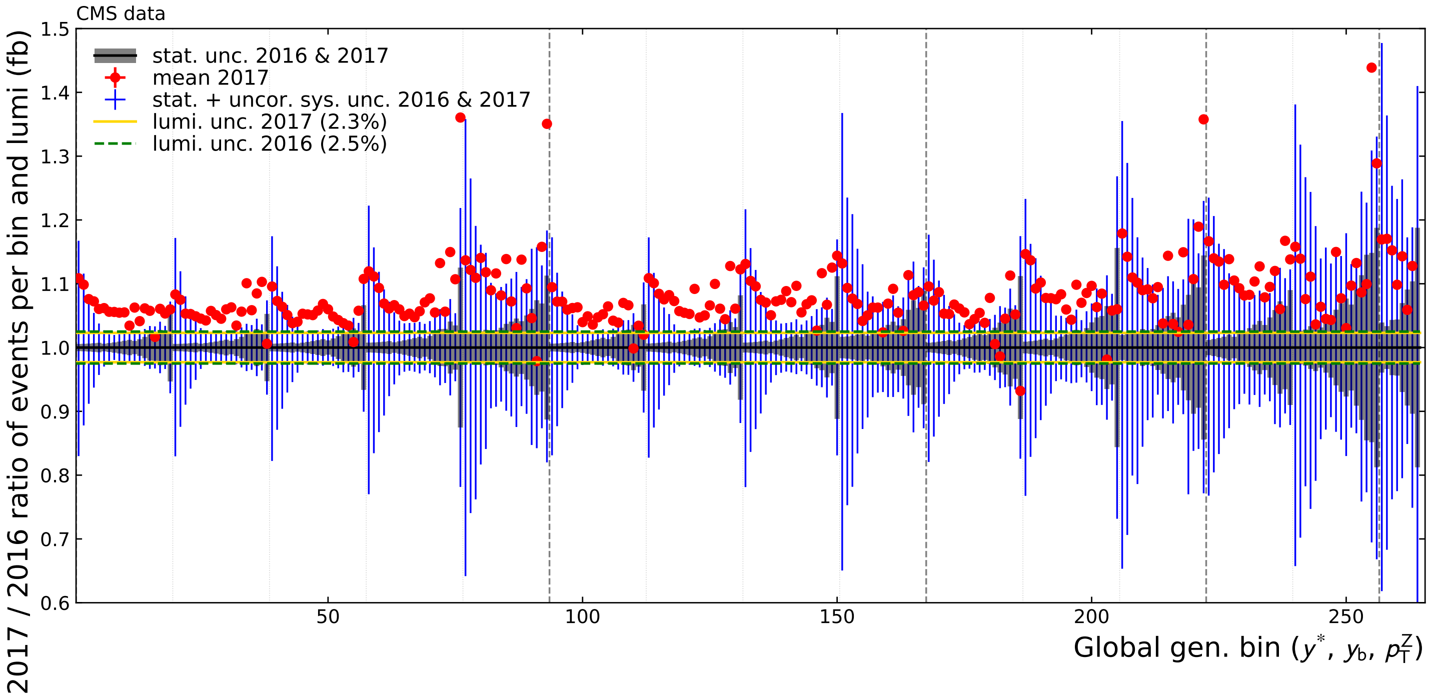


Figure 4.10: Shown is the ratio of 2017 over 2016 data at reconstruction level for all y_b y^* p_T^Z bins divided by the integrated luminosity of the corresponding year. The systematic uncertainty band includes trigger efficiency, L1 trigger bug correction, lepton ID and isolation selection and JEC uncertainties. The luminosity uncertainties are shown for both years. It is assumed that the systematic uncertainties between the two years are independent.

4.4 Unfolding

The recorded data on reconstruction level can not be easily compared with measurements from other detectors or predictions from theory calculations. Therefore, it is necessary to consider detector effects and derive a measurement that is independent of the detector response. This was done by this and the former analysis to measure the triple-differential Z+jet cross-section with data recorded by the CMS detector. For this, the knowledge of the full event simulation is used that also includes the detector simulation. With this knowledge, it is possible to study the change of the true value of an observable with respect to the measured value. The measured value is the value on the reconstruction level. The true value studied in this analysis is the value on the simulation's particle level. These truth level values are comparable with other measurements and theory predictions. The particle level is only known in event simulation and in the following referred to as generator level.

With unfolding, it is possible to transform the distribution of an observable on reconstruction level (h_{reco}) to the corresponding distribution on generator level (h_{gen}). The transformation from h_{reco} to h_{gen} in a discretized phase space is described by the matrix K_{ij}. The matrix represents the probability that an event in bin j on generator level is measured with a value that results in bin i on reconstruction level. Since this matrix describes the migration of events on the generator level to events on the reconstruction level, this matrix is called the migration matrix. Detector inefficiencies can result in misidentified events and are considered during the unfolding procedure. The distribution of such misidentified events, further referred to as fakes, is described by the histogram h_{fake}. That results in the equation:

$$h_{\text{reco}} = K h_{\text{gen}} + h_{\text{fake}} \tag{4.5}$$

Therefore, the events studied in bins of the three variables y_{b} y^* and p_{T}^{Z} are binned according to the serialised binning introduced in section 4.1 and depicted in Figure 4.1. The response matrix shown in Figure 4.11 is filled with events of the full event simulation of the signal process at NLO accuracy.

The migration matrix based on the full simulation is mostly diagonal and has a low condition number. This enables us to use a simple matrix inversion to unfold the data on reconstruction level. The unfolding is performed with the software TUnfold. [47] The migration matrix with 264 × 264 bins is filled with about 187 million events at NLO accuracy. Due to the limited number of events in the full simulation sample, the unfolding can be sensitive to statistical fluctuations. Furthermore, some bins with an insufficient number of events can contain a negative number of events caused by negative event weight due to the nature of the used NLO predictions. This negative weight are a result of interference terms in the matrix elements at NLO accuracy. To prevent this unphysical behavior, bin entries in the migration matrix that are negative are set to zero.

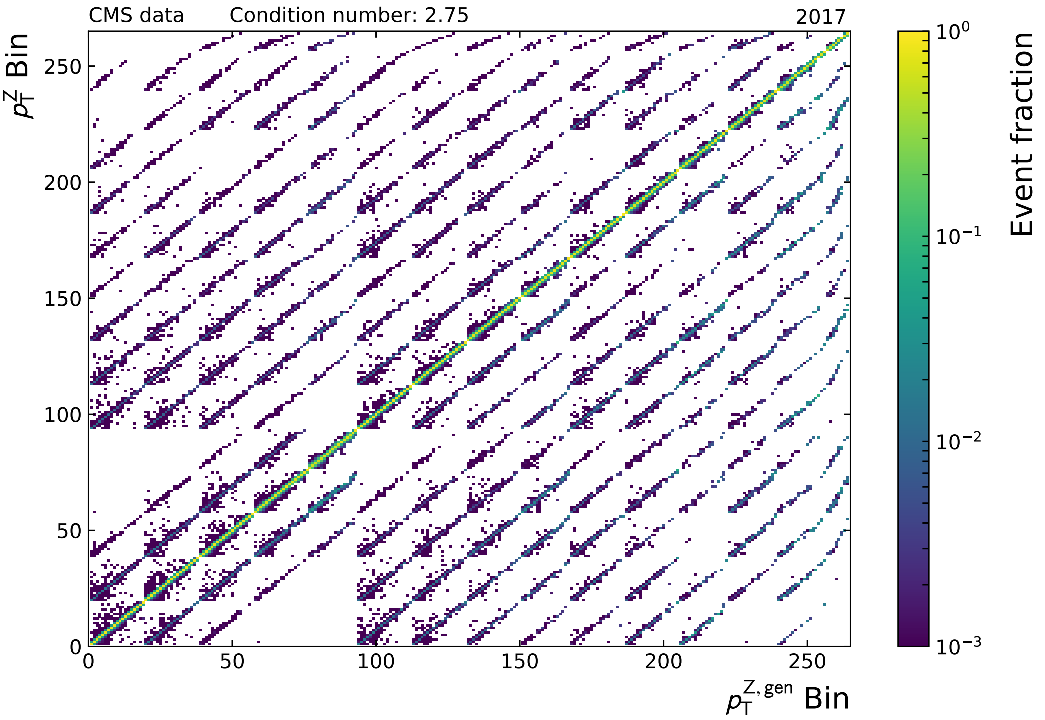


Figure 4.11: Migration matrix based on full simulation of the signal MC sample for the 2017 detector. Due to the fact that most of the entries are on the diagonal and the low condition number, it is possible to unfold the data via matrix inversion. Since the number of events is limited in the full simulation dataset the unfolding method is sensitive to statistical fluctuations in the migration matrix.

To check that the unfolding method is valid, the event distribution on reconstruction level of the full simulation of the signal process is unfolded with the migration matrix derived from the same simulation. If the method is valid, the unfolded distribution is identical to the distribution on generator level. Figure 4.12 shows the distribution on reconstruction level, the unfolded distribution, and the distribution on generator level. As expected, the unfolded distribution is equal to the distribution on generator level. Therefore, the unfolding method is valid.

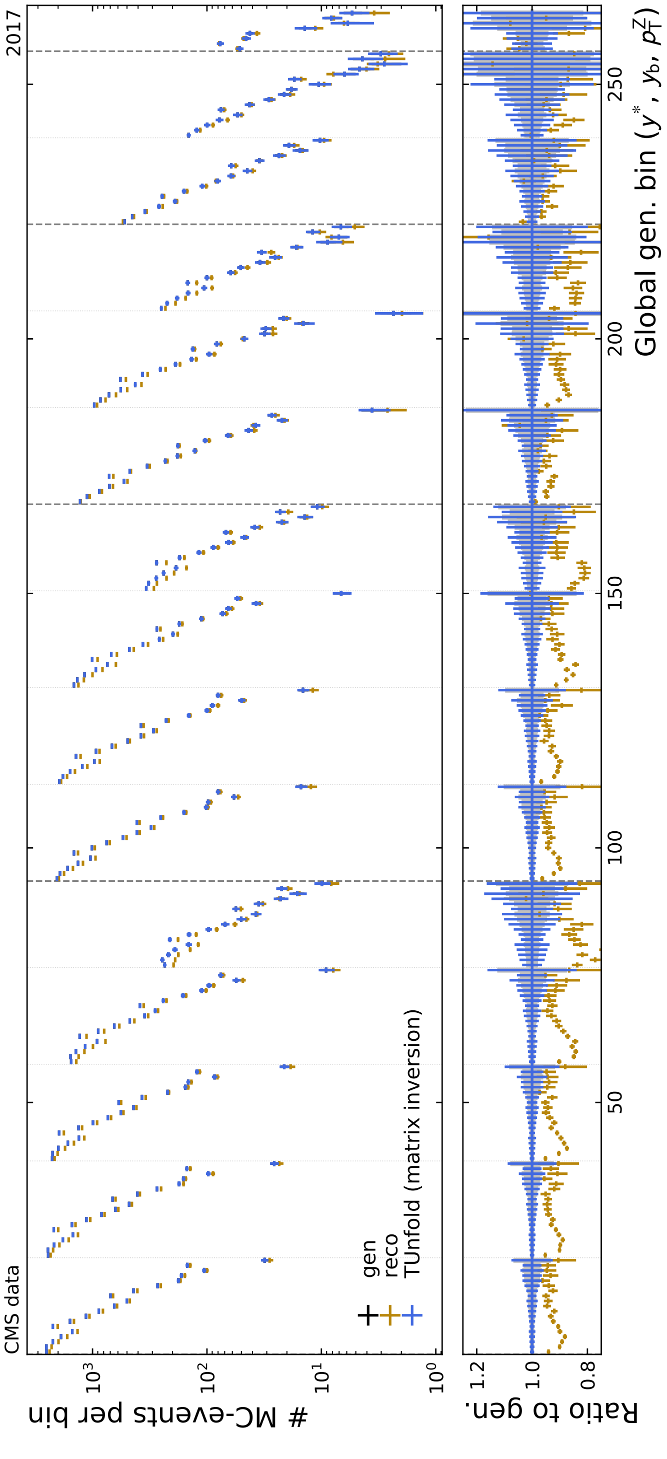


Figure 4.12: Shown are the distribution on reconstruction level, the unfolded distribution, and the distribution on generator level. The black error bars and the gray uncertainty bands in the ratio show the statistical uncertainty of the event distribution on the generator level. The yellow uncertainty bars represent the statistical uncertainty of the event distribution on the reconstruction level. The blue error bars represent the statistical uncertainty of the event distribution on the reconstruction level propagated through the migration matrix.

4.5 Forward Smearing

One option to overcome the limited statistical significance of the full simulation sample is to produce more events. However, the production of full simulated events is computationally very intensive. A more efficient approach used and validated by T. Berger in his analysis [31]. According to this method, referred to as forward smearing, a simple toy Monte Carlo simulation is performed to produce a large number of events with a strongly reduced amount of CPU-power. Therefore, the main systematic effects that contribute to event migrations are modeled in a parametric approach.

For this purpose, the resolution of the variables y^{Z}, y^{jet1}, and $p_{\mathrm{T}}^{\mathrm{Z}}$ are studied in the full simulation and an adequately chosen functional form is fitted to describe their dependence on $p_{\mathrm{T}}^{\mathrm{Z}}$. Another effect is the possibility that the hardest reconstructed jet is not created from the parton with the highest transverse momentum, which needs to be considered. Furthermore, misidentification and detector efficiency are determined for each bin.

The three resolutions for y^{Z}, y^{jet1}, and $p_{\mathrm{T}}^{\mathrm{Z}}$ are estimated for each y_{b} y^* $p_{\mathrm{T}}^{\mathrm{Z}}$ bin. The $p_{\mathrm{T}}^{\mathrm{Z}}$ resolution is defined as:

$$R(p_{\mathrm{T}}^{\mathrm{Z}}) = \frac{p_{\mathrm{T}}^{\mathrm{Z,reco}} - p_{\mathrm{T}}^{\mathrm{Z,gen}}}{p_{\mathrm{T}}^{\mathrm{Z,gen}}} \quad . \tag{4.6}$$

The y^{Z} and y^{jet1} resolutions are defined as

$$R(y^{\mathrm{Z}}) = \left|y^{\mathrm{Z,reco}} - y^{\mathrm{Z,gen}}\right| \tag{4.7}$$

$$R(y^{\mathrm{jet1}}) = \left|y^{\mathrm{jet1,reco}} - y^{\mathrm{jet1,gen}}\right| \quad . \tag{4.8}$$

Thereby, the distributions of $R(p_{\mathrm{T}}^{\mathrm{Z}})$, $R(y^{\mathrm{Z}})$, and $R(y^{\mathrm{jet1}})$ are determined by a 90% truncated root mean square (RMS). The truncation to 90% removes unphysical outliers in the distributions. This value is then corrected to consider the 90% truncation and estimate the RMS for the non-truncated distribution.

Figure 4.13 shows the resolution of y^{Z}, y^{jet1}, and $p_{\mathrm{T}}^{\mathrm{Z}}$ for two y_{b}-y^* bins. The resolution for all y_{b}-y^* bins can be found in Appendix A.6. The truncated RMS underestimates the uncertainties of the resolutions. In most of the y_{b}-y^* bins, the fit of the function has a goodness-of-fit $(\chi^2/N.D.F) > 1.5$. In such y_{b}-y^* bins, the uncertainty of the fit is scaled by $\sqrt{\chi^2/N.D.F}$ to handle the underestimated uncertainty.

To take into account the selection on generator and reconstruction level, the acceptance and fakerate are introduced. The acceptance is defined as:

$$A_{\mathrm{gen.bin}} = \frac{\#(\text{events in gen. bin \& event in any reco. bin})}{\#(\text{events in gen. bin})} \tag{4.9}$$

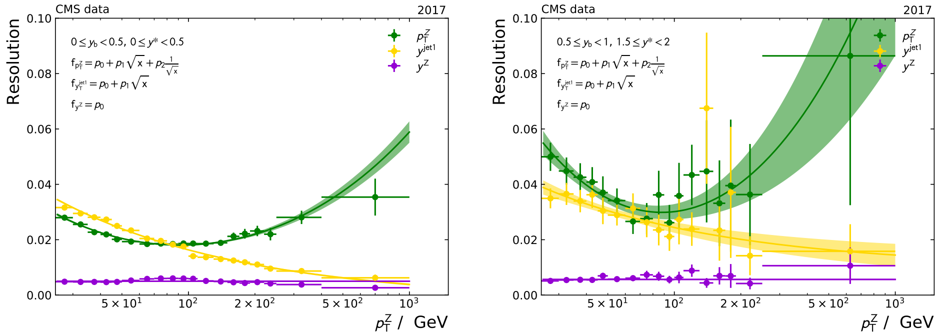


Figure 4.13: Shown is the resolution of y^{Z}, y^{jet1}, and $p_{\mathrm{T}}^{\mathrm{Z}}$ determined from full simulation via truncated RMS. A parametric function is fitted to the data points to get the resolutions with respect to statistical uncertainties of the data point. In some y_{b}-y^* bins the uncertainty of the fit covers the uncertainty of the data point, as can be seen in the left plot. In most of the y_{b}-y^* bins the uncertainty of the fit does not cover the uncertainty of the data points. This is due to the unfitting parametrization, which leads to a $\chi^2/N.D.F > 1.5$, as is the case in the right plot. In such cases the uncertainty of the fit is scaled by $\sqrt{\chi^2/N.D.F}$ to handle the underestimated uncertainty. The plots show the scaled fit uncertainty.

The fakerate is defined as:

$$F_{\mathrm{reco.bin}} = 1 - \frac{\#(\text{events in rec. bin \& event in any gen. bin})}{\#(\text{event in reco. bin})} \tag{4.10}$$

It is assumed that the acceptance and fakerate follow a smooth $p_{\mathrm{T}}^{\mathrm{Z}}$ dependent function. Therefore, for each y_{b}-y^* bin, the acceptance and fakerate are fitted. Figure 4.14 shows two y_{b}-y^* bins, all y_{b}-y^* bins are shown in Appendix A.7.

The reconstructed jet with the highest transverse momentum can originate from different partons on generator level. Usually, the jet with the highest transverse momentum on generator level is within a radius of $R = 0.4$ in the $\eta - \phi$ plane around the jet with the highest transverse momentum on reconstruction level, further referred to as a matched jet. This is the case when the reconstructed jet originates from the parton from the hard interaction process with the highest transverse momentum. In some events, the jet with the highest transverse momentum on the reconstruction level originates from a different parton in the main scattering process but not the parton with the highest transverse momentum. This jet is further referred to as switched. In this case, within a radius of $R = 0.4$ in the $\eta - \phi$ plane around the jet with the highest transverse momentum on reconstruction level, the matching jet on generator level is not the jet with the highest transverse momentum on generator level. The last possibility is that the jet with the highest transverse momentum originates from pileup. In this case, within a radius of $R = 0.4$ in the $\eta - \phi$ plane

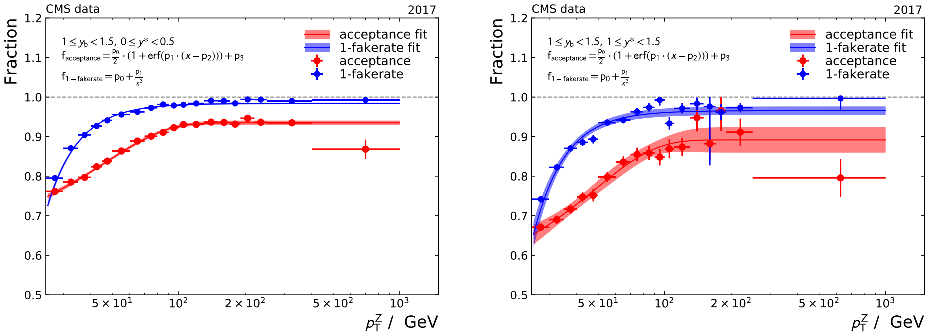


Figure 4.14: The acceptance and fakerate is shown for two y_b-y^* bins. The uncertainties of the data points and the fit are determined with binominal uncertainties by TEfficiency in ROOT [48]. The left plot shows an y_b-y^* bin with a large number of events that results in a low statistical uncertainty of the data points. That results in a low uncertainty on the fitted function. The right plot shows a y_b-y^* bin with a low number of events, which results in a higher fit uncertainty.

around the jet with the highest transverse momentum on reconstruction level, no jet on generator level is found. Figure 4.15 shows the fraction of matched, switched, and pileup jets for two y_b-y^* bins, all the y_b-y^* bins are shown in Appendix A.8. It can be observed that the fraction of switching events is less than 1% and on the order of the uncertainty of the switching probability. Therefore, it is only necessary to fit a parametric function for the switching probability to the fraction of switching events.

The forward smearing for a given number of events to generate is performed as follows: First, random events with y^Z, y^{jet1}, and p_T^Z are generated according to the corresponding distributions and correlations in the full simulation of the signal process. These values obtained are the generator-level quantities used to fill the migration matrix. Based on these values y_b and y^* are calculated for the event.

Second, the switching probability is applied. The probability of the corresponding y_b-y^* bin and p_T^Z value is determined. Based on a randomly generated number, it is checked if an event is matched or switched. If the event is switched, another value of y^{jet1} is chosen for the event according to the y^{jet1} distribution of the signal-process full simulation and is used in the following. Furthermore, values of y_b and y^* are calculated for that event with the new y^{jet1} value. Therefore, the generator-level event information is updated. Otherwise, the values remain unchanged.

Third, the values of y^Z, y^{jet1}, and p_T^Z are smeared according to the resolution for the corresponding y_b-y^*-p_T^Z, which are determined as described above. The smeared values represent the values of an event on the reconstruction level. Based on the smeared values, the y_b and y^* on reconstruction level are calculated.

Fourth, it is checked if the event with its true values determined in the first step

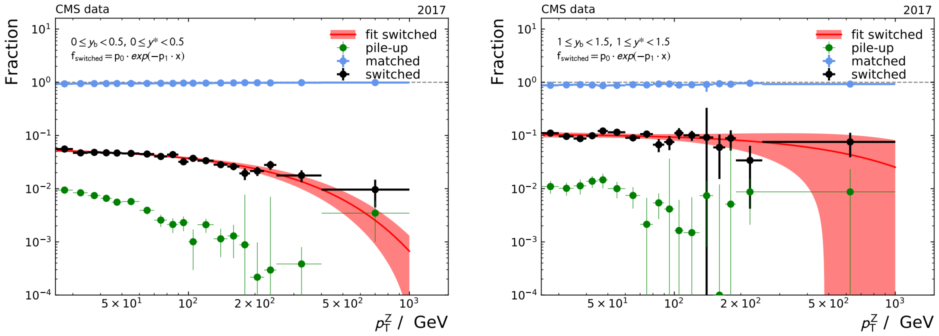


Figure 4.15: Shown is the fraction of matched, switched and pileup events. The probability of a switching event is estimated by a fit of a parametric function to the fraction of switching events. The left plot shows a y_b-y^* bin with a high number of events. In some y_b-y^* bins the number of events is insufficient. That results in a negative fraction of events in some p_T^Z-bins. This behavior is handled by changing the corresponding bin value with the average of its neighboring bins and increasing its uncertainty by a factor of three.

passes the acceptance and the fakerate with the smeared values. Both checks are done with a randomly generated number between zero and if the number is lower than the acceptance / fakerate probability, the event pass. If the event passes both selections, the event is inserted into the migration matrix.

In the forward smearing approach, 10 billion events are produced to fill the migration matrix, which is shown in Figure 4.16. The event fraction in the diagonal bins of this migration matrix is higher than in the migration matrix based on the full simulation shown before (Figure 4.11). The matrix build via forward smearing also has a low condition number that allows to still use matrix inversion.

4.6 Uncertainties

Unfortunately, not all detector and modeling effects are understood with infinite precision. Therefore, uncertainties on these effects are determined that have to be propagated through the analysis to determine the systematic uncertainties of the measured values.

The following systematic uncertainties are considered in this analysis:

Trigger efficiency: The trigger efficiency is determined with a few percent of combined statistical and systematic uncertainty. Both are taken into account to determine the trigger efficiency uncertainty. The uncertainty values per p_T^μ and η^μ are provided by the Muon POG. [32, 33]

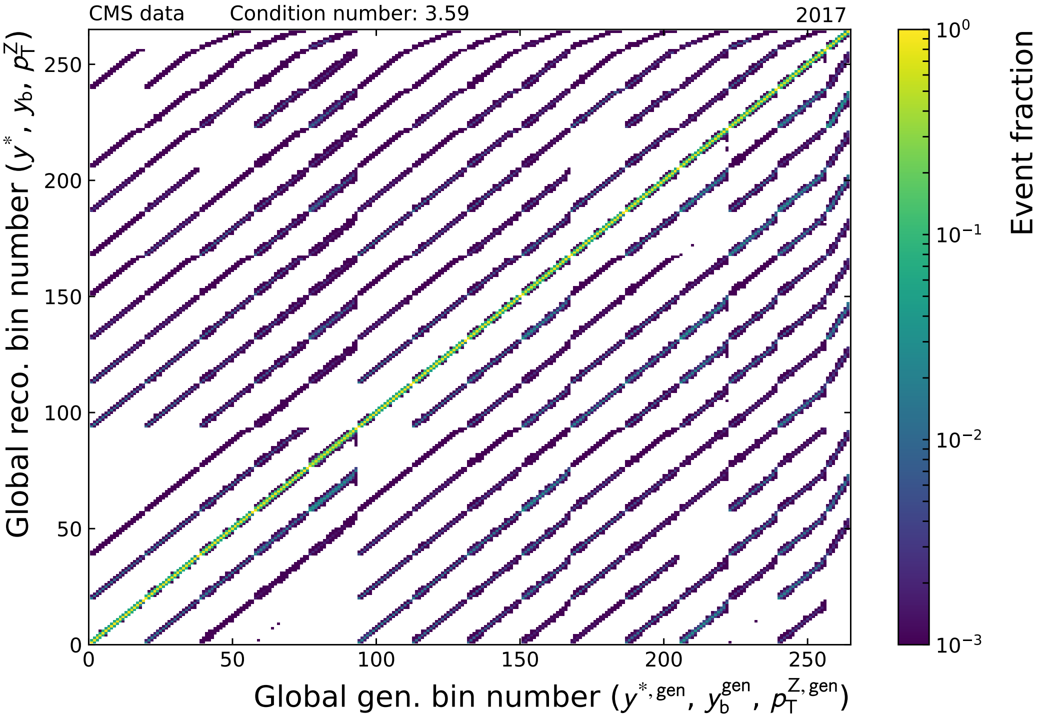


Figure 4.16: Migration matrix based on forward smeared MC events from the signal process Monte Carlo dataset for the 2017 detector. With forward smearing it is possible to produce a number of 10 billion events to fill the migration matrix. Consequently, statistical fluctuations are reduced. Due to the low condition number, it is possible to perform a simple matrix inversion to unfold the data.

L1 trigger bug correction: The uncertainty of the correction of the L1 trigger bug is provided by CMS and depends on the transverse momentum and η of the photons and jets detected in the ECAL.

lepton ID and lepton isolation: The selection efficiency of the lepton ID and lepton isolation has statistical and systematic uncertainties. These are dependent of p_T^μ and η^μ. The uncertainties on the scaling factors used for muons are provided by the Muon POG. [32, 33]

JEC: The JEC uncertainty is dependent on p_T^{jet1} and η^{jet}. The uncertainty values are provided by CMS for the different p_T^{jet1} and η^{jet} values. [38]

Unfolding: The statistical uncertainty due to the limited amount of events in the migration matrix is drastically reduced via the forward smearing method. However, the resolutions, switching probability, acceptance, and fakerate determined in the forward smearing method include statistical uncertainties. They are smaller than the statistical uncertainties of the values based on the full simulation, but not negligible. The systematic uncertainty of the unfolding is determined by unfolding the recorded data on reconstruction level with 100 different migration matrices, obtained by varying the resolutions, switching probabilities, acceptances, and fakerates according to their uncertainties. The RMS of the the unfolded data per y_{b}-y^{*}-$p_{\mathrm{T}}^{\mathrm{Z}}$ bin is used to determine the uncertainty of the unfolding method.

The uncertainty of the JER is not taken into account in this analysis. The former analysis performed by T. Berger [31] showed that the JER uncertainty is negligible compared to the other uncertainties studied in this analysis. The JEC uncertainties can be further reduced by a better understanding of the detector and the improvement of reconstruction algorithms. This is expected to happen in a reconstruction of the full Run 2 data. The statistical uncertainty of the measurement of the triple-differential Z+jet cross-section can be reduced by combining the data recorded over several years.

Figure 4.17 shows the uncertainties of the unfolded data recorded in 2017 for two y_{b}-y^{*} bins. The uncertainties for all y_{b}-y^{*} bins are shown in Appendix A.9. In the low-$p_{\mathrm{T}}^{\mathrm{Z}}$ region, the JEC uncertainty is dominant, while in the high-$p_{\mathrm{T}}^{\mathrm{Z}}$ region the statistical uncertainty in most of the y_{b}-y^{*} bins is dominant. At high y^{*}, the dominant uncertainty in the high $p_{\mathrm{T}}^{\mathrm{Z}}$ region is the unfolding uncertainty.

4.7 Unfolded 2017 Data

To compare the measurement of the triple-differential Z+Jet cross-section with other measurements or theory predictions, the recorded data has to be corrected for detector effects. Because the unfolding is only designed to unfold the signal distribution, the background contribution is removed from the recorded data. To determine the background contribution, the distributions of the simulated background processes are scaled to the integrated luminosity of the recorded dataset. In this analysis, the correction of detector effects in data is performed via unfolding on recorded data distribution after removing the background contribution.

For the theory predictions, the event distributions on generator level of an NLO and LO simulation are used. The NLO simulation is produced via MadGraph Z+0,1,2 jets at NLO accuracy merged with the FXFX method interfaced to Pythia 8 while the LO simulation is produced via MadGraph Z+0,1,2,3,4 jets merged with the MLM method interfaced to Pythia 8. Figure 4.18 shows the unfolded data with the

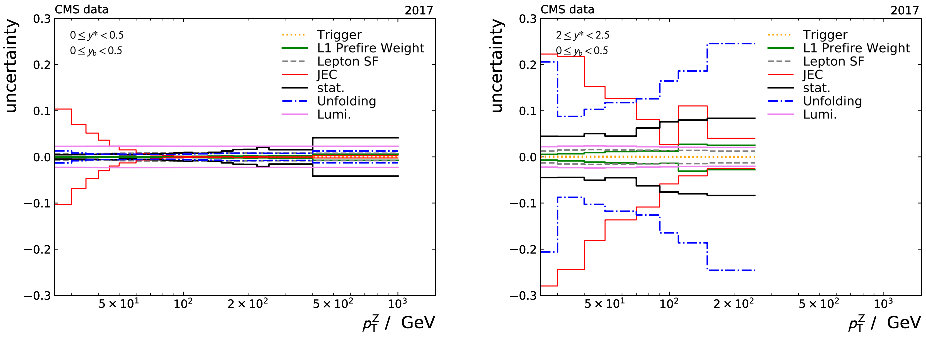


Figure 4.17: The statistical and systematic uncertainties of the unfolded data are shown in two y_b-y^* bins. In the low p_T^Z region the JEC uncertainty is dominant. In most of the y_b-y^* bins in the high p_T^Z region the statistical uncertainty in most of the y_b-y^* bins is dominant, this is shown in the left plot. At high y^*, the dominant uncertainty in the high p_T^Z region is the unfolding uncertainty. This is shown in the right plot.

statistical and systematic uncertainty and the two theory predictions in two y_b-y^* bins.

The measured triple-differential Z+Jet cross-section is higher than the theory prediction at LO accuracy. The difference between the measurement and the theory prediction at LO is not covered by the statistical and systematic uncertainties of the measurement. However, the measured triple-differential cross-sections are within the uncertainties almost comparable to the theory prediction at NLO accuracy. The differences between measurement and theory predictions are expected to be further reduced with predictions at NNLO accuracy. The differences between the measured triple-differential cross-section and theory predictions indicate necessary corrections on PDFs. For the corrections on the PDFs, theory predictions at NNLO accuracy are needed to reduce the uncertainty to the same level as observed in the measurement. Due to a changed phase space between the former analysis by T. Berger and this analysis and the format in which the theory predictions are provided, the theory predictions used in the former analysis can not be used. Therefore, new theory predictions are required and are currently in production.

4.8 Computing Resource Requirements

All the described steps are performed on the computing resources at the Institute of Experimental Particle Physics (ETP), which are discussed in section 6.1.1. The datasets used in this analysis are produced by CMS and are used for various other analyses. For the production of simulation datasets, a very large amount of comput-

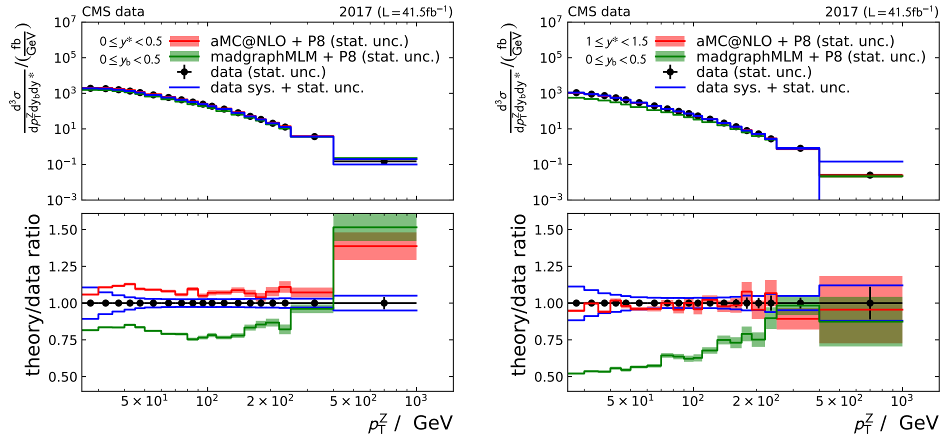


Figure 4.18: The statistical and systematic uncertainties of the unfolded data as well as theory predictions are shown in two y_{b}-y^* bins. The right plot shows an y_{b}-y^* bin with a good agreement between measurement and theory prediction with statistical uncertainty at NLO accuracy. The left plot is an example for a region in which the difference between theory and data indicates necessary corrections to the PDFs.

ing resources is used. [49]

All datasets are converted into another file format that reduces the amount of stored information per event. The initial datasets have a size of about 35 TB and are reduced to about 8.6 TB due to the selection and conversion. For one conversion run of the datasets used in the analysis of the data recorded in 2017 about 50 000 CPU hours are needed. This conversion step has in total been performed four times due to improved and updated detector simulations and improvements in the analysis during its initial stages.

The final event analysis needs about 3000 CPU hours runtime to apply it on data and simulation datasets. This analysis step has overall been performed eight times due to updated datasets, analysis improvements and studies of systematic effects in the analysis. The size of the produced output files is approximately 500 GB.

The last processing step with the final event selection and creation of histograms and plots has been performed about 30 times due to updated datasets, analysis improvements and studies of systematic effects. Each run takes about 100 CPU-hours. This is a rough average estimate due to the fact that some parts are run more often than others. The final outputs are files containing histograms and plots.

For the forward smearing procedure described in section 4.5, additional CPU hours are required to produce the required simulations to fill the migration matrix for unfolding. The production of 10 billion events via forward smearing takes about 400

CPU-hours. This has also been repeated about 100 times during development and study of systematic uncertainties due to unfolding, as mentioned in section 4.6.

Chapter 5

Computing in Scientific Communities

Nowadays, computers are indispensable for scientific research, such as solid state-physics, astrophysics, or HEP. For example, computers are vital to analyze data, simulate physics, or reconstruct data from detector signals to physical observables. For a better understanding of systems in more detail and on a larger scale than currently, more data and computing power is required.

End-user devices, such as desktop-PCs and laptops, are often used for office work, e.g. reading and writing e-mails, creating presentations, etc. It is also possible to develop and test software on these computers. Some current desktop-PCs and laptops have enough computing power to run simple simulations or analyses. They are typically used for users as an entry point to other resources.

For more significant and complex tasks, e.g. simulations or analyses, dedicated machines are usually used. These machines provide more storage, memory, and computing power than end-user devices, enabling more and larger computing tasks.

If several of such machines are needed, they are often combined into a cluster. As resource management gets more complex in this case, a batch system is usually used to distribute and manage the computing tasks such as simulation and analyses, so-called jobs, to dedicated machines, so-called worker nodes.

Clusters can be categorized into High-Performance Computing (HPC) and High-Throughput Computing (HTC) systems. At HPC clusters, the batch system, hardware, and software stack are designed to process jobs that run on several worker nodes. The distribution of jobs to several worker nodes is necessary due to the limited amount of computing resources one node provides. One example of such a complex task is a weather forecast, where the area to be simulated is split into subareas. Each of these subareas is processed by one job instance. For such complex simulations or analyses, the single job instances must communicate among each other to propagate their changes to other instances. To avoid unnecessary waiting time during the communication, HPC worker nodes provide a low latency network connection such as InfiniBand [50].

In contrast to HPC clusters, HTC clusters do not focus on low latency networks and the scheduling of several worker nodes. HTC clusters provide resources for jobs that run independently of each other, as the underlying tasks can be split into several parts and can therefore be trivially parallelized. An example of such a job is the reconstruction of HEP events. Every single event can be reconstructed independently of the others. As a consequence, the network connection inside a HTC cluster is more optimized on throughput than on low latency.

Some scientific communities, e.g., that of structural biology studying the structure of macromolecules such as DNA and proteins, use several clusters [51]. The usage of several clusters can be organized as a so-called *Grid* [52]. The interaction with a Grid enables access to all clusters via one entry point instead of one entry point per cluster, often referred to as a Grid site. A Grid simplifies the usage and organization of several clusters for the communities.

For efficient usage on a continental or global scale with dozens of Grid sites, it is necessary that all systems inside a Grid can run the experiment / scientific community software. Therefore, experiments and communities support and verify their software for at least one defined software stack. This defined software stack is then provided by the sites. However, non-Grid resources, e.g. HPC clusters, do not provide the needed software stack for non-designated user groups. Virtualization and container technologies enable users and communities to create and save their needed software environment on one machine and use the same software environment on other machines or an entire cluster. As a result, cluster administrators have to install only one container software instead of all the various software-stacks the users require. The virtualization and container technologies also enable to provide the same software environment on several clusters, that will be discussed in chapter 5.3.

Special Grid-file-transfer protocols, such as XRootD [53] and FTP with Grid authentication (GridFTP) [54], enable to read and write data from remote clusters. However, the additional traffic causes challenges which will be discussed in chapter 7.

Due to increasing demand in scientific computing, scientific communities have to look for more computing resources in addition to resources dedicated to them. One example is Folding@home, a system that enables to run protein unfolding on temporarily available clients. Therefore, everyone with a common desktop computer can provide free computing resources to Folding@home. [55] In April 2020, during the Covid-19 pandemic, the computing power of 2.4mil TFlops was available via Folding@home due to the support of many people and organizations [56]. This is about six times the computing power of the most powerful HPC cluster (Summit) at the time [57]. The following chapters discuss concepts and software to integrate temporarily available resources in a heterogeneous resource pool. The concepts and software discussed are exemplarily discussed for the HEP community. However, they can also be applied to other scientific communities that have similar computing requirements.

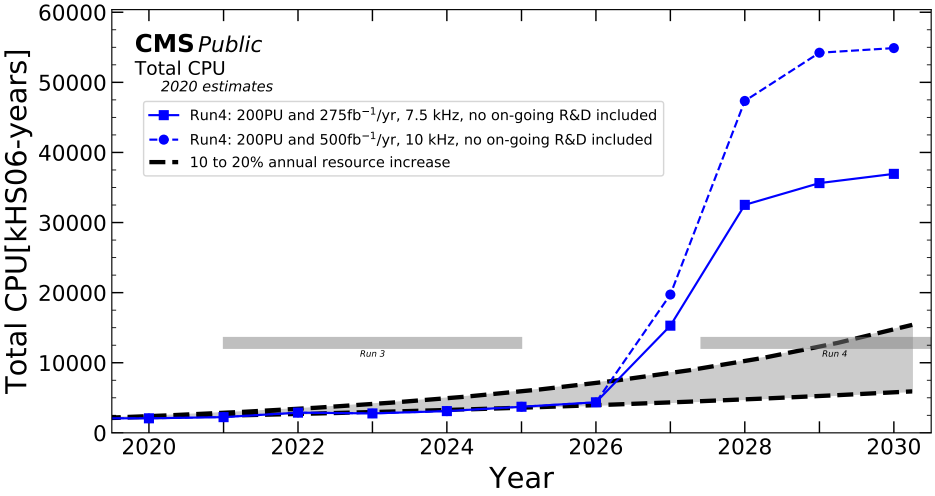


Figure 5.1: Estimated computing power available to the CMS collaboration assuming a flat budget and a performance increase by 10-20% per year due to technological improvement. This is compared to different models of the needed computing resources. [59]

5.1 Computing in HEP

One of the biggest scientific communities is the HEP community. Current HEP detectors, such as the CMS detector, produce several petabytes of data per year that enable physicists to find rare processes and reduce statistical uncertainties for precision measurements. To process and analyze this data, a considerable amount of computing power, in the order of several hundred thousand CPU cores, is needed. For the LHC collaborations, the computing power is mainly provided by the global infrastructure of the Worldwide LHC Computing Grid (WLCG), see section 5.2.

During future run periods of the LHC, especially after the major upgrade to the HL-LHC, an immense amount of computing resources is required [58]. Assuming a continuous funding at the current level (flat budget), it is challenging for the WLCG to provide the requested computing power.

Based on the planned performance of the HL-LHC, various estimations of the computing power required for the processing of the produced amount of data are made. Some estimations can be seen in Figure 5.1. Several approaches are being studied to prepare the collaborations for the HL-LHC computing challenge, such as adaptations of the computing model, development of faster event simulations, and software optimizations [60]. One adaptation of the computing model is the usage of additional resources not directly provided by Grid sites. In addition to Grid sites, resource providers such as HPC centers and cloud providers can be utilized by HEP. However, the usage of these additional resources needs some adaptions and preparation. [61, 62]

All the Grid sites within the WLCG provide computing resources that fulfill the software and hardware requirements for HEP jobs. For WLCG sites, these requirements are CPUs based on x86-64 architecture with at least 2 GB system memory and at least 20 GB scratch space per CPU core. Furthermore, worker nodes need an Ethernet connection to read and write data from and to Grid storage. [63–65]

Most HEP collaborations' software environments are based on an operating system (OS) similar to RedHat Enterprise Linux (RHEL) such as CentOS or Scientific Linux [66]. Due to the end of support for RHEL 6 and similar OSs, most collaborations updated to RHEL 7 until 2020 [67]. Additionally, Cern Virtual Machine File System (CVMFS), which provides current experiment software on a large scale, is necessary for the HEP software environment.

Computing resources that are not dedicated to or supporting the HEP community, so-called opportunistic resources, usually do not provide the required software stack. In the following, some example showcases present the challenges to make these computing resources available for HEP, specifically for the ETP computing infrastructure.

One of the first computing resources dynamically included in the ETP computing infrastructure was the HPC cluster `bwForCluster NEMO` at Freiburg. The cluster is specifically designed for the neuroscience, elementary particle physics, and microsystems engineering communities in Baden-Württemberg, Germany. Therefore, the cluster provides the required software environment for all communities. `BwForCluster NEMO` worker nodes use the OS CentOS 7, which was, at the start of the `bwForCluster NEMO` cluster, a common OS. It supports low latency connections between the worker nodes, which some workflows of the neuroscience and microsystems engineering communities require. At the start of the `bwForCluster NEMO`, the HEP physics software was not verified and supported for CentOS 7. However, it is possible to provide the needed software environment via virtualization. At the `bwForCluster NEMO` cluster, it is possible to get virtual machines running on HPC worker nodes. The virtual machine integrates into the batch system of the ETP. This enables to use these resources similar to worker nodes at the ETP. Via a batch job at the `bwForCluster NEMO` cluster, it is possible to request a virtual machine. The batch job triggers that a virtual machine starts on the same worker node via the virtual machine infrastructure OpenStack [68]. This allows that resources for all three scientific communities can be managed and accounted through the same batch system. However, managing an additional infrastructure to provide virtual machines complicates the administration. Additionally, running jobs inside virtual machines results in a resource overhead which reduces the usable computing power. Therefore, most clusters do not support virtualisation [69].

Container technology is a more lightweight technique to provide a software environment than virtual machines. The first resources made available through container technology at ETP are the desktop-PCs. Modern Desktop-PCs typically provide sufficient computing capacity to also run batch jobs parallel to the regular desktop usage. However, the primary purpose of desktop-PCs is different from running batch jobs. For example, desktop-PCs at ETP have the Linux distribution Ubuntu as OS that is more suitable for desktop usage than an OS used for clusters. The HEP software is not verified for the used OS on the desktop-PCs at the ETP. The batch system `HTCondor`, which the ETP uses, enables running batch jobs in containers via the container software `docker` [70]. As a result, it is possible to support another software environment for each job on the same infrastructure.

Furthermore, `HTCondor` also supports suspending batch jobs according to given metrics such as system load or current daytime. Thereby, the desktop-PCs' free computing resources can be used by the batch system, while desktop-PC usage is preferred. [71]

The container technology is also an option for HPC clusters. However, most of the HPC clusters, such as the ForHLR II at KIT, do not provide `docker` due to security concerns. An alternative to `docker` is the container software *singularity* [72]. It has been developed specifically for HPC clusters. It provides less functionality than `docker`, e.g., no network monitoring, but supports the necessary features to enable the HEP software environment on an HPC cluster while satisfying security concerns of cluster operators. This has been tested at the ForHLR II and is now in production.

With virtualization and container technologies, it is possible to provide a specific software environment for different resource providers. Furthermore, all these resources can be integrated into one batch system via the `drone` concept, discussed in chapter 5.3. The management of these resources in a heterogeneous environment requires a new concept for efficient usage of these resources, see chapter 5.4. These concepts enable usage of a wide variety of computing resources to dynamically extend current systems and use existing resources more efficiently.

5.2 Global Infrastructure: WLCG

The WLCG is a federation of about 170 data and computing centers, so-called Grid sites, in 42 countries [73]. This federation provides storage and computing power to the LHC collaborations, such as CMS. The original design of the WLCG introduced tiers of Grid sites [74]. The Tier0 site is the entry point for the recorded event data from the detectors and is located at CERN. The recorded events, so-called raw data, are archived, and a first reconstruction is performed. A copy of the raw data is distributed among all Tier1 centers for the primary purpose of backup and further data processing. Furthermore, Tier1 sites, such as GridKa [75], store datasets of raw data and simulated events on short- and long-term storage systems. In addition, further event reconstructions, event simulations, and end-user analyses are performed at Tier1 sites. Tier2 sites do not provide long-term storage. They are focused on event simulations as well as end-user analyses and provide an online storage system for that purpose. Additional to the Tier2 sites, end-user analyses run on university resources, often referred to as Tier3 resources. However, these resources are not officially part of the WLCG and primarily used by local users.

5.3 Integration of resources and software provisioning

To provide the huge amount of computing power, such as the hundreds of thousands of CPU cores for HEP collaborations, in an automated and efficient way, *batch systems* such as `HTCondor` are used. Batch systems are resource management software distributed across many machines, so-called nodes. Machines that provide computing power are called worker nodes. Users have access to worker nodes by submitting non-interactive executables with corresponding attributes, such as the needed runtime or number of needed CPU cores, as so-called jobs to the batch system. A service of the batch system schedules these jobs to worker nodes by matching the requirements specified by the jobs and worker nodes.

HEP collaborations make their computing resources available to their members via a batch system. As mentioned before, each WLCG site provides computing resources for HEP collaborations. Yet, instead of submitting batch jobs directly to some Grid sites, collaboration members submit those batch jobs to a global batch system instance of their collaboration. Therefore, the collaborations include resources of the Grid sites into their batch system instance. This enables users to run their batch jobs at any resource available to the collaboration as a single point of entry. The usage of the resources is transparent to the user, as the batch system of the collaboration introduces a layer of abstraction from the resources of the Grid sites. Such a global batch system instance that integrates multiple resources is referred to as overlay batch system (OBS).

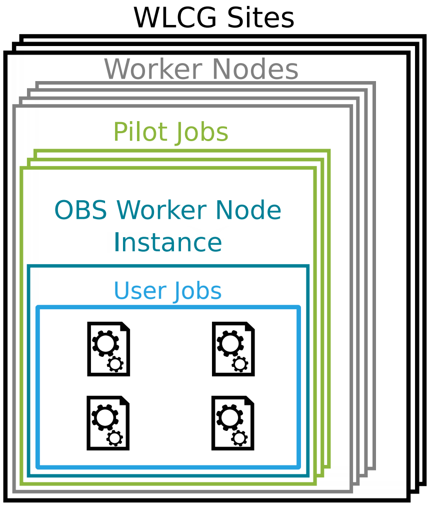


Figure 5.2: Each WLCG site has several worker nodes. Inside each worker node, multiple pilot jobs can run. Inside a pilot job, an OBS worker node instance of a collaboration is running, which starts and manages the user jobs.

The integration of computing resources of a Grid site is done by a placeholder job, a so-called pilot job, running on a worker node of that Grid site. Inside the pilot job runs a worker node instance of the OBS. Now, it is possible for the OBS to schedule batch jobs to the worker node instance running at a Grid site. The structure of worker node instances and around is shown in Figure 5.2.

However, the pilot concept has limitations. First, pilot jobs are designed to run as batch jobs. This works fine for Grid sites because they provide their computing resources via batch systems. However, other providers, such as commercial cloud providers, offer their resources as virtual machines or containers.

Second, the user jobs need a verified software environment for accessing and analyzing HEP data. In the pilot concept, the pilot does not foresee to provide the needed software environment by itself. It is assumed that the software environment for accessing and analyzing HEP data is available on the integrated resource by default. Some collaborations extended the pilot concept to provide verified software environments with containers [76]. However, this extension was not designed for opportunistic resources, only for Grid sites. These Grid pilots are able to provide different verified software environments on Grid sites. The support of several software environments enables a smooth transition from RedHat 6 based OSs to RedHat 7 based OSs for the uses while Grid sites updated their worker nodes.

Third, Grid sites provide special entry points for Grid jobs, so-called *Computing Elements*. Other computing resource providers, such as commercial cloud providers and HPC centers, do not use *Computing Elements*.

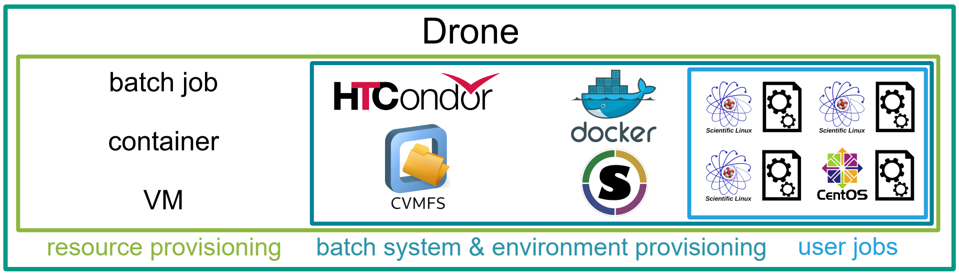


Figure 5.3: The components of a drone processing user jobs. The green box represents the resource provisioning component. The dark blue box represents the environment provisioning. Inside the provided environment, the user jobs run.

For opportunistic resources, a more generalized concept is needed. The first two points, usage of resources not provided via a batch job and providing of the software environment, are handled by the `drone` concept which will be discussed in the following. The third point, requesting resource providers outside the WLCG, will be discussed in chapter 5.4. A `drone`, similar to a pilot job, is a placeholder at the resource provider that allocates resources into the OBS. However, it can be a batch job, virtual machine, or a container, depending on how the provider offers computing resources. This enables, in contrast to the pilot concept, the usage of resources also from commercial cloud providers in the form of a container or virtual machine. Similar to the pilot job, a worker node instance of the OBS runs inside a `drone`. Pilot jobs expect that the software environment for HEP workflows is already provided. However, `drone`s also provide the required environment for HEP workflows or pilots themselves. These two aspects of the `drone` concept result in two components of a `drone`, which are shown in Figure 5.3.

The first component is the *resource provisioning* component. This component defines how the computing resources are provided, e.g., as a batch job on bare metal, as a container, or as a virtual machine. Inside the resource provisioning component, the second component, the *batch system and environment provisioning* component, is located to provide the environment required by the jobs. The possible combinations of provided resource and software environment provisioning for pilot jobs and drones inside that resource is shown in Table 5.1. The resource provisioning component, e.g., a virtual machine or a container, also runs a worker node process of the OBS. The batch job of the OBS runs inside the provided software environment. It is also possible to run a `drone`, as well as a pilot, inside a `drone`. This enables the expansion of WLCG sites with `drone`s, see Chapter 6.2. Although this concept is designed for HEP, other scientific communities can use it as well. The concept was used, for example, to provide resources at KIT to the microbiology community via the folding@home and rosetta@home project, aiming for a better understanding of

environment resource	native	container	VM
batch job	pilot & drone	pilot & drone	drone
container	drone	drone	drone
VM	drone	drone	drone

Table 5.1: Possible combinations of software environment and resource provisioning for drone and pilot concepts. While the pilots are designed as batch jobs, `drone`s follow a multifaceted approach, allowing a wider range of combinations.

the SARS-CoV-19 virus [77].

5.4 Resource Management

Using the `drone` concept, it is possible to integrate resources into an OBS. However, it is necessary to provide resources for a `drone`, e.g. through a batch job or via an API call to a virtual machine. If resources from multiple providers should be integrated, it is necessary to decide how many resources should be requested at which provider.

To address this issue, several resource managers were developed, such as the *glideinWMS* (*glideinWMS*) `factory` [78], the *Cloudscheduler V2* [79], and Responsive On-Demand Cloud-enabled Deployment (*ROCED*) [80]. These resource managers decide how many resources of which type they have to request based on the current number of jobs in the job queue of the OBS. However, each of these resource managers was designed for a different purpose. In the following, their decision process and their limitations are discussed.

Every sizeable HEP collaboration uses an instance of a workload management system (WMS), such as *glideinWMS* at CMS. The central components of a WMS are an OBS and a resource manager. The resource manager has to provide resources that are preferred by the job scheduler of the OBS. As the WLCG sites provide comparable computing resources, the resource pool is rather homogeneous. However, the WLCG sites differ in the stored datasets. To ensure minimal job efficiency limitations due to insufficient network bandwidth, the WMS of the collaboration prefers to start jobs at WLCG sites, that have corresponding datasets. The consideration of the datasets results in a heterogeneity of the resource pool in the scheduling process. Therefore, the resource scheduler should provide that type of resources that the job scheduler needs. Otherwise, resources can only be used inefficiently.

The current resource managers in WMSs, such as the Factory from the *glideinWMS*, can handle the current situation of about 170 WLCG sites. However, the *glideinWMS* factory is not designed to manage opportunistic resources.

Opportunistic resources can provide a big part of the computing power needed for the HL-LHC. Therefore, some collaborations integrate opportunistic resources from different providers, as e.g., the Swiss Centre for Scientific Computing (CSCS) in Switzerland or the HPC cluster ForHLR II at KIT. These resources are either integrated as a dedicated WLCG site such as the CSCS [81] or as extensions of existing sites, as is the case for ForHLR II and GridKa, which will be discussed in chapter 6.2. If the number of sites increases, also the complexity of the resource management increases. Additionally, opportunistic resources differ in software and hardware from those provided by WLCG sites. Resource managers currently used to manage resources in the WLCG are not designed to handle a heterogeneous resource pool. Therefore, additional software is needed for making opportunistic resources available for the HEP community and ensure an efficient usage.

The resource manager *Cloudscheduler V2* is developed to manage opportunistic resources to extend the existing computing resource pool of a single Grid site or user group. *Cloudscheduler V2* requests additional virtual machines at cloud providers based on attributes of waiting jobs. Jobs are categorized by their requirements, such as requested memory or CPU cores. *Cloudscheduler V2* has a list of provider and virtual machine flavor pairs for each of these job categories. This list contains only one pair per provider and is ordered by priority. Periodically, *Cloudscheduler V2* checks the availability of virtual machines at each provider and the jobs in queue. For each job category, *Cloudscheduler V2* requests virtual machines that are available. If some virtual machines of one flavor are not fully utilized, *Cloudscheduler V2* stops requesting further virtual machines of that flavor. One limitation of this approach is that the batch system can schedule jobs differently than *Cloudscheduler V2* expects: If a job category is scheduled differently than expected by the batch system, the actually used virtual machines are overused while the expected virtual machines are underused. This results in inefficient resource usage.

A similar approach is used by the resource manager *ROCED*, which was developed and used at KIT. *ROCED* queries the OBS job queue for jobs that are able to run on a given virtual machine type. Based on this, *ROCED* calculates and requests as many `drone`s as needed to satisfy the required amount of CPU cores. *ROCED* does this for each type of virtual machine independent of the others. However, some jobs can run on different types. As a result, *ROCED* count these jobs several times, which leads to an overestimation of demand.

Each of these discussed resource managers predicts how many resources are needed at each provider. As long as the resources and `drone`s are homogeneous, the prediction is straightforward and accurate. In a heterogeneous system with different kinds of `drone`s, complex batch system policies or significant latency between resource request and availability, the matching and scheduling of jobs to resources gets more complex which results in less accurate predictions. In turn, inaccurate predictions cause that too many, too few or the wrong resources are requested, leading to inefficient use of resources.

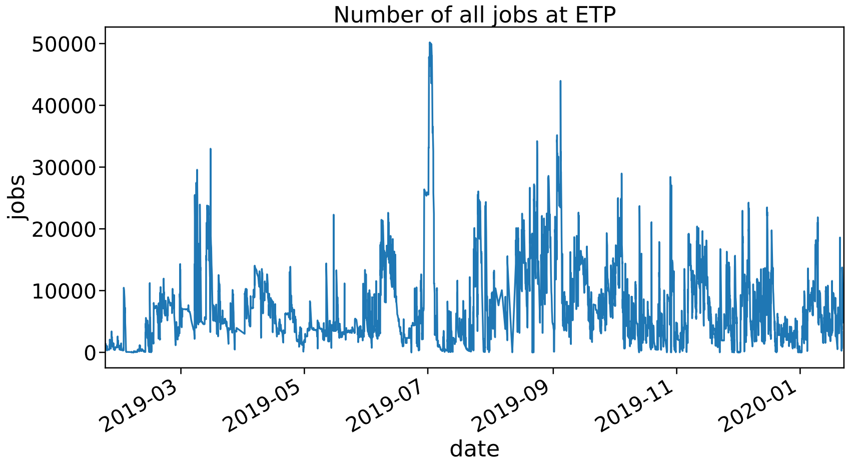


Figure 5.4: Number of jobs (running and idle) in the ETP batch system over a year.

To avoid requesting inefficient resources, it is necessary to know how the batch system will fill up available resources. However, the job scheduler makes decisions based on the current situation. For an exact prediction of a job scheduling decision, it is necessary to know the future situation at the time of the decision. This includes information about the batch system job queue, as well as the availability of resources. The predictions of jobs and resource availability have to be as accurate as possible for the period between a resource being requested and usable. Otherwise, too many or too few resources could be requested, resulting in unused resources or a longer job waiting time. However, not even the prediction of the number of jobs in the job queue is trivial. The number of batch jobs varies widely in an OBS that is mainly used by end-users, such as at the OBS at ETP. Figure 5.4 shows the number of jobs in the ETP batch system over a time period of one year.

Different approaches to predicting the number of jobs and the precise requirements in CPU cores, disk, and memory, based on historical data, have been studied. However, these predictions were found to be only valid for a time range of a few minutes. [82] For opportunistic resources, this delay between a resource is requested and is available depends mostly on the operating model of the provider. Commercial cloud providers, which usually have free resources, can provide resources within minutes. HPC clusters typically have a higher delay in providing resources due to their high utilization, which results in additional time passing while the resource manager waits for a free resource. Therefore, predictions are difficult for all resource providers, since the delay between requesting a resource and its availability is for most resource providers longer than the valid time range of the prediction.

Planned based resource manager also needs a precise prediction of the availability

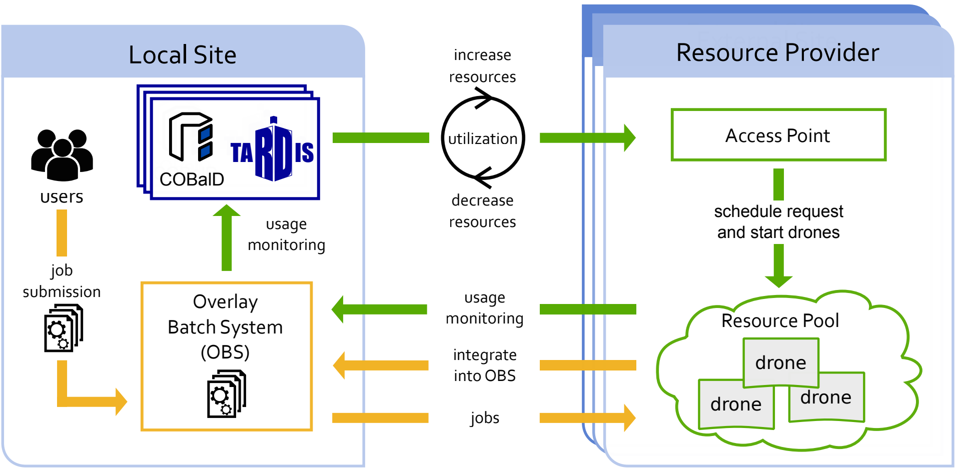


Figure 5.5: With `COBalD` and `TARDIS` it is possible to transparently integrate resources from multiple resource providers. Furthermore, it is possible to run one `COBalD` / `TARDIS` instance for each resource provider. `COBalD` and `TARDIS` monitor resource usage via the OBS. Based on the usage of resources, the number of resources will be increased or decreased via the access point of the provider. After requesting new resources, the resource provider schedules the resource request and starts the drones. The `drone` itself integrates the resources into the OBS. The OBS schedules jobs to the drones and provides information about the resource usage for `COBalD` and `TARDIS`.

of the resources. The availability of resources depends, among other things, on the usage of other users. The demand for resources from other users at cloud providers and HPC clusters are also challenging to predict, which increases the complexity to predict the availability of further resources.

A new approach was developed to provide a resource management that works transparently for users and does not rely on resource and job predictions. The basic idea of the resource management approach is based on a feedback loop. The batch system resource usage is fed back to the resource manager that decides to increase or decrease the number of resources based on their usage. Therefore, the resource manager requests more resources that are well-used and releases resources that are poorly used. This approach is implemented by the resource management software `COBalD - the opportunistic Balancing Deamon` (`COBalD`) and **Transparent Adaptive Resource Dynamic Integration System** (`TARDIS`) and shown in Figure 5.5.

This feedback loop approach has some advantages over methods used in the resource managers described before. First, it is more straightforward to react to the current state of the OBS instead of predicting the OBS job scheduler. Second, the resource manager and the job scheduler operate as two separate systems. The job scheduler makes decisions based on the jobs and available resources for them. Thereby, the resource manager makes decisions based only on the utilization of the resources available to the batch system. This also enables the resource manager to react automatically to changes of the job scheduler. Third, it is possible to split the entirety of resources into smaller parts, where each part has one resource manager. Since the resource manager only needs the utilization of the resources managed by itself for the decision, it is possible to split the resource pool into subsets. This enables to run multiple instances independent of each other, which results in high scalability. Furthermore, it is possible to configure each resource manager instance according to the general conditions at resource providers, such as limited time range for usage or network limitations.

`COBalD` [83] is the decision component of the resource management. It handles resources on an abstract level, allowing for various kinds of resources, e.g., the number of `drone`s for an OBS. `COBalD` manages resources in the form of pools. Each pool can contain resources or other pools to provide a hierarchical structure. A pool has the attributes `demand`, and `supply` of resources as well as occupancy, and suitability describing the usage of the resources. The decision to increase or decrease the `demand` is taken by a decision instance, referred to as controller. Several metrics can be used to define the usage, e.g. the fraction of used CPU cores, percentage of memory usage, allocated disk space, and used network bandwidth. `COBalD` reduces the complexity by combining the metrics into two metrics. occupancy describes how much of the provided resource is used and whether it can fulfill further demand, e.g. run further batch jobs, whereas the suitability describes how well the resource matches the current demand. The occupancy is per definition bigger or equal to the suitability. If the occupancy is higher than a configured limit, the controller will increase the `demand`. On the other hand, if the suitability is below a configured limit, the controller will decrease the `demand`. Otherwise, the `demand` remains unchanged.

However, `COBalD` only interacts with pools, which are an abstraction of resources. To manage, request, and release resources via `drones`s for an OBS, resource life cycle management is needed. For that, `TARDIS` is developed. `TARDIS` determines the metrics `supply`, occupancy, and suitability. Occupancy and suitability are based on the fraction of used to available resources, such as memory or CPU cores. One example of that fractions is the CPU-efficiency of a process, where the used CPU time is divided by the run time of the process. However, the scheduling decision of an OBS is based on the resources allocated to batch jobs and not by their actual usage. Therefore, OBS counts a resource as used when it is assigned to a batch job. For these resources, the fraction of allocated resources to available resources is used. As mentioned before, occupancy and suitability are combinations of several metrics. Figure 5.6 illustrates the definition of occupancy and suitability.

The occupancy is defined as the maximum of the ratios for each of the available

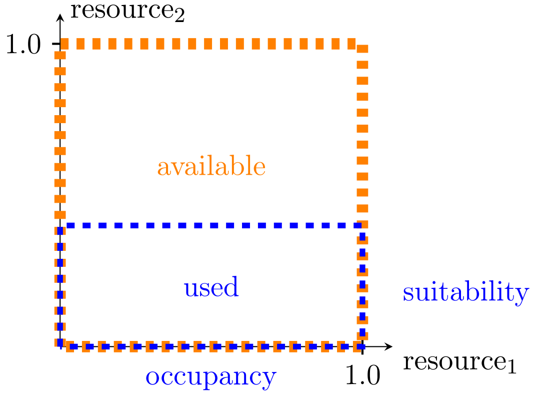


Figure 5.6: The orange box illustrates a drone with two resources. The axes represent the resources. The blue box illustrates how much of each resource is used. Since $resource_2$ is the least utilized resource, it determines the suitability, while $resource_1$, being the most used resource corresponds to the occupancy.

resources.

$$\text{occupancy} = max\left(\frac{\text{resource}_1^{\text{used}}}{\text{resource}_1^{\text{available}}}, \frac{\text{resource}_2^{\text{used}}}{\text{resource}_2^{\text{available}}}, \cdots \frac{\text{resource}_n^{used}}{\text{resource}_n^{\text{available}}}\right) \tag{5.1}$$

The suitability is described by the minimum of the ratios of used to available resource.

$$\text{suitability} = min\left(\frac{\text{resource}_1^{\text{used}}}{\text{resource}_1^{\text{available}}}, \frac{\text{resource}_2^{\text{used}}}{\text{resource}_2^{\text{available}}}, \cdots \frac{\text{resource}_n^{used}}{\text{resource}_n^{\text{available}}}\right) \tag{5.2}$$

Based on the occupancy and suitability, the controller of a pool adjusts the `demand`. `TARDIS` requests as many `drones` as needed to fulfill the `demand`. The `supply` is also defined by `TARDIS` and refers to how many CPU cores are available in the OBS. Both, the `demand`, and the `supply`, are defined in a unit of CPU cores.

Chapter 6

Infrastructures with Dynamic and Heterogeneous Computing Resources

The increasing data rate of current HEP experiments makes it necessary to use additional computing resources. These usually have different hardware configurations or environments which result in a heterogeneous infrastructure. The integration, management and usage of such resources are tested at the ETP and GridKa computing infrastructure.

6.1 ETP Computing Infrastructure

At the ETP, a computing infrastructure is operated to provide computing power for local research groups. About 40 users from different collaborations (CMS, Belle II, and Alpha-Magnet-Spectrometer (AMS)) use this computing infrastructure for their research. Compared to thousands of members in a HEP collaboration this is a relatively small user community. This enables direct communication with end-users from various collaborations and makes the ETP an ideal test environment for developments in computing for HEP. Software and concepts developed and tested at the ETP can later be used by other data centers, such as the WLCG Tier1 center GridKa which collaborates closely with the ETP.

6.1.1 Computing Resources at ETP

Several computing resources from various providers are available to the members of the ETP. Desktop-PCs enable the members to do their office work, write software, and connect to development machines. The development machines are designed for developing and testing analysis software. In addition, they serve as a place to run short and compute inexpensive interactive programs. Computing tasks exceeding the scope of the development machines are run via a batch system on worker nodes.

At the ETP, an `HTCondor` [84] batch system ensures a high utilization and efficient usage of a huge amount of computing resources. Additional to the worker nodes at the ETP, the HTCondor instance also uses resources dynamically provided via COBalD and TARDIS, since the ETP HTCondor instance acts as an OBS. The resources within this OBS are divided into several classes. The resources within a resource class are similar in terms of hardware, environment, and resource handling.

Some resource classes include machines that are located at ETP and within the ETP network. Therefore, machines of these resource classes can access additional services at ETP, such as ETP file servers. These resource classes are:

- *blade*: Older dedicated worker nodes that provide hardware similar to Grid job requirements.
- *schnepf*: Dedicated worker nodes with recent hardware, designed for end-user analysis. They have a higher memory per core ratio than the machines of the *blade* resource class.
- *supermachines*: Dedicated worker nodes for high throughput workflows equipped with Solid State Disks (SSDs) and additional Hard Drive Disks (HDDs) for caching.
- *desktop*: Desktop-PCs are also included in the ETP OBS. These desktop-PCs provide a sufficient amount of resources for the common desktop-PC usage and additional batch system jobs. Since desktop-PCs' main purpose is on the direct user interaction, batch system jobs are suspended, if user interaction takes place. As the network bandwidth for the desktop-PCs is limited and needs to be sufficient for user interaction, it is desired to run only jobs requiring a low data throughput on the desktop-PCs.

Most of the resource classes in the ETP OBS are not physically located at the ETP, these are:

- GridKa School resources (*GKS*): Virtual machines provided by GridKa and managed via OpenStack. These resources are used for training courses at the yearly GridKa school. When the resources are not needed for GridKa school, the ETP is able to use these resources.
- OpenTelekomCloud (*OTC*): In the scope of the Helix Nebula Science Cloud project [85], GridKa, as well as ETP, were allowed to use resources from the Open Telekom Cloud (OTC). Also, OTC uses OpenStack to manage their virtual machines. OTC provides different kinds of virtual machines, which enable to use of various hardware setups.
- *Exoscale*: Also in the scope of the Helix Nebula Science Cloud project the commercial provider Exoscale [86] was used. They provided virtual machines via CloudStack [87].

Resource Class	CPU cores	Resource Class	CPU cores
desktop	300	ForHLR2	up to 800
blade	288	OTC	up to 500
schnepf	192	Exoscale	up to 400
supermachine	96	BWFORCLUSTER	up to 6000

Table 6.1: Number of CPU cores per resource class. On the left side are the resource classes located at ETP. On the right side are the external resource classes.

- *ForHLR2*: The ForHLR II [88] is an HPC cluster at KIT, part of the bwHPC-C5 project [89]. This cluster is designed for HPC jobs that require several nodes and uses the batch system SLURM [90]. Due to the nature of scheduling multi-node jobs, frequently unclaimed resources are available. These unclaimed resources can be used by HEP HTC jobs, since these can be typically executed on single nodes.
- *BWFORCLUSTER*: The high-performance compute cluster `bwForCluster NEMO` [91] is also part of the bwHPC-C5 project and uses the batch system MOAB [92]. While the ForHLR II is a general-purpose HPC cluster, the `bwForCluster NEMO` was specifically designed for the neuroscience, elementary particle physics, and microsystems engineering communities in Baden-Württemberg. As the ETP is part of the elementary particle physics community in Baden-Württemberg, a dedicated share of the provided computing resources has been granted.

Table 6.1 shows the number of CPU cores for each resource class.

External resources are challenging, due to different operation policies, operating systems, and deployed software stacks. To cope with this diversity of software environments, the `drone` concept (see 5.3) is developed to enable transparent usage for the users. The `drone`s used for the different resource classes are not only different in how they provide the required software environment, they also differ in hardware configurations (e.g. memory, CPU cores, and disk space) provided in the OBS. An overview of the resources and how the software will be provided at the different resource classes is shown in Table 6.2.

6.1.2 Resource management at ETP

External resources used by ETP are managed using the software `COBalD` and `TARDIS`. In combination with the `drone` concept, this enables using resources from multiple providers in a dynamic and transparent way. Figure 6.1 shows the number of used CPU cores per resource class at the ETP. At the peak, a maximum number of 6402 CPU cores were used by the ETP OBS. This corresponds to the magnitude of the share of the CMS collaboration at the Tier1 center GridKa.

Resource class	Resource Provisioning	Environment Provisioning	RAM(GB) per Core
GKS	VM	docker	2
OTC	VM	docker	2-4
Exoscale	VM	docker	2-4
ForHLR II	batch job	singularity	3.2
BWFORCLUSTER	VM	docker	5

Table 6.2: Table of resource classes and deployed `drones` (resource and software environment provisioning) for integration into the ETP OBS

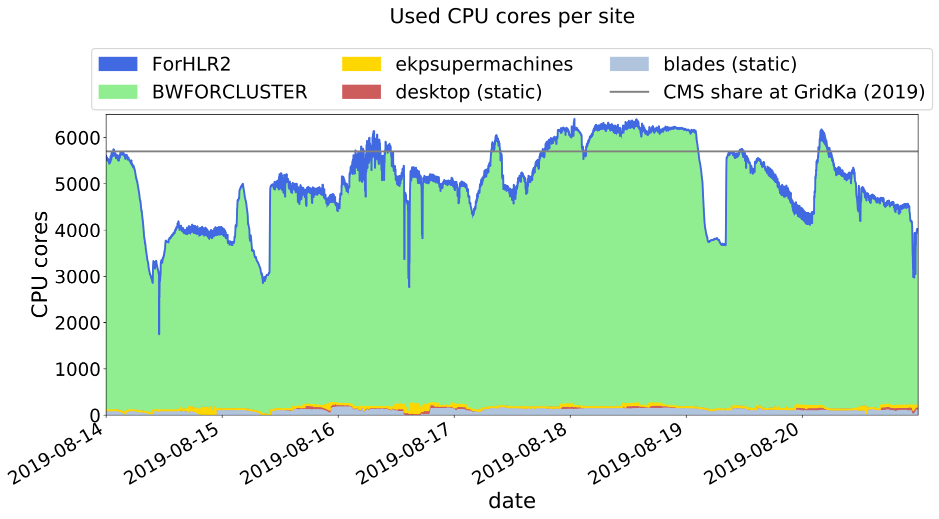


Figure 6.1: Used CPU cores per cloud site by ETP.

The resource management software `COBalD` needs some metrics to decide how many `drone`s have to be provided. For OBS usage, these metrics usually are derived from ratios of used divided by available CPU cores, memory, and disk space in the OBS per `drone`. As described in chapter 5.4, these ratios are combined to two metrics: suitability and occupancy.

First, the occupancy for the ETP resources is defined as:

$$\text{occupancy}_{\text{ETP}} = max\left(\frac{\text{CPUcores}^{\text{used}}}{\text{CPUcores}^{\text{avail.}}}, \frac{\text{memory}^{\text{used}}}{\text{memory}^{\text{avail.}}}, \frac{\text{disk space}^{\text{used}}}{\text{disk space}^{\text{avail.}}}\right) \tag{6.1}$$

If enough jobs are in the OBS to fill all available `drone`s, the occupancy for all `drone`s is one. In case of low job pressure, the occupancy drops to lower values.

To describe how well the resources meet the resource demand of jobs, the suitability is used. For the ETP the suitability is defined as:

$$\text{suitability}_{\text{ETP}} = min\left(\frac{\text{CPUcores}^{used}}{\text{CPUcores}^{avail.}}, \frac{\text{memory}^{used}}{\text{memory}^{avail.}}, \frac{\text{disk space}^{used}}{\text{disk space}^{avail.}}\right) \tag{6.2}$$

The values of the $\text{suitability}_{\text{ETP}}$ are, per definition, lower than or equal to the $\text{occupancy}_{\text{ETP}}$.

The resource management operation with these metrics used by `COBalD` and `TARDIS` can be shown by a single resource provider. Figure 6.2 shows the number of used CPU cores at the *BWFORCLUSTER* integrated via `drone`s. These `drone`s are managed by one instance of `COBalD` and `TARDIS` dedicated to the *BWFORCLUSTER*.

`COBalD` and `TARDIS` enable managing several resources from different providers in a transparent and easy way. Figure 6.3 shows the $\text{occupancy}_{\text{ETP}}$ for the external classes of resources *ForHLR2* and *BWFORCLUSTER*, as well as the internal resource class *schnepf*. The resource classes *BWFORCLUSTER* and *ForHLR2* provide resources at HPC clusters. Each of both resource classes is managed by one dedicated `COBalD` and `TARDIS` instance. According to the limited lifetime of the `drone`s of the resource classes *BWFORCLUSTER* and *ForHLR2*, the worker node instance inside each `drone` checks if the requested run time of a batch job is lower than the remaining lifetime of the `drone` before accepting new jobs. The rejection of batch jobs that request a longer run time than the remaining lifetime of `drone`s results in unused resources. The amount of unused resources is reduced by automatically stopping the `drone`s after ten minutes without running a batch job. Therefore, the occupancy of the resource classes *BWFORCLUSTER* and *ForHLR2* is usually close to 1.0 and demands for more `drone`s. Sometimes the occupancy for these two resource classes is close to 0.0 with no demand. In this case, `COBalD` and `TARDIS` provide one `drone` to check whether there is demand for further `drone`s of that resource class. If the occupancy of a resource class is lower than its threshold `COBalD` and `TARDIS` do not increase the number of `drone`s.

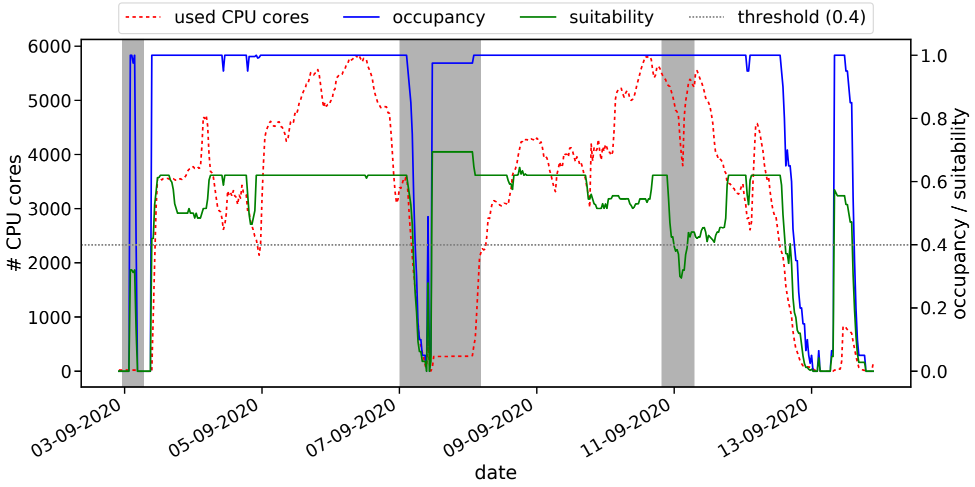


Figure 6.2: `Drones` managed via `COBalD` and `TARDIS` at the *BWFORCLUSTER*. Shown is the number of used CPU cores, the average occupancy and the average suitability of `drones` over a few days. In the left gray area, a negligible number of `drones` starts. The occupancy is close to 1.0 which means that there is demand for more `drones` and no further jobs can be scheduled to these `drones`. However, the suitability is below the threshold of 0.4. This means that the `drones` are not well used and that `TARDIS` does not request more of them and starts draining. After the jobs inside a `drone` are finished the `drones` shuts itself down. Between the gray areas, the values of occupancy and suitability change from time to time. This is caused by starting other kinds of jobs on these `drones`. The occupancy is close to 1.0 and the suitability is above the threshold. Therefore, more `drones` are requested. According to the limited lifetime of `drones` and the availability of resources at the `bwForCluster NEMO` the number of used CPU cores decreases sometimes. In the second gray area from left, there is first no further demand for resources which can be seen in the decrease of used CPU cores, suitability, and occupancy. After a short amount of time no further `drones` are needed which is shown by the small peak with low suitability. Later, the demand increases. This is visible in the number of used CPU cores, occupancy, and suitability. In the third gray area from the left, the provided resources do not match well the current mixture of jobs. That is shown in the reduced suitability and number of used CPU cores and results in a reduced number of `drones`. After the mixture of jobs changed the suitability as well as the number of `drones` and used CPU cores increase.

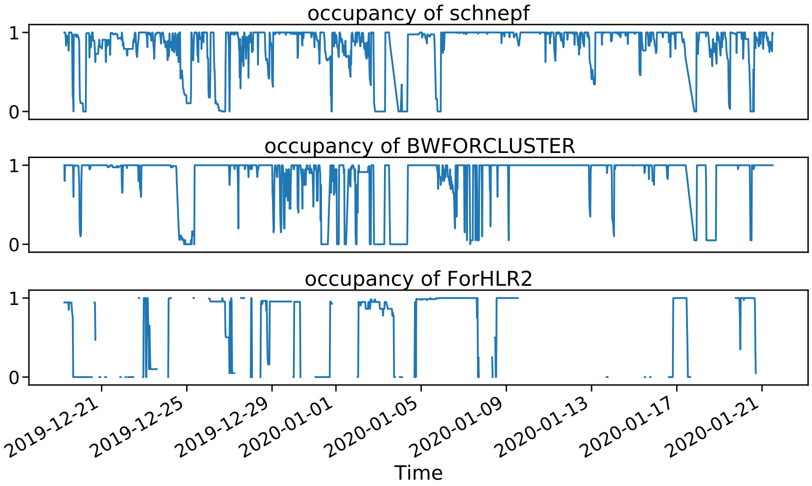


Figure 6.3: Average occupancy of `drones` over time for the three resource classes *schnepf*, *BWFORCLUSTER*, and *ForHLR2* in production. During periods without data points, no `drones` of the corresponding resource class were running.

The decision to decrease the number of `drones` is based on the suitability. Figure 6.4 shows the suitability over time for different classes of resources.

The resource classes *BWFORCLUSTER* and *ForHLR2* show higher suitability than the *schnepf* resource class. The resources of the *schnepf* class are designed to run almost any job that is submitted by the end-users to the OBS of the ETP. Furthermore, these machines are also designed to be suitable for workloads with higher memory requirements. Consequently, they provide more memory and disk space per job than currently needed. Therefore, not all resources (CPU cores, memory, disk space) are fully occupied the whole time. Furthermore, the *schnepf* resource class provides only a small fraction of the computing power available for the user at ETP (see Figure 6.1). Therefore, the users optimize their jobs, and their requirements, to the class that provides the majority of resources. This is *BWFORCLUSTER*, with up to 6000 CPU cores between the years 2018 and 2020.

If the suitability of a resource class is too bad, `TARDIS` reduces the number of `drones` for the corresponding resource class. The definition of a badly-used resource is different for users or funding agencies. A funding agency would set a high eligibility threshold to ensure that a low percentage of resources are unused. A high threshold would result in fewer but well-used resources. On the other hand, users would set a low eligibility threshold, resulting in more available resources. For *BWFORCLUSTER* and *ForHLR2* this threshold is set to 0.5. Both resource classes provide resources from HPC clusters. The usage of these resources does not cause

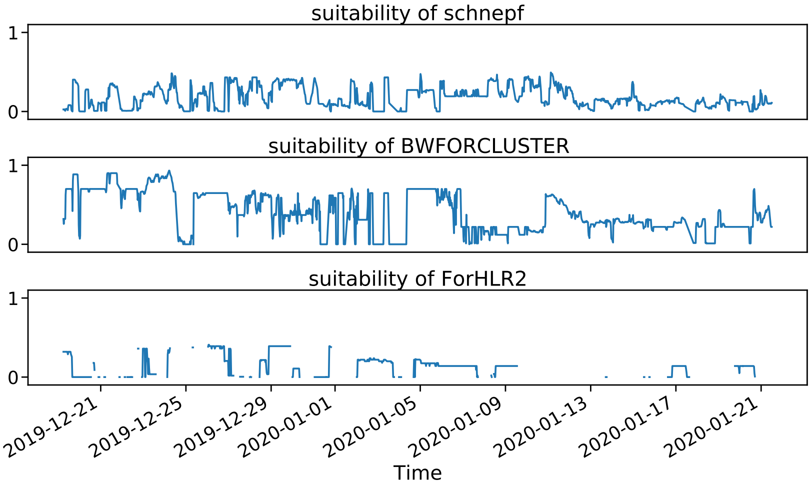


Figure 6.4: Average suitability of `drones` over time for the three resource classes *schnepf*, *BWFORCLUSTER*, and *ForHLR2* in production. During periods without data points, no `drones` of the corresponding resource class were running.

additional costs. For these resources, the suitability threshold is set to a value that enables to use of the resources also in a less optimal way. However, it is possible to consider costs for procuring or using the resource into the occupancy and suitability. Commercial cloud providers calculate the costs based on the usage. Therefore, the suitability threshold for such a provider is set to a higher value, such as 0.8 for *OTC*.

The OBS at ETP transparently provides computing resources to users. However, the used concepts and software can easily be adapted to other communities and institutes. Furthermore, the concept is scalable and allows including more sites where each site has one `COBalD TARDIS` instance and can be configured at wished granularity by splitting different resource classes in multiple hierarchically structured pools.

6.2 Opportunistic Computing Resources in the Grid

After a successful testing period at the ETP `COBalD` and `TARDIS` are also used at GridKa. However, the situation at a WLCG site is different than at an institute. Users do not submit directly to the WLCG site; they submit their jobs to the collaborations' OBS. The OBS schedules the jobs to pilots running at WLCG sites. WLCG sites see only pilots, not the user jobs, in their batch system. Furthermore, instead of fluctuating demand for resources, collaborations usually have idle jobs in their OBS queue. Therefore, collaborations are pleased to use additional resources.

However, opportunistic resources have some drawbacks. As described before, opportunistic resources are not dedicated to HEP; it is challenging to predict when and how long these resources are available. Furthermore, it can happen that the resources must be released immediately (so-called pre-emption), which results in killing of `drone`s including the jobs running inside. This usually leads to a complete loss of the results computed so far.

Some collaborations use dedicated services, such as the ATLAS Event Service [93] for the production of Monte Carlo simulated events that save each event after production on remote storage. However, these solutions also need additional infrastructure and management. For this reason, these solutions are only used by a few collaborations. Jobs failing due to, for example, releasing of opportunistic resources are usually handled by the OBS and re-scheduled automatically for execution.

Furthermore, the provisioning and managing of opportunistic resources is more complex than usual WLCG site resources; some sites have dedicated entry points for opportunistic resources. Therefore, the same software is used as for the dedicated HEP resources. These entry points, so-called *Computing Element*s, accept jobs from outside a WLCG site and submit these jobs to the batch system of the WLCG site. Thereby, the *Computing Element* software, such as, `HTCondor-CE` [94] and `NorduGrid ARC-CE` [95] take care of authentication, authorization, and site specific adjustment of metadata of a batch job.

6.2.1 Additional Resources accessible via GridKa

The usage of a separate *Computing Element* enables the collaboration to control which jobs run on opportunistic resources. Furthermore, the collaborations can send jobs to these resources that can run efficiently on them. Therefore, GridKa provides an extra OBS for opportunistic resources accessible via a *Computing Element*.

This also enables further providers to contribute to resources to the Grid. Instead of running complex Grid services such as *Computing Element*s, they can dynamically integrate resources into an OBS via the lightweight resource manager `COBalD` and `TARDIS`. Besides, they still have full control over their resources by running and managing their `COBalD` and `TARDIS` instance.

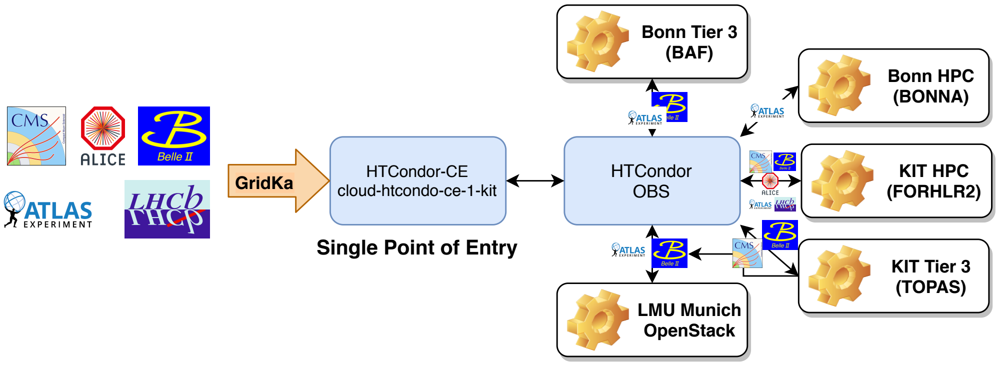


Figure 6.5: The collaborations send, as usual, pilots to a specific *Computing Element* at GridKa. The *Computing Element* instance cloud-htcondor-ce-1-kit sends the request for a `drone` as a new job to an OBS HTCondor instance dedicated to opportunistic resources. Inside this OBS are resources integrated via `drones` from Bonn, KIT, and Munich. Depending on the policies of the resource provider, `drones` accept pilots from all or predefined collaborations. As Munich, for example, has only a Belle II and an ATLAS group, their `drones` accept only Belle II and ATLAS `drones`. [97]

The University of Bonn is the first partner that runs a `COBalD` and `TARDIS` instance to provide free resources to the Belle II and ATLAS collaborations outside of KIT. [96] If they need their resources for local users, the `drones` integrating the resources to the GridKa OBS get drained.

Additional to the resources provided by the University of Bonn GridKa operates an entry point for other opportunistic resources provided by the KIT Tier3, the ForHLR II, and the LMU Munich. Figure 6.5 shows the setup to provide access to opportunistic resources for the collaborations.

This setup is mainly used by the ATLAS, Belle II, and CMS collaborations and is shown in Figure 6.6. Thereby, up to about 4500 CPU cores from opportunistic resources are used, which correspond to about 9% of the GridKa computing resources in 2020.

However, most opportunistic resources do not provide a high bandwidth network connection to Grid storage. The available bandwidth could be insufficient for some jobs. Therefore, it is necessary to consider the available bandwidth when managing opportunistic resources.

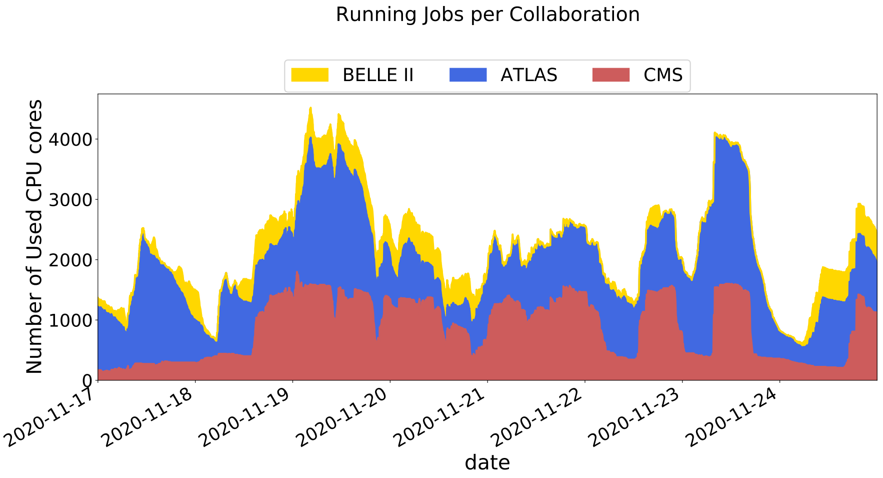


Figure 6.6: Number of used CPU cores per collaboration (ATLAS, Belle II, CMS) at opportunistic resources accessible via GridKa.

Chapter 7

Network aware Resource Scheduling

The usage of opportunistic resources brings further challenges, such as providing a dedicated software environment and managing resources at a large number of providers. In addition to those challenges, the HEP community and other scientific communities have several workflows that process large amounts of data. WLCG sites are designed for a mix of CPU-intensive jobs such as Monte Carlo production, and I/O-intensive jobs such as end-user analysis, and event reconstruction. To cope with I/O-intensive jobs, Grid sites provide a high bandwidth connection between their storage system and worker nodes. The collaborations schedule jobs to Grid sites where the corresponding datasets are stored and can benefit from the high bandwidth connection.

Opportunistic computing resources have to read data from and write their results to a Grid site. When jobs on opportunistic resources read files, mostly streaming via `XRootD` [53] is used. The streaming runs in parallel to the data processing, which reduces the CPU idle time compared to copying the full files to the worker node first. Furthermore, streaming also reduces the worker node's required storage space because only the results are temporarily stored locally. Result files of jobs are usually stored locally on the worker node and at the end of the job copied to Grid storage. In time of high performance storage systems and SSDs the local storage is usually not the limiting factor. The processing, simulation, or analyzing of events is usually limited by the input throughput. The bandwidth between opportunistic computing resources and Grid sites is usually lower than inside a Grid site. Furthermore, worker nodes at one provider have to share the bandwidth with traffic of processes from other users at the same provider. This can result in insufficient network bandwidth for I/O-intensive jobs. Such an insufficient network bandwidth between a worker node and the used storage system reduces the CPU-efficiency because the needed data are not available in time.

7.1 Correlation between CPU-efficiency and Network Throughput

To prove this assumption, the correlation between incoming network bandwidth and average CPU-efficiency for I/O-intensive end-user jobs is studied. Therefore, we collected job attributes of about 1.7 million jobs between 21.4.2019 and 24.2.2020.

The CPU-efficiency is defined as the CPU-time divided by CPU-count and wall-clock time. We expect for I/O-intensive workflows, that jobs with the same executable but different input data have the same ratio between average incoming network throughput and CPU-efficiency. The job information about CPU-time is provided by the `HTCondor` batch system at the ETP and GridKa directly. `HTCondor` gets the CPU-time values via the Linux kernel feature *cgroups*. The amount of network traffic used by a job is also provided by `HTCondor` for jobs that run in a `docker` container. The information about CPU-time via cgroups and the network traffic provided by the `docker` daemon is exact. However, `HTCondor` only updates these values periodically every 10 minutes. This results in an uncertainty in CPU-time and network traffic. However, the size of the relative uncertainty decreases with the job runtime.

To get a reliable set of I/O-intensive workflows, only jobs that run at least 30 min are used. These jobs are clustered by the attributes: working directory on the submit machine, executable, and user. From that set of workflows, only workflows with a median network throughput of above $1\,\mathrm{MB\,s^{-1}}$ are analyzed. After filtering, this study comprises 134 workflows. Figure 7.1 shows three out of these I/O-intensive workflows.

For each of these workflows, the correlation between the average incoming network throughput and the CPU-efficiency is determined. The number of workflows per correlation bin is shown in Figure 7.2. More than 77% (103) of the workflows have a correlation coefficient above 0.5. Some of the workflows with a negative correlation between CPU-efficiency and average incoming network throughput, see Figure 7.3. These clusters are different workflows with the same job attributes which result in a misclustering. This happens when users change their code, mostly because of bug fixing, and resubmission of jobs.

These results show that insufficient network throughput results in reduced CPU-efficiency. In addition to insufficient network bandwidth, problems with and throughput limitation of the Grid storage can reduce the CPU-efficiency. Furthermore, the correlation between average incoming network throughput and CPU-efficiency also depends on the performance of the CPU. With the same average incoming network throughput, a more powerful CPU has a lower CPU-efficiency than a less powerful one, due to its higher processing speed. This will be further discussed at the end of this chapter.

One possibility of avoiding the network limitations at opportunistic resources is to

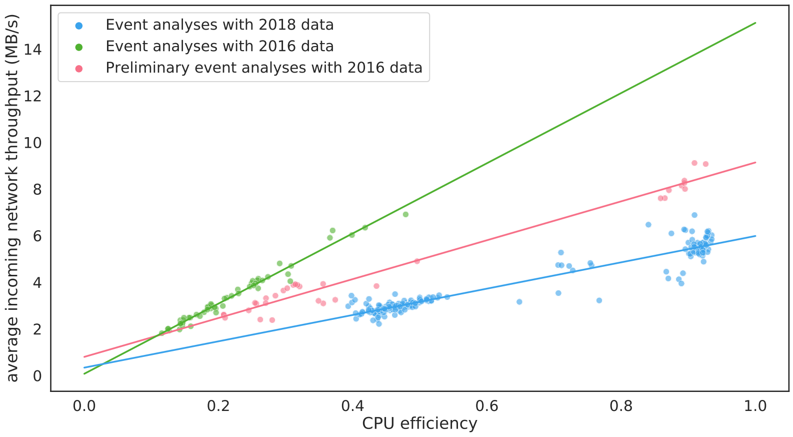


Figure 7.1: Three different workflows are shown and marked with differing colors. Each dot represents one job. The line represents a linear regression to the jobs of a workflow. This leads to a high correlation between average network throughput and CPU-efficiency. The workflows shown differ in the input data and in the analysis code which results in different slopes of the fitted function. Each of the workflows does event based analysis on events recorded by the CMS detector. The green and pink workflows analyze events recorded in 2016 with different event reconstruction and analysis software. The blue workflow analyses events recorded in 2018, also with another version of the analysis software compared to the 2016 analyses.

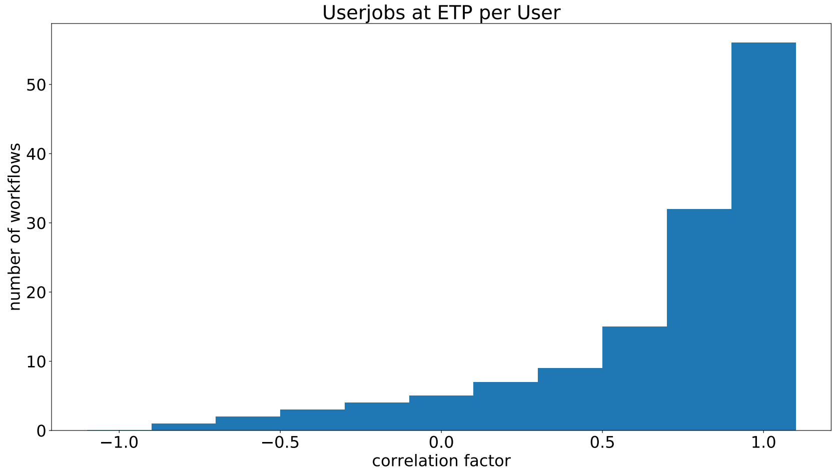


Figure 7.2: Histogram of the number of workflows per correlation factor between average incoming network throughput and CPU-efficiency.

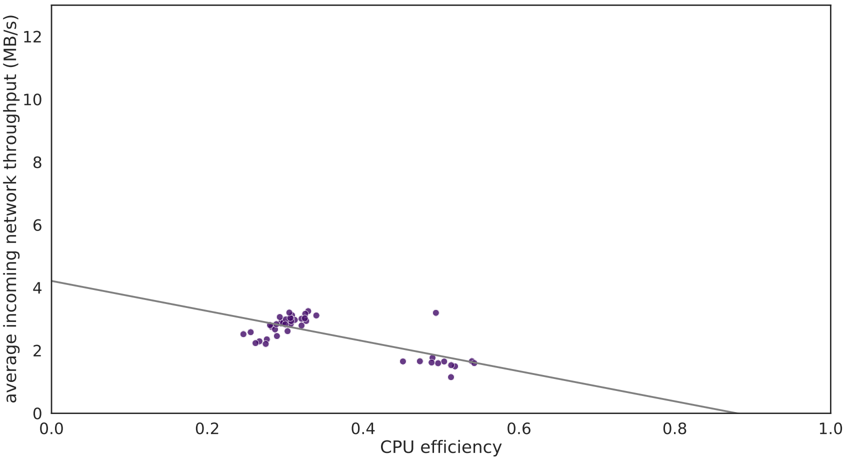


Figure 7.3: Average incoming network traffic over CPU-efficiency for different jobs of one clustered workflow. Two job clusters are visible: one with about 30% CPU-efficiency and one with about 50% efficiency. The gray line shows a linear function fitted to all job points. All jobs in that workflow run at the `bwForCluster NEMO` cluster. The negative slope of this function represents the unexpected behavior of the complete cluster.

schedule only CPU-intensive jobs with a low requirement of network bandwidth to these resources. However, this would result in a higher concentration of high I/O jobs at Grid sites. The Grid sites are designed for a mixture of I/O-intensive and CPU-intensive jobs. Without significant changes at Grid sites, a higher concentration of I/O-intensive jobs would result in CPU-inefficiencies and I/O-intensive jobs would then not profit from opportunistic resources.

Another possibility of avoiding network bandwidth limitations is to cache data close to opportunistic computing resources[98]. Due to improvements in analysis techniques and a better understanding of the detector during the course of an analysis, it is common for several jobs to run over the same files in an end-user analysis. Such analysis can profit from cached data. Some resource providers, such as HPC clusters, have a high performance storage system within their infrastructure, which can be used as cache storage. The available network bandwidth inside a cluster is usually higher than the outgoing network bandwidth. However, jobs only profit from caching if the required files are already in the cache. Furthermore, due to limited cache storage, not all files can be cached for a longer time. Therefore, some jobs have to read their files from remote storage.

To avoid reduced CPU-efficiencies caused by insufficient network bandwidth, job scheduling must take into account the available network bandwidth. One approach would be to include the network bandwidth as a resource of a worker node. This would enable a batch system such as `HTCondor` to schedule jobs based on the requested network bandwidth. This approach requires that users request a network bandwidth for their jobs. This is similar to the number of CPU cores, RAM, and disk space that users already request per job. However, for these values, users have more experience due to the development and testing phase of their code. This is often done on development machines where users can log in and monitor the resource usage of their programs such as RAM and CPU with standard system tools. For the network usage, this is more complex and requires further effort for the users. As a result, the value given by users for the required network bandwidth could often be inaccurate.

Furthermore, the bandwidth between a worker node and a storage system requires knowledge about the network topology. However, this is often unknown to the batch system and not static over a longer period of time. Additionally, the network may be used by other network traffic not related to job usage. This requires a continuous adjustment of available network bandwidth to take into account varying network usage. This in turn requires a continuous determination of the available bandwidth. Without access to up-to-date monitoring information, network benchmarks could be used. Network benchmark tools such as `iperf` [99] determine the network bandwidth by exhausting the network with data transfers and determine the available bandwidth by the measured throughput. This method would interfere with other benchmarks and jobs and cause artificial traffic in the network.

Our approach to considering the available network throughput in resource scheduling is to use an indirect measurement of the available network throughput. Figure 7.2 shows the correlation between the CPU-efficiency and average incoming network throughput for I/O-intensive jobs. An insufficient incoming network throughput results in a reduced CPU-efficiency. Therefore, it is possible to detect network bandwidth limitations via reduced CPU-efficiency of jobs. By taking into account the average CPU-efficiency in the occupancy and suitability of `drone`s `TARDIS` is able to react to network limitations.

This enables to schedule `drone`s on resources that where the network bandwidth is not saturated. Since jobs can only be scheduled to resources included in the batch system, the `drone` scheduling considering network throughput also affects the job scheduling. This can be used to adjust the mixture of I/O-intensive and CPU-intensive jobs, described in the following. Two kinds of `drone`s run on the same resource provider. One kind of `drone` accepts only CPU-intensive jobs, and the other accepts I/O- and CPU-intensive jobs. `TARDIS` requests both kinds of `drone`s, as long as all jobs at the resource provider are running efficiently. When the CPU-efficiency drops, due to insufficient network bandwidth of the resource provider, `TARDIS` automatically reduces the number of `drone`s that accept I/O-intensive jobs. This results in a lower number of running I/O-intensive jobs on that resource provider. The `drone`s which accept only CPU-intensive jobs are not affected and `TARDIS` requests further `drone`s of that kind. Thereby, the ratio of CPU-intensive and I/O-intensive jobs changes until all jobs run efficiently on that resource provider.

This resource scheduling enables an indirect job scheduling of I/O-intensive jobs. However, it is necessary that jobs get flagged as I/O-intensive jobs. Furthermore, the OBS has to consider this flag when scheduling jobs. Except for this flag, our approach provides a dynamic and simple way to efficiently schedule I/O-intensive jobs transparent for the user maintaining a high CPU-efficiency and avoiding network limitations.

To show that our approach works, it was benchmarked on a real system. An I/O-intensive workflow was submitted, which is scheduled to a static resource optimized for I/O-intensive jobs and dynamic resources optimal for CPU-intensive jobs. The resource for I/O-intensive jobs was a worker node of the TOpAS cluster with 42 CPU-cores. The ForHLR II was used and managed via `TARDIS` for CPU-intensive jobs. Both resources were exclusively used for that benchmark. The I/O-intensive workflow includes 280 single-core jobs.

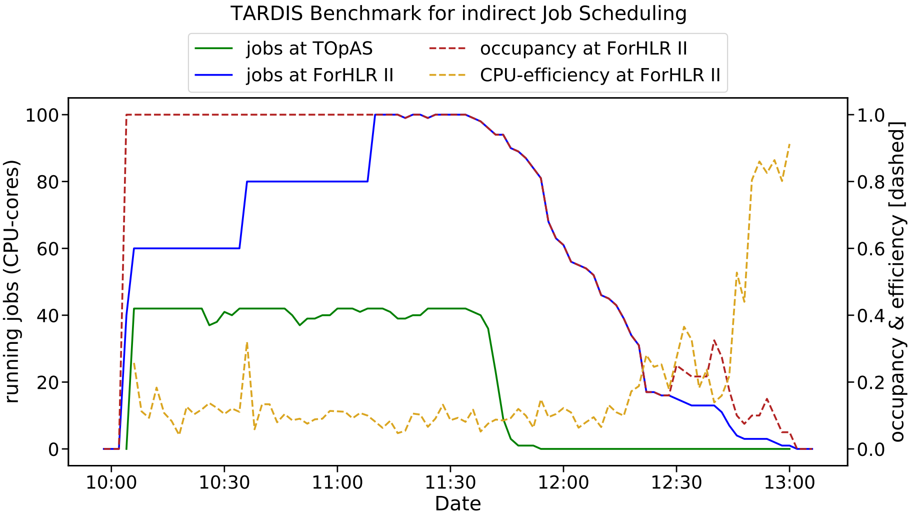


Figure 7.4: At the beginning of the benchmark jobs start in one ForHLR II `drone` (20 CPU cores per `drone`) and on one `TOpAS` worker node. `TARDIS` requests further `drones` according to the high occupancy. While `TOpAS` is almost finished with jobs at 11:45, the `drones` run progressively less jobs at ForHLR II. This results in a decreasing occupancy at ForHLR II until some `drones` shut down.

7.2 Benchmark: Network aware Resource Scheduling with Two Sites

To show the impact of our approach, the benchmark was initially performed without considering the network throughput. Figure 7.4 shows the number of running jobs at both resources, and CPU-efficiency of jobs as well as occupancy on ForHLR II for the case of `TARDIS` managing `drones` at ForHLR II independent of the CPU-efficiency.

For the next benchmark, the configuration of `HTCondor` and `TARDIS` are extended to provide and consider network throughput via CPU-efficiency. The configuration parts are available in appendix C. Figure 7.5 shows the number of running jobs at both resources, CPU-efficiency of jobs, and occupancy on ForHLR II.

In the first benchmark, without considering CPU-efficiency, 104 jobs (37%) run at ForHLR II, and 176 jobs (63%) run on the `TOpAS` node. In the second benchmark, considering CPU-efficiency more jobs run at `TOpAS`: 78 jobs (28%) on ForHLR II and 202 jobs (72%) on `TOpAS`. Both benchmarks run about 3 h, however, the benchmark considering CPU-efficiency used fewer resources, which would then be available to process other workflows.

This shows that `TARDIS` is able to schedule jobs indirectly via resource scheduling according to network bandwidth. It also results in less wasted resources due to

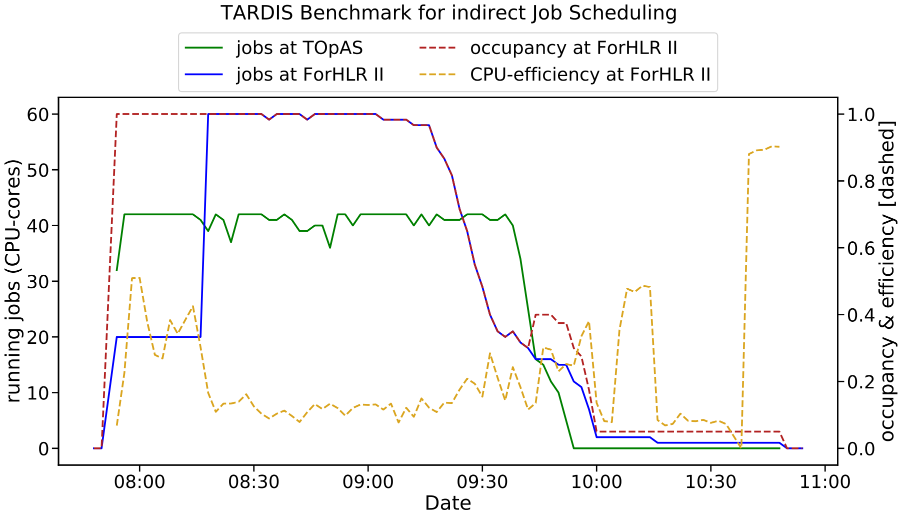


Figure 7.5: At the beginning of the benchmark jobs start in one ForHLR II `drone` (20 CPU cores) and on the `TOpAS` worker node. `TARDIS` has a response time of a few minutes to react on the low CPU-efficiency so two further `drone`s start. After that, `TARDIS` drains the `drone`s which results in a decreasing occupancy at 09:20. However, on `TOpAS` further jobs start. The increase in occupancy at 11:40 is a result of stopped `drone`s.

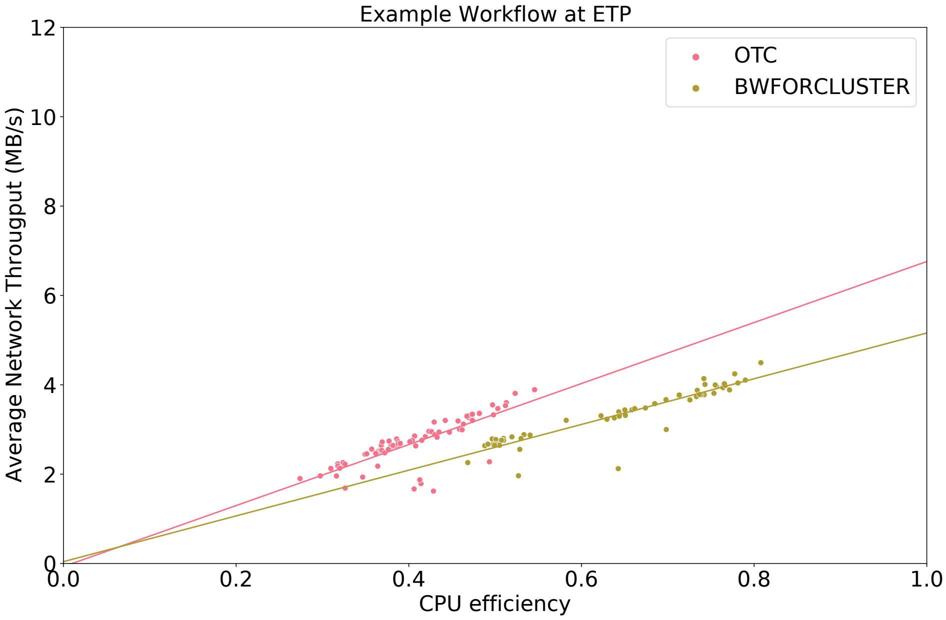


Figure 7.6: Average incoming throughput over CPU efficiency for one workflow at *BWFORCLUSTER* and OTC resources. The slope of the fitted linear function hints at the CPU performance of the used resources. In this case, resources at OTC have a higher CPU performance than resources at *BWFORCLUSTER*.

insufficient network throughput.

An additional improvement would be to schedule CPU-intensive jobs to resources with more powerful CPUs while I/O-intensive jobs could run on less powerful CPUs. As mentioned before, the correlation between average incoming network throughput and CPU-efficiency can also depend on the CPU performance.

Figure 7.6 shows one I/O-intensive end-user analysis workflow running on two types of resources. A linear function is fitted to the jobs per type of resource. The slope of function can be used as an indication of the performance of the used resources. The greater the slope the more powerful is the CPU, because the CPU can process more data per time.

The information about average network throughput over CPU-efficiency can be used to estimate the CPU performance of resources. Therefore, the estimation of CPU performance would be possible without reserving resources for dedicated CPU benchmarks. Furthermore, it would be possible to estimate available performance in real-time. This type of CPU performance estimation would be useful for shared or temporary available resources. However, this needs further investigation. Currently, CPU benchmarks are used to estimate the CPU performance. This is discussed in the next chapter.

Chapter 8

CPU Performance Benchmarks

Commercial cloud providers offer a massive amount of computing resources on-demand. They can contribute to the needed computing power for analysis, simulation, and reprocessing events for the HL-LHC era. However, it is necessary to know their performance and prices to decide whether it is more cost effective to procure additional resources for WLCG sites or use resources from commercial cloud providers.

There are different programs, such as the CPU benchmark suite from the Standard Performance Evaluation Corporation (SPEC)[100], which can be used determine computing performance. Until 2005, the computing performance was almost exclusively dependent on the CPU frequency. However, modern CPUs provide additional features that can improve the performance e.g., vectorization, which allows applying one operation on several data at the same time. Therefore, the run time of a program depends on the provided features of a CPU and the program's availability to use these. As a result, the SPEC benchmark suite provides multiple benchmarks to determine the performance of a CPU with respect to specific features and applications.

To measure the performance of a CPU for HEP applications, the HEPiX working group uses a specific set of benchmarks provided by the SPEC benchmark suite version 2006. The set of selected benchmarks as well as the metric to weight the different benchmark results are combined to the *HEP-SPEC06 benchmark*, which is used to determine the HEP-SPEC06 (HS06) score. The HEP-SPEC06 benchmark is the standard benchmark in HEP for CPU performance. Nowadays, one CPU usually includes several physical CPU cores. Measure the performance of complete CPUs; an HEP-SPEC06 benchmark runs several instances, usually the number of CPU cores visible to the operation system. However, due to hyper-threading the number of physical CPU cores can differ from the number of CPU cores visible to the operating system, also called *logical CPU cores*.

Almost every commercial cloud provider offers virtual machines. Some of these providers use one physical CPU core for several virtual machines. Thereby, multiple consumers share a physical system via multiple virtual machines. Due to the shar-

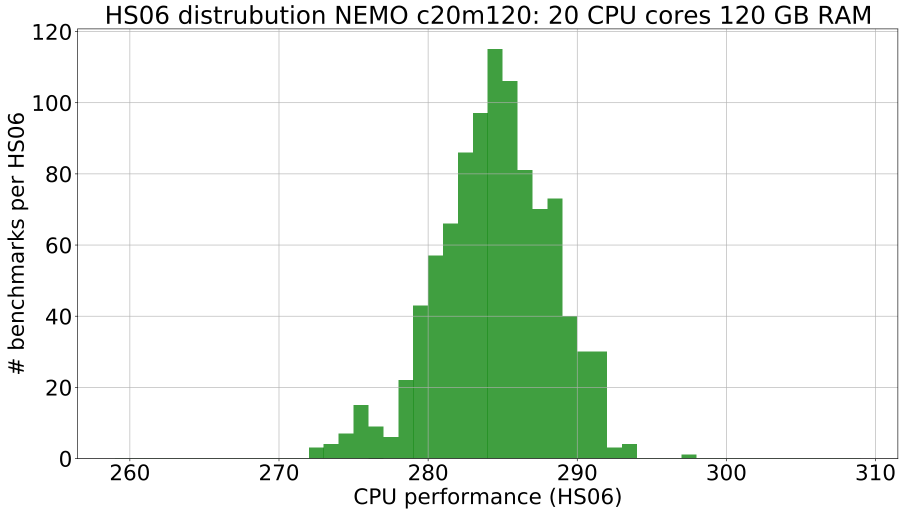


Figure 8.1: HS06 score distribution of benchmark jobs run in VMs at the `bwForCluster NEMO` cluster. The VMs have 20 CPU cores, and 120 GB memory. The mean of the distribution is 284 HS06 with a minimum of 272 HS06 and a maximum of 297 HS06.

ing and overbooking of physical CPU cores, it is necessary to run benchmarks to estimate the provided CPU performance. HS06 benchmarks were run to compare the performance of WLCG site resources with resources at cloud providers and their performance over time.

The main computing resource of the ETP between 2017 and 2020 is the `bwForCluster NEMO` HPC-cluster in Freiburg. Its scheduling policy enables the usage of different configurations of virtual machines on their physical nodes. The used nodes at `bwForCluster NEMO` are equipped with two E5-2630V4 processors. This results in 20 physical CPU cores per physical node.

The benchmarks were submitted as batch jobs to allow for a sufficiently large number of benchmark results and to study the performance over time. The distribution of the HS06 scores of benchmarks run at the `bwForCluster NEMO` cluster is shown in Figure 8.1.

The variance in the HS06 values is caused by additional programs running on the host and the availability and usage of the CPU's turbo boost technology which enables to increase the CPU frequency for some CPU cores while other CPU cores are less used. Due to the `bwForCluster NEMO` cluster scheduling policy, only jobs/virtual machines from one user run on a physical node. For commercial cloud providers, it is often the case that also virtual machines from other users are on the same physical

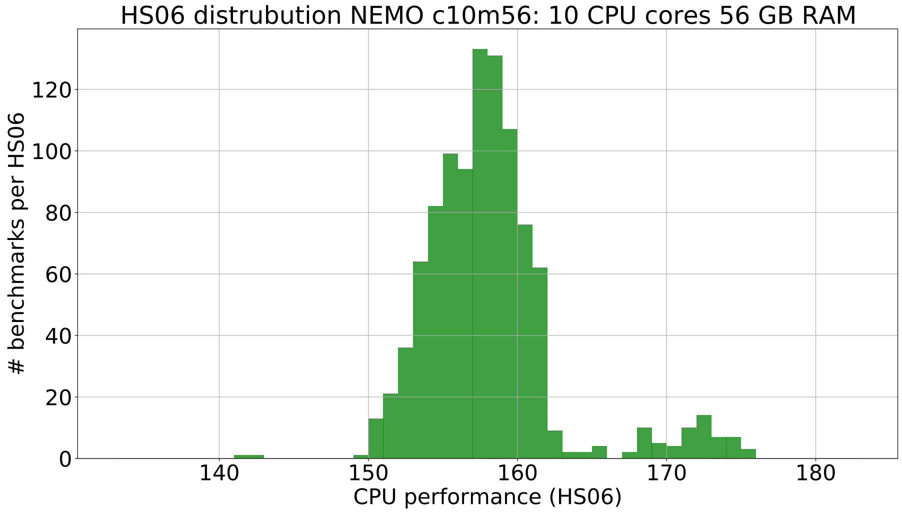


Figure 8.2: HS06 score distribution of benchmark jobs run in VMs at the NEMO cluster. The VMs have 10 CPU-cores, and 56 GB memory. The mean of the distribution is 158HS06, minimum of 141HS06, maximum of 176HS06, and a variation of 4.5 HS06.

host as ours. To study the influence of multiple virtual machines on the same physical machine, we also run benchmarks at the `bwForCluster NEMO` cluster with 10 CPU core virtual machines.

The usage of 10 CPU core virtual machines enables us to run two virtual machines on one physical host. Figure 8.2 shows the distribution of the benchmark results with 10 CPU core virtual machines. An HS06 score below half of the 20 CPU core distribution is expected due to the reduced number of CPU cores and the additional overhead of an extra operation system. Instead, the mean of the distribution is at 158HS06 which is higher than half the median for 20 CPU core virtual machines. The HS06 score of the 10 CPU core virtual machines also has a wider spread. The tails of the HS06 score distribution towards higher values are caused by the last benchmarks that run without another virtual machine on the physical host and profit from the turbo boost technology. Furthermore, a delay between the starts of benchmarks on the same host causes the increased HS06 scores.

These differences from expected values show that the CPU performance of a shared system is not trivially predictable. During the `Helix Nebula Science Cloud` project, HS06 benchmarks are run at OTC on two different kinds of virtual machines, so-called *flavors*. The specifications are shown in Table 8.1. The benchmarks are the same as performed at `bwForCluster NEMO`; HS06 runs as a job with as many

flavour	CPU cores	RAM (GB)	sharing	price (€/h)
s2.2xlarge.4	8	32	multiple VMs on one phys. machine overbooked CPUs	0.43
h1.2xlarge.4	8	32	multiple VMs on one phys. machine	0.66

Table 8.1: Overview of benchmarked virtual machine flavours at OTC. The number of CPU cores corresponds to the number of CPU cores inside the virtual machine. The main difference between the two flavours is, how they are shared with other virtual machines on a physical machine. The costs do not include storage of the virtual machine. Date: June 2018

flavour	mean	variance	minimum	maximum
s2.2xlarge.4	142.6	1.1	137.5	145.9
h1.2xlarge.4	148.7	11.9	123.9	167.5

Table 8.2: HS06 benchmark results of s2-flavour and h1-flavour virtual machines at Open-TelekomCloud

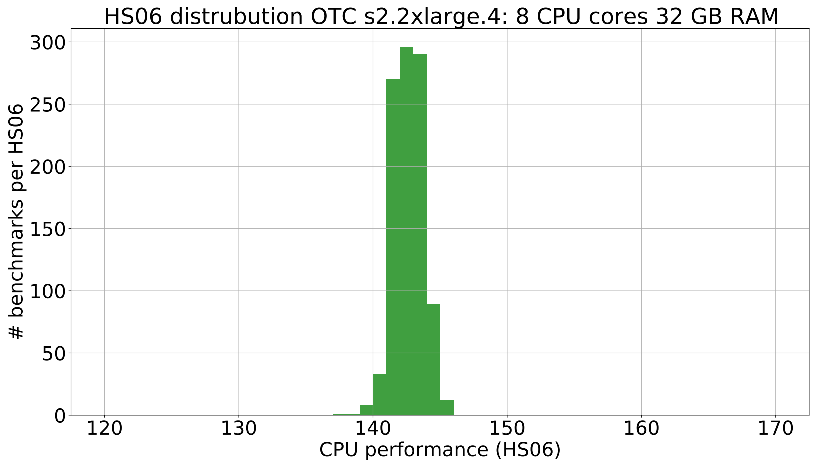
instances as CPU cores in the virtual machine.

The machines of the *s2.2xlarge.4* flavor, in the following s2-flavor, share CPU cores with other virtual machines on the same physical machine. Whereas for the machines of the *h1.2xlarge.4* flavor, in the following h1-flavor, the number of all CPU cores in the virtual machines correspond to the number of CPU cores of the physical machine. Therefore, it is expected that the benchmark distribution of the s2-flavor machines has a larger variance than the distribution of the h1-flavor machines. The HS06 score distribution of the two flavours is shown in Figure 8.3, and their mean, minimum, maximum, and variance are shown in Table 8.2. For each flavor 1000 HS06 benchmarks were performed.

The mean of the HS06 score distribution for the h1-flavor virtual machine is about 4% higher than the mean score of the s2-flavour virtual machines. Other than expected, the HS06 score distribution of the s2-flavour virtual machines has a smaller variance than the machines of the h1-flavour.

The two virtual machine flavours are designed for different purposes. OTC has different types of physical machines. Because each type provides only a few virtual machine flavors, s2-flavor runs on other physical machines than h1-flavor virtual machines. The s2-flavor is for "general purpose" where OTC provides much more instances/resources than for more specialized flavors. The h1-flavor is designed for "high performance" application which also includes a low-latency and higher bandwidth network connection compared to the s1-flavor. [101]

Due to their special purpose, h1-flavor machines are available in small numbers



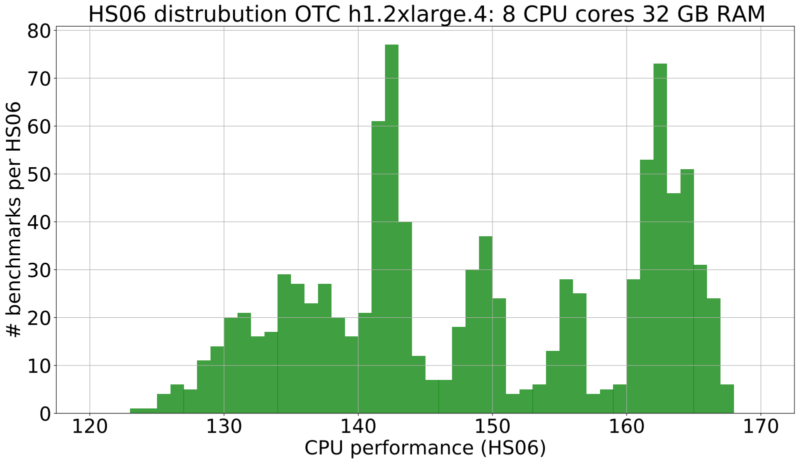


Figure 8.3: HS06 score distribution of s2-flavour and h1-flavour virtual machines at Open-TelekomCloud.

and provide more computing power than s2-flavor virtual machines. During the benchmark phase, all physical machines for the h1-flavor virtual machines were temporarily fully occupied. Thereby, the virtual machines influence each other. This would result in a greater variance distribution of HS06 scores as seen in Figure 8.3 (bottom).

Figure 8.4 shows the HS06 score distribution for each of the virtual machines of s2-flavor and h1-flavor. The HS06 score distributions of the s2-flavor virtual machines are similar to each other which means that the computing power of the physical machines where these virtual machines run is equal. The HS06 score distributions of the h1-flavor virtual machines looks different than the distributions of the s2-flavor. Some distributions of the virtual machines have several peaks, which result from changed CPU usage of other virtual machines on the same physical machine over time.

Due to these high variances in the computing performance of resources provided by commercial cloud providers, it is necessary to run benchmarks before and while using and paying for these. These benchmarks also show that commercial cloud resources are more expensive than usual Grid sites. It would cost about 21 mil. € per year to obtain the same amount of pledged resources for the WLCG collaborations (343.155 HS06) that GridKa provided in 2020 without including additional costs such as network traffic, storage, and personnel. This amount of money is in the same magnitude as the operating costs at GridKa.

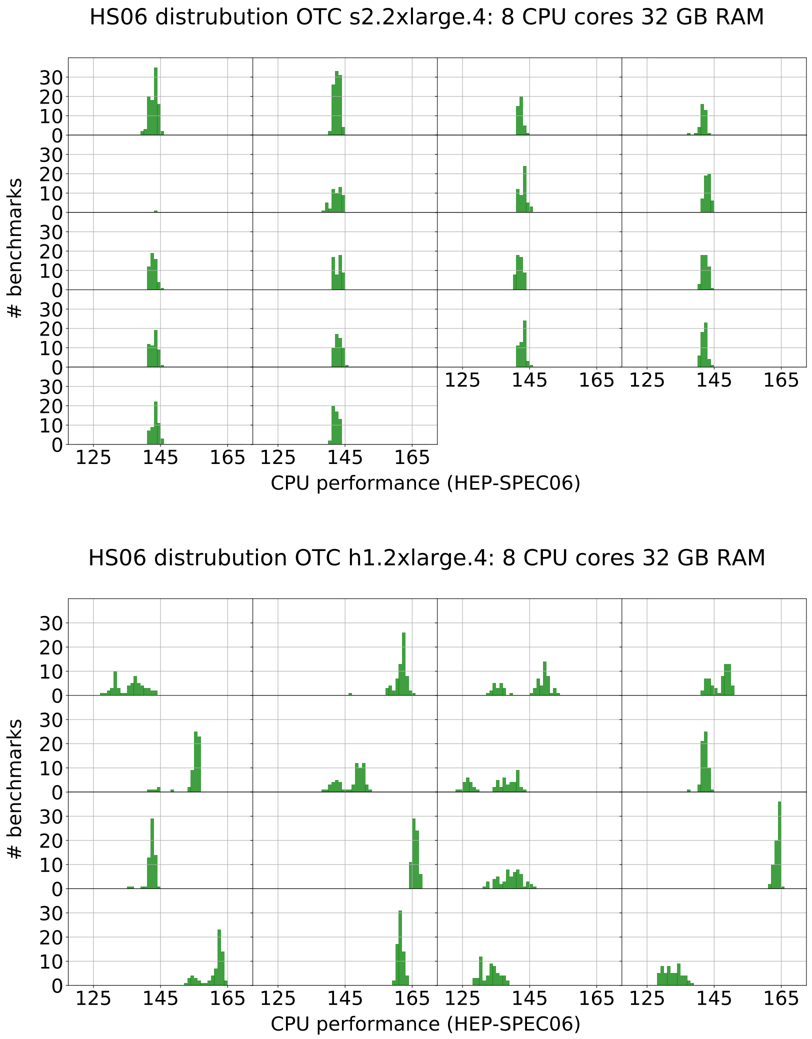


Figure 8.4: HS06 score distribution of s2-flavour and h1-flavour per virtual machine at OpenTelekomCloud.

Chapter 9

Conclusion

This thesis presents the triple differential Z+jet cross-section measurement based on data recorded by the Compact Muon Solenoid (CMS) detector in 2017. Thereby, detector effects are corrected via unfolding to yield a detector independent measurement. The cross-section is measured in bins of three observables: the difference and sum of the rapidity of the Z boson and the jet with the highest transverse momentum, as well as the transverse momentum of the Z boson. These variables are optimal to reduce the uncertainties in parton distribution functions (PDFs). This measurement is compared to theory prediction at leading order (LO) and next-to-leading order (NLO) accuracy. As expected, the NLO prediction is closer to the measured cross-section than the LO prediction. However, differences between the NLO prediction and measured cross-section that are not covered by the systematic and statistical uncertainties are observed, see Appendix B. The differences indicate necessary corrections on the PDFs. Therefore, this measurement can help to improve and further constrain PDFs.

More data, and improved theory predictions can reduce uncertainties of such cross-measurements further. With the planned upgrade of the Large Hadron Collider (LHC) to the High Luminosity LHC (HL-LHC) the amount of data will increases enormously and the statistical uncertainty of such cross-section can be reduced. However, the computing demand for High Energy Physics (HEP) analyses will also increase enormously due to the increasing data rate provided by the HL-LHC and due to the more complex event topolgies requiring more reconstruction efforts. Furthermore, the computing demand will increase to provide more precise theory predictions.

Additional computing resources that are not dedicated to HEP, such as High-Performance Computing (HPC) clusters or commercial cloud providers, can mitigate the increasing demand. Although these additional resources improve the situation, they also introduce some challenges. One of these challenges is the dynamic and transparent integration of such heterogeneous resources. Considering this, the `drone` concept has been developed. The `drone` concept enables the integration of various types of resources by providing a solution to transparently provision the required software environment.

Due to the integration of various resources from different providers, the heterogeneity of resource pool increases. A management system consisting of two components, `COBalD - the opportunistic Balancing Deamon` (`COBalD`) and **T**`ransparent` **A**`daptive` **R**`esource` **D**`ynamic` **I**`ntegration` **S**`ystem` (`TARDIS`), has been developed to handle such heterogeneous resource pools. This resource management system uses a feedback loop approach that enables it to react dynamically to changing demand. Therefore, `COBalD` and `TARDIS` release resources that are badly utilized and request further well-utilized resources. Furthermore, the resource management system was extended to react to insufficient network bandwidth.

Another challenge is the estimation of the CPU-performance these resources provide. Therefore, benchmarks were performed. These benchmarks showed that the CPU-performance is not constant over time on shared systems such as HPC clusters or resources by commercial cloud providers. That results from the variable utilization of the system caused by other users.

The `drone` concept and the resource management system `COBalD` and `TARDIS` enable to integrate resources transparently into an existing batch system. Following this path, more than 6000 CPU-cores were usable for the Institute of Experimental Particle Physics (ETP) members, without being actively supplied by the institute, see Figure 6.1. Furthermore, with the same concept about 4500 additional CPU-cores have been made available to the Belle II and LHC collaborations via GridKa, see Figure 6.6. For this, GridKa provides an single point of entry and a batch system instance for dynamically integrated resources. These resources are provided by different institutes in Germany, such as the University of Bonn, the Leibniz Supercomputing Centre in Munich, and KIT. Also, additional computing resources from commercial cloud providers were integrated during the European `Helix Nebula Science Cloud` project. Although the development of the `drone` concept and `COBalD`/`TARDIS` has been started in the HEP community, the concept is applicable for other scientific communities. For example, KIT provided was able to provide resources to the microbiology community during the COVID19 pandemic.

Appendix A

Data-MC Comparison

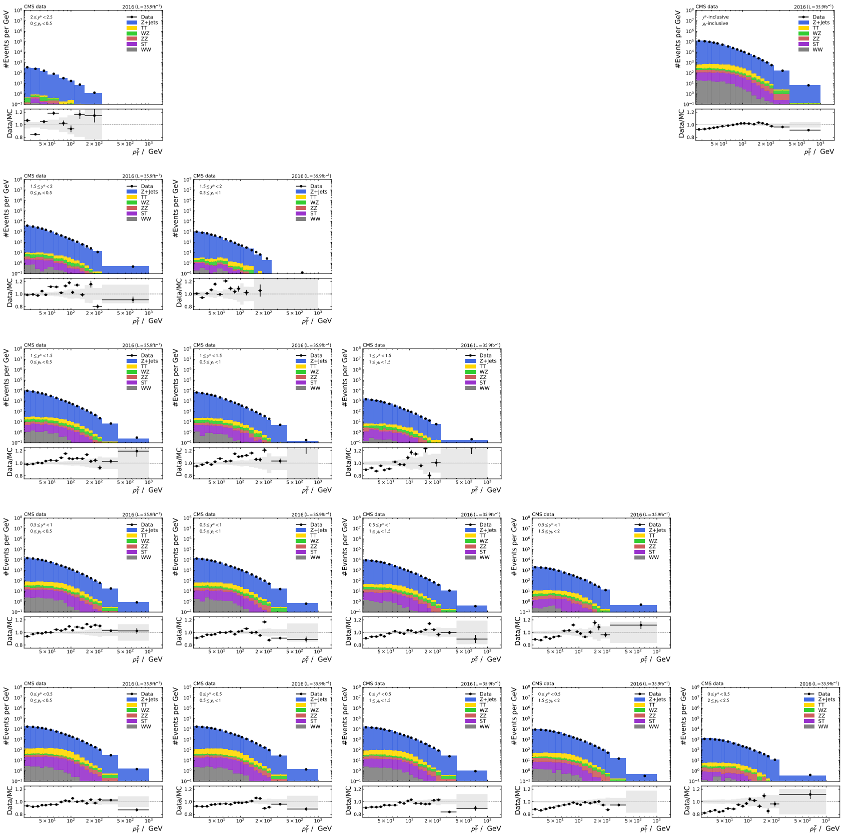


Figure A.1: Data-MC comparison of 2016 datasets with 2016 cuts. Data corrected due to detector effects and scaled according the selection efficiency in the MC.

Table A.1: List of used datasets in the analysis by Thomas Berger of the data recoded in 2016

Dataset	**Size (TB)**	**# events**
Data		
/SingleMuon/Run2016[B-H]-17Jul2018-v1/MINIAOD	16.2	806.815.953
Signal MC		
/DYJetsToLL_M-50_TuneCUETP8M1_13TeV-madgraphMLM-pythia8/[...]_ext2-v1/MINIAODSIM	2.9	96.658.943
Background MC		
/TTJets_TuneCUETP8M1_13TeV-madgraphMLM-pythia8/[...]-v1/MINIAODSIM	0.5	10.139.950
/WZJToLLLNu_TuneCUETP8M1_13TeV-amcnlo-pythia8/[...]-v1/MINIAODSIM	0.6	1.930.828
/ZZ_TuneCUETP8M1_13TeV-pythia8/[...]_ext1-v1/MINIAODSIM	0.03	998.034
/WZJToLLLNu_TuneCUETP8M1_13TeV-amcnlo-pythia8/[...]-v1/MINIAODSIM	0.6	1.999.000
/ST_tW_antitop_5f_inclusiveDecays_13TeV-powheg-pythia8_TuneCUETP8M1/[...]_ext1-v1/MINIAODSIM	0.3	6.933.094
/ST_tW_top_5f_inclusiveDecays_13TeV-powheg-pythia8_TuneCUETP8M1/[...]_ext1-v1/MINIAODSIM	0.3	6.952.830

[...]=RunIISummer16MiniAODv2-PUMoriond17_80X_mcRun2_asymptotic_2016_TrancheIV_v6

Table A.2: List of used datasets in the updated analysis of the data recoded in 2016

Dataset	**Size (TB)**	**# events**
Data		
/SingleMuon/Run2016[B-H]-17Jul2018-v1/MINIAOD	16.2	806.815.953
Signal MC		
/DYJetsToLL_M-50_TuneCUETP8M1_13TeV-amcatnloFXFX-pythia8/[...]_ext2-v1/MINIAODSIM	3.5	120.777.245
Background MC		
/TTJets_TuneCUETP8M2T4_13TeV-amcatnloFXFX-pythia8/[...]-v2/MINIAODSIM	1.9	43.845.135
/WZ_TuneCUETP8M1_13TeV-pythia8/[...]_ext1-v2/MINIAODSIM	0.09	2.997.571
/ZZ_TuneCUETP8M1_13TeV-pythia8/[...]_ext1-v2/MINIAODSIM	0.03	998.034
/WW_TuneCUETP8M1_13TeV-pythia8/[...]-v1/MINIAODSIM	0.2	6.988.168
/ST_tW_antitop_5f_inclusiveDecays_13TeV-powheg-pythia8_TuneCUETP8M1/[...]_ext1-v1/MINIAODSIM	0.3	6.933.094
/ST_tW_top_5f_inclusiveDecays_13TeV-powheg-pythia8_TuneCUETP8M1/[...]_ext1-v1/MINIAODSIM	0.3	6.952.830
/ST_t-channel_antitop_4f_InclusiveDecays_TuneCP5_PSweights_13TeV-powheg-pythia8/[...]-v1/MINIAODSIM	0.7	17.780.700
/ST_t-channel_top_4f_InclusiveDecays_TuneCP5_PSweights_13TeV-powheg-pythia8/[...]v1/MINIAODSIM	1.2	31.848.000

[...]=RunIISummer16MiniAODv3-PUMoriond17_94X_mcRun2_asymptotic_v3

Table A.3: List of used datasets for the analysis of the data recorded 2017

Dataset	**Size (TB)**	**# events**
Data		
/SingleMuon/Run2017[B-F]-31Mar2018-v1/MINIAOD	22.9	769,080,716
Signal MC		
/DYJetsToLL_M-50_TuneCP5_13TeV-amcatnloFXFX-pythia8/[...]_ext3-v1/MINIAODSIM	7.7	187,128,994
/DYJetsToLL_M-50_TuneCP5_13TeV-madgraphMLM-pythia8/RunIIFall17MiniAODv2-PU2017RECOSIMstep_12Apr2018_94X_mc2017_realistic_v14_ext1-v1/MINIAODSIM	2.0	49.125.561
Backgrounds MC		
/TTJets_DiLept_TuneCP5_13TeV-madgraphMLM-pythia8/[...]-v1/MINIAODSIM	1.6	28,380,110
/WZ_TuneCP5_13TeV-pythia8/[...]-v1/MINIAODSIM	0.16	3,928,630
/ZZ_TuneCP5_13TeV-pythia8/[...]-v1/MINIAODSIM	0.01	1,925,931
/WW_TuneCP5_13TeV-pythia8/[...]-v1/MINIAODSIM	0.3	7,765,828
/ST_t-channel_antitop_4f_inclusiveDecays_TuneCP5_13TeV-powhegV2madspin-pythia8/[...]-v2/MINIAODSIM	0.17	3,675,910
/ST_t-channel_top_4f_inclusiveDecays_TuneCP5_13TeV-powhegV2madspin-pythia8/[...]-v2/MINIAODSIM	0.27	5865875
/ST_tW_antitop_5f_inclusiveDecays_TuneCP5_13TeV-powheg-pythia8/[...]-v2/MINIAODSIM	0.4	7,977,430
/ST_tW_top_5f_inclusiveDecays_TuneCP5_13TeV-powheg-pythia8/[...]-v2/MINIAODSIM	0.4	7,794,186

[...]=RunIIFall17MiniAODv2-PU2017_12Apr2018_94X_mc2017_realistic_v14

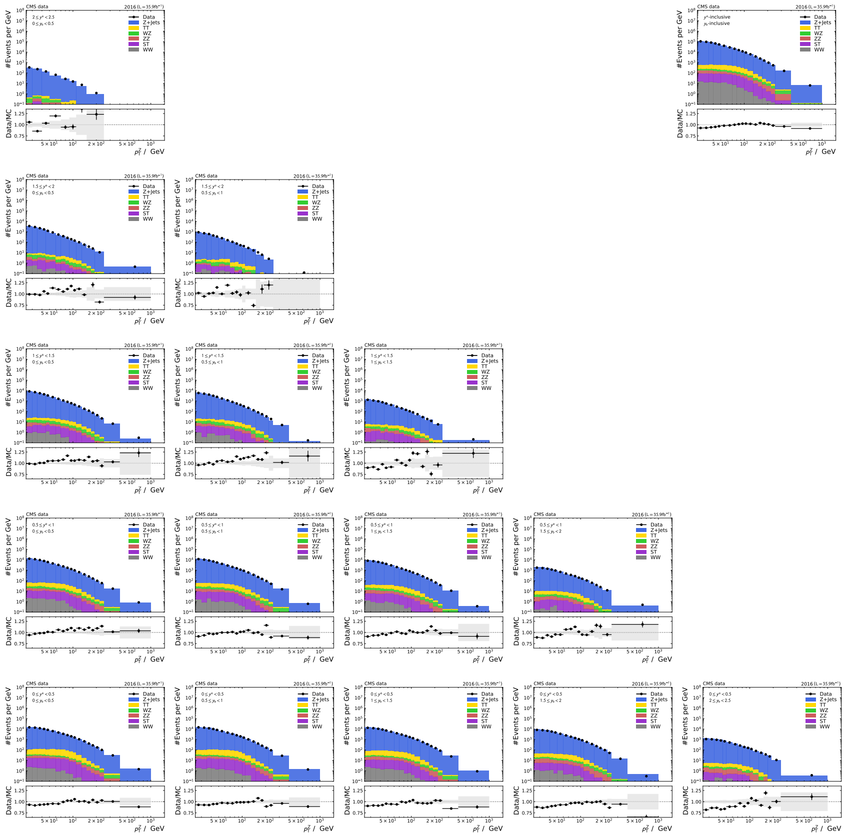


Figure A.2: Data-MC comparison of 2016 datasets with 2017 cuts. Data corrected due to detector effects and scaled according the selection efficiency in the MC.

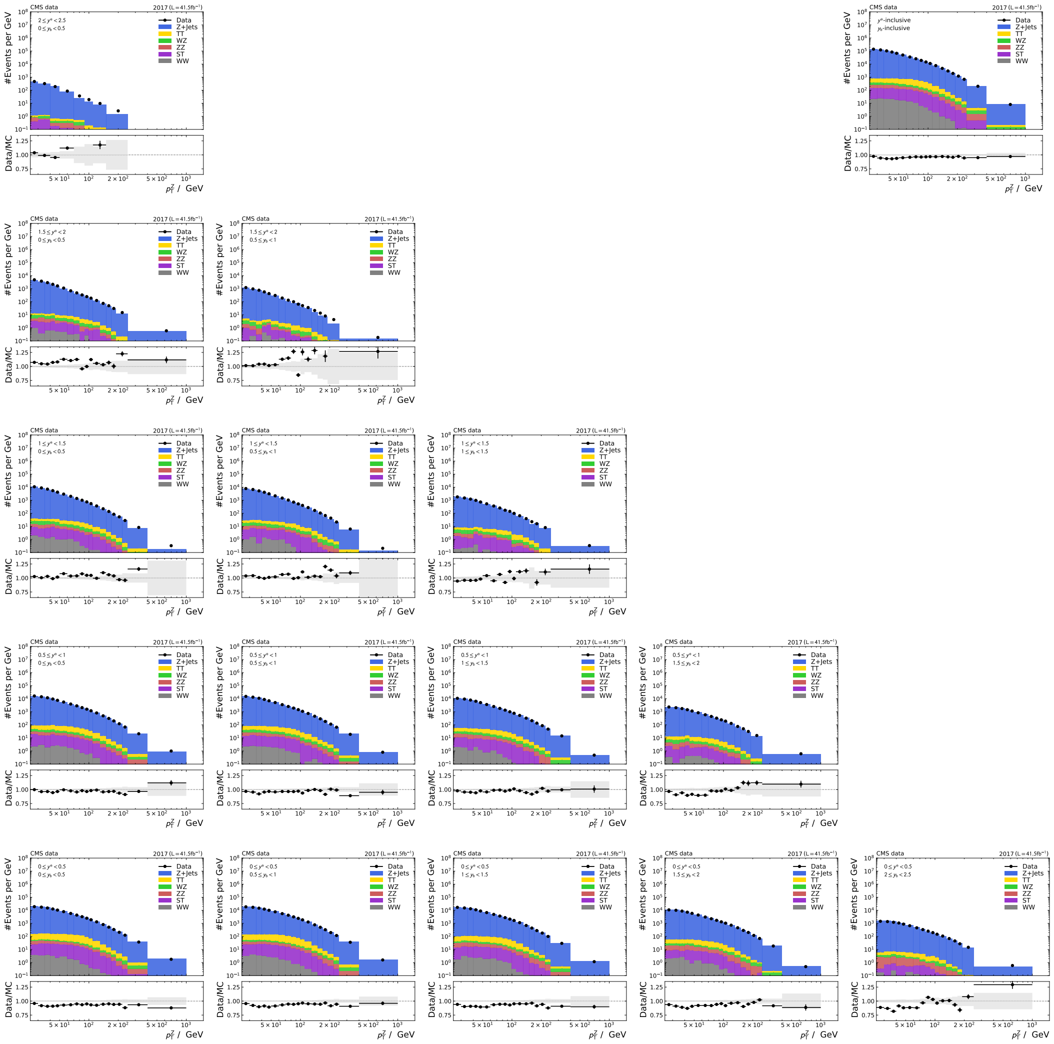


Figure A.3: Data-MC comparison of 2017 datasets. Data corrected due to detector effects and scaled according the selection efficiency in the MC.

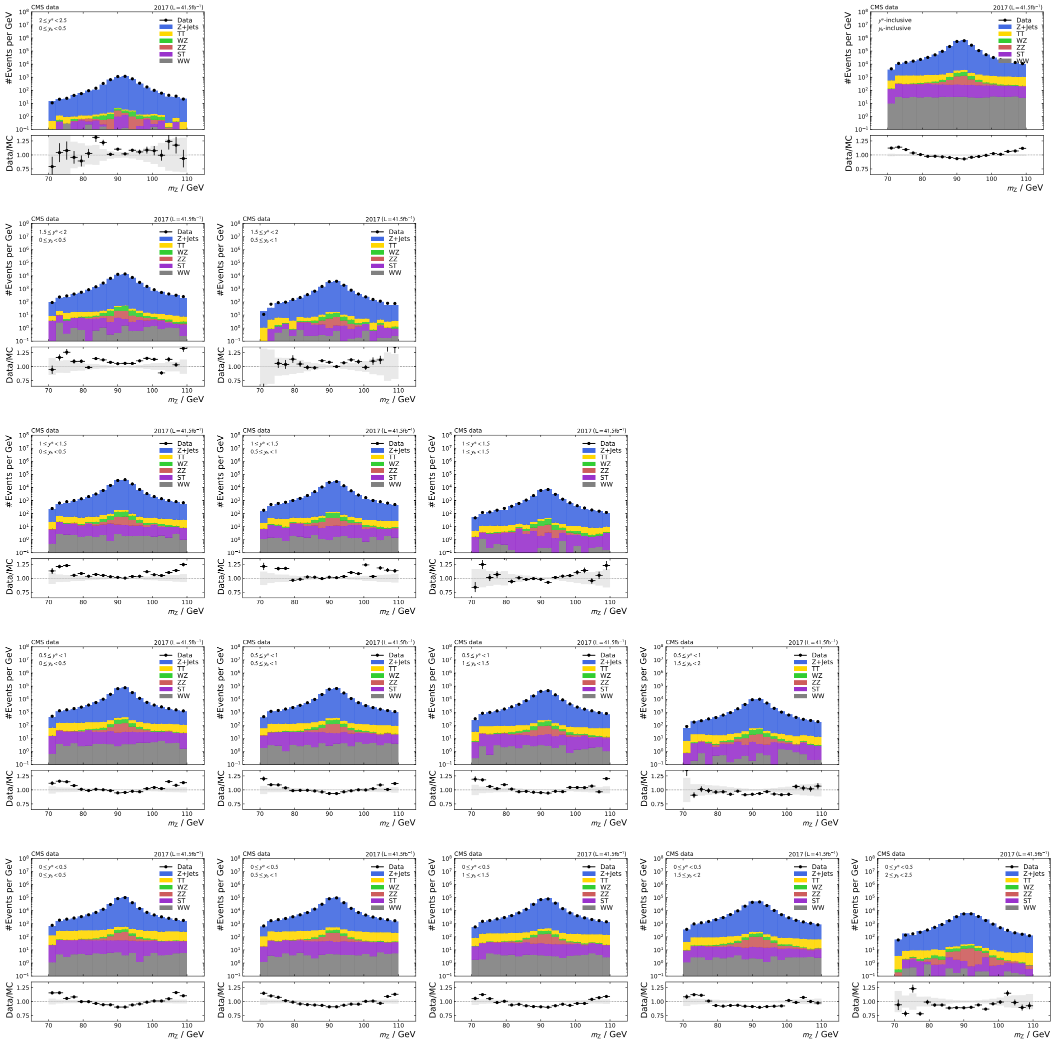


Figure A.4: Data-MC comparison of 2017 Z_{mass} distribution. Data corrected due to detector effects and scaled according the selection efficiency in the MC.

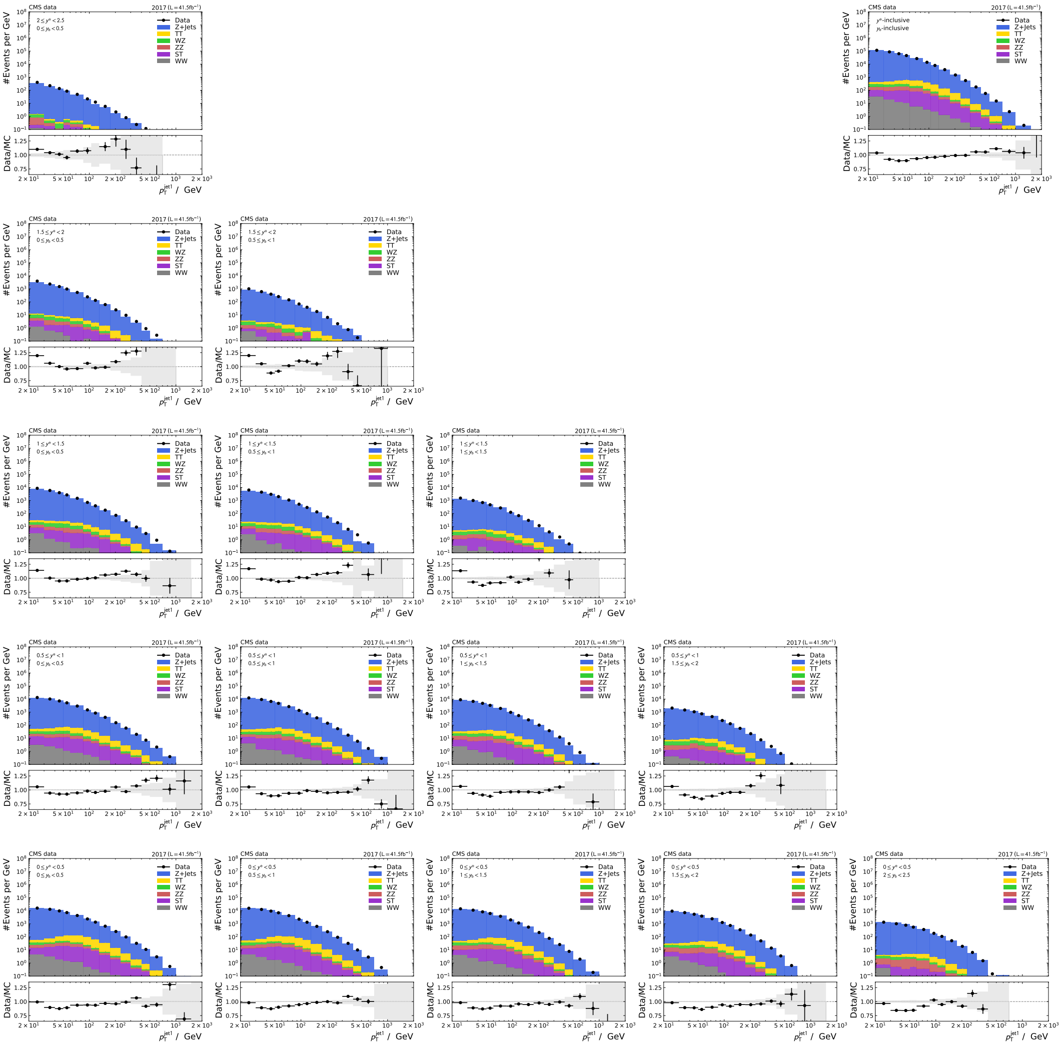


Figure A.5: Data-MC comparison of 2017 p_T^{jet1} distribution. Data corrected due to detector effects and scaled according the selection efficiency in the MC.

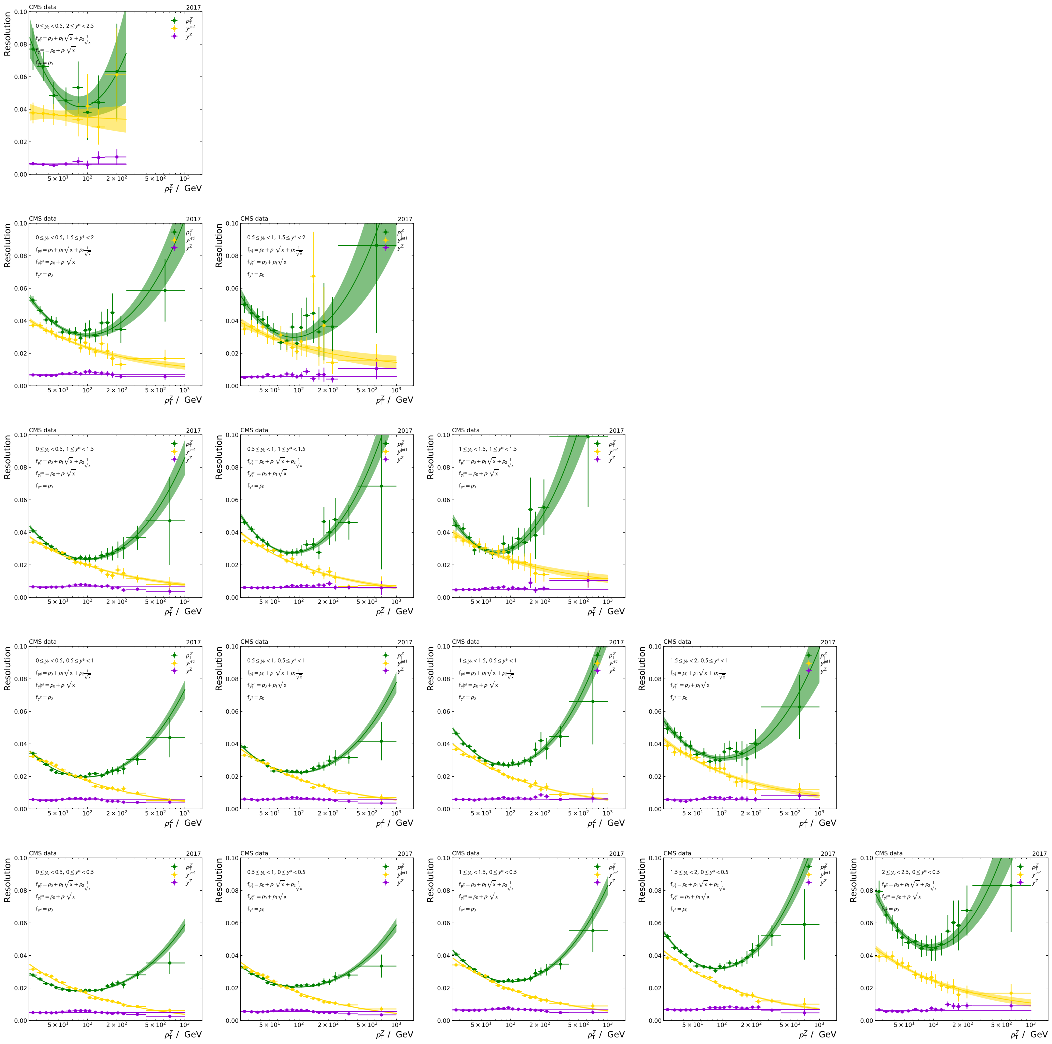


Figure A.6: Shown is the resolution of y^{Z}, y^{jet1}, and $p_{\mathrm{T}}^{\mathrm{Z}}$ determined from full simulation via truncated root mean square (RMS). A function is fitted to the data point to reduce the uncertainty of the resolutions. In some y_{b} y^* bins the uncertainty of the fit covers the uncertainty of the data point, If the $\chi^2/N.D.F) > 1.5$ the fit uncertainty is scaled by $\sqrt{\chi^2/N.D.F}$. The plots show the scaled fit uncertainty.

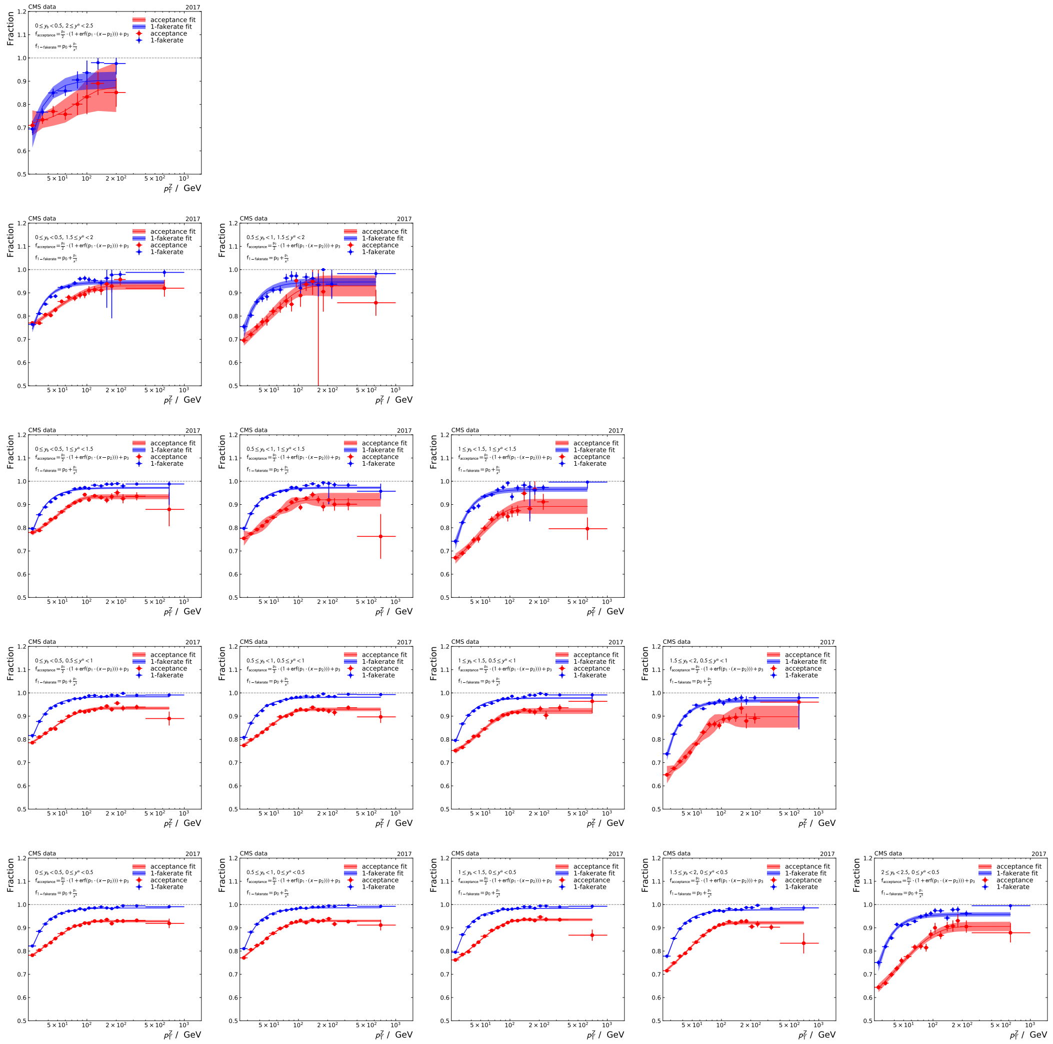


Figure A.7: Shown is the resolution of y^{Z}, y^{jet1}, and $p_{\mathrm{T}}^{\mathrm{Z}}$ determined from full simulation via truncated RMS. A function is fitted to the data point to reduce the uncertainty of the resolutions. In some y_{b} y^* bins the uncertainty of the fit covers the uncertainty of the data point, If the $\chi^2/N.D.F)$ > 1.5 the fit uncertainty is scaled by $\sqrt{\chi^2/N.D.F}$. The plots show the scaled fit uncertainty.

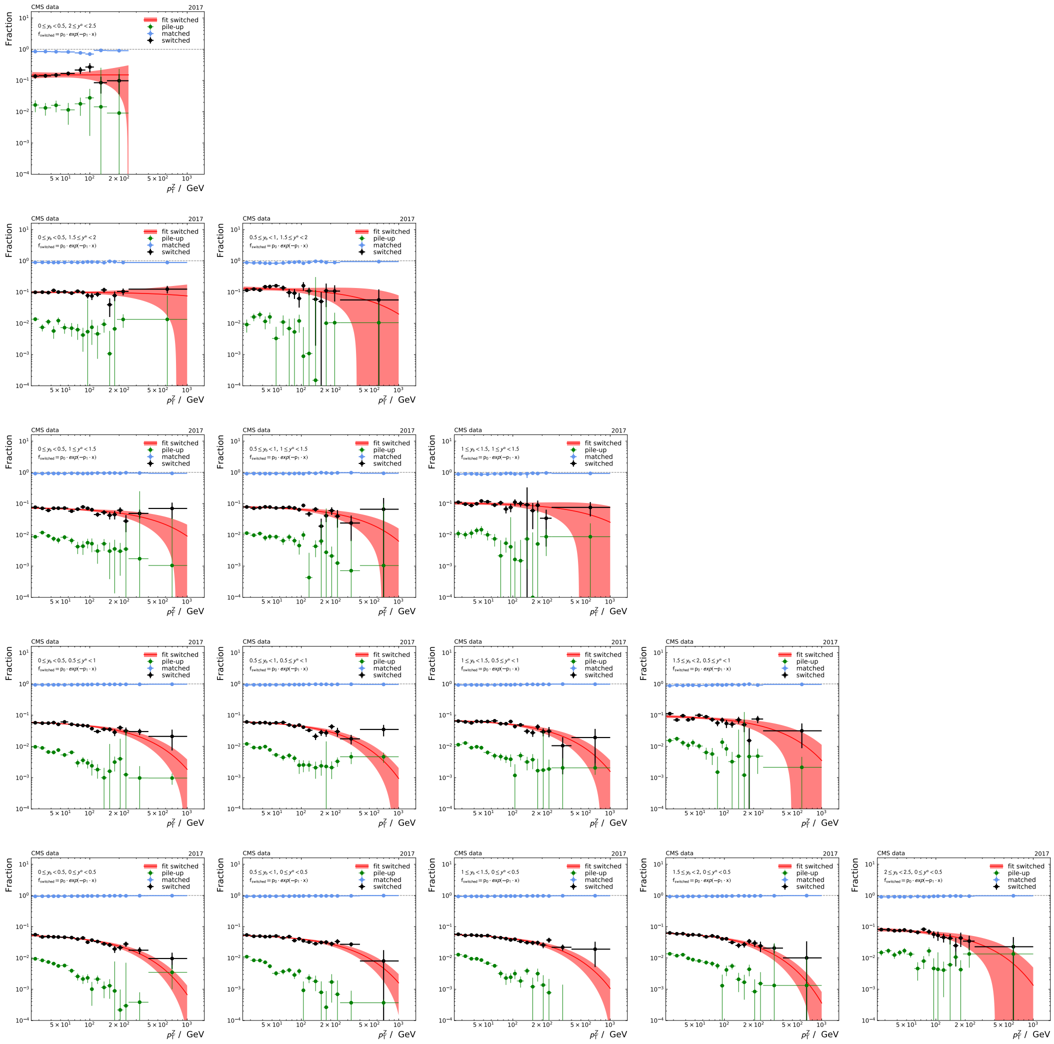


Figure A.8: Fraction of reconstructed leading jet is generated leading jet (matched), reconstructed leading jet is a cone of R = 0.3 around a generated jet but not the generated leading jet(switching), and no generator jet is within a cone of R = 0.3 around the reconstructed leading jet(pile-up) based on 2017 MC. The fraction of pile-up is negligible. Therefore, only the fraction of switching events is necessary for forward smearing. The estimated switching probability with uncertainty used for forward smearing is fitted to the fraction of switching events.

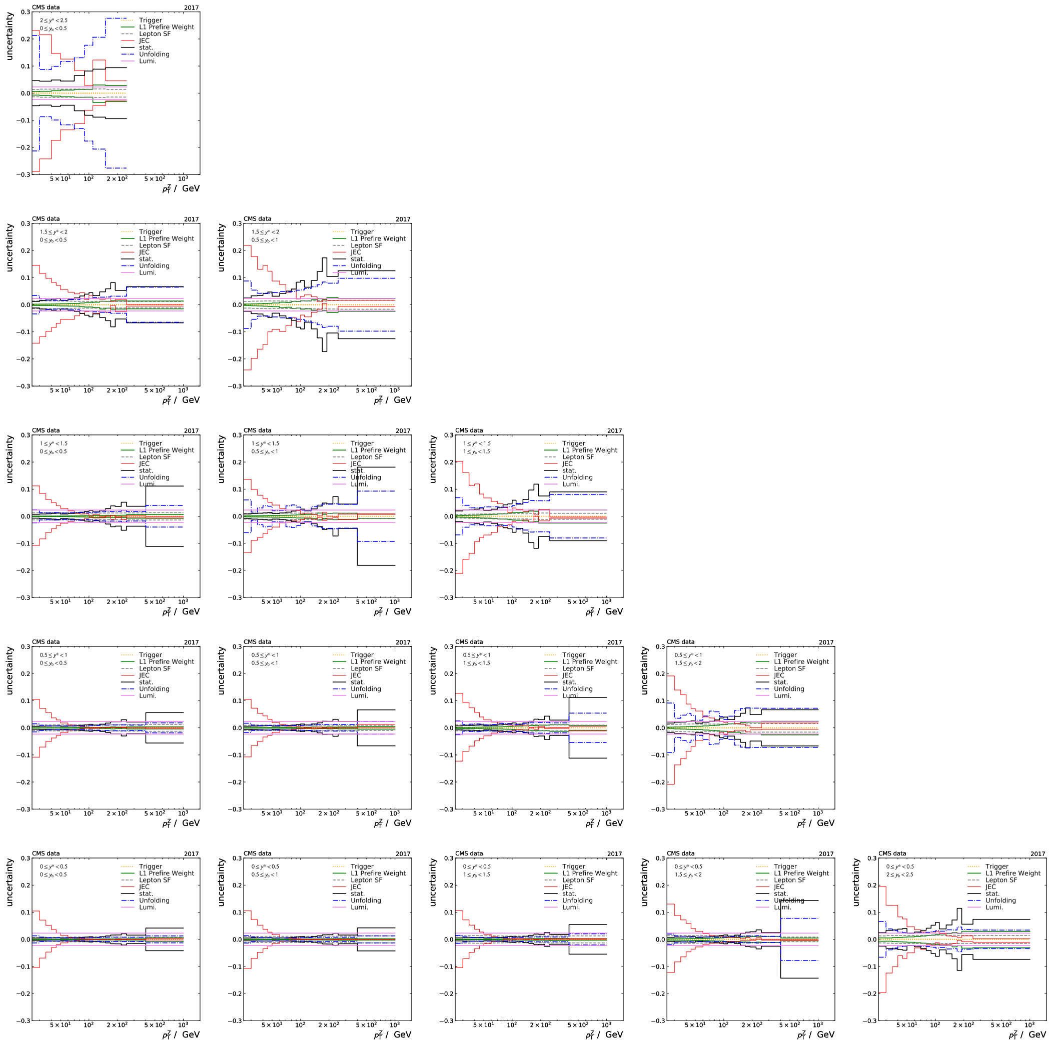


Figure A.9: Shown is the systematical and statistical uncertainties of the triple differential Z+jet cross-section based on data recorded 2017.

Appendix B

Comparison of measured and predicted cross-section

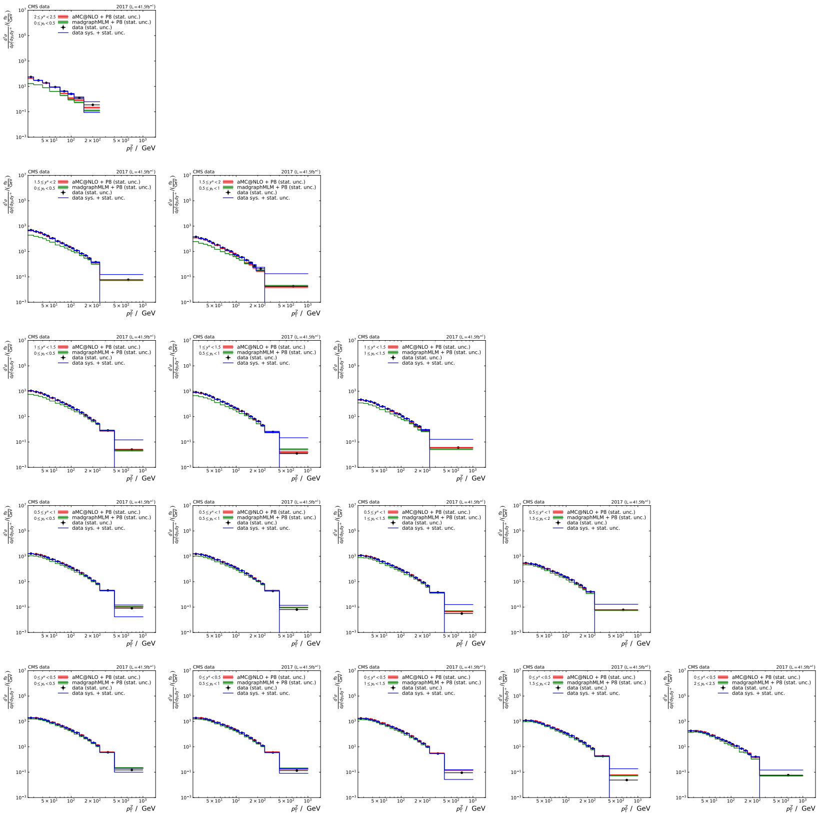


Figure B.1: Triple differential Z+jet cross-section based on 2017 data (black) with statistical and systematic uncertainties (blue) is shown. The theory prediction is based on the simulated signal sample on generator level with the stat. uncertainty of the sample at LO (green) and NLO (red) accuracy.

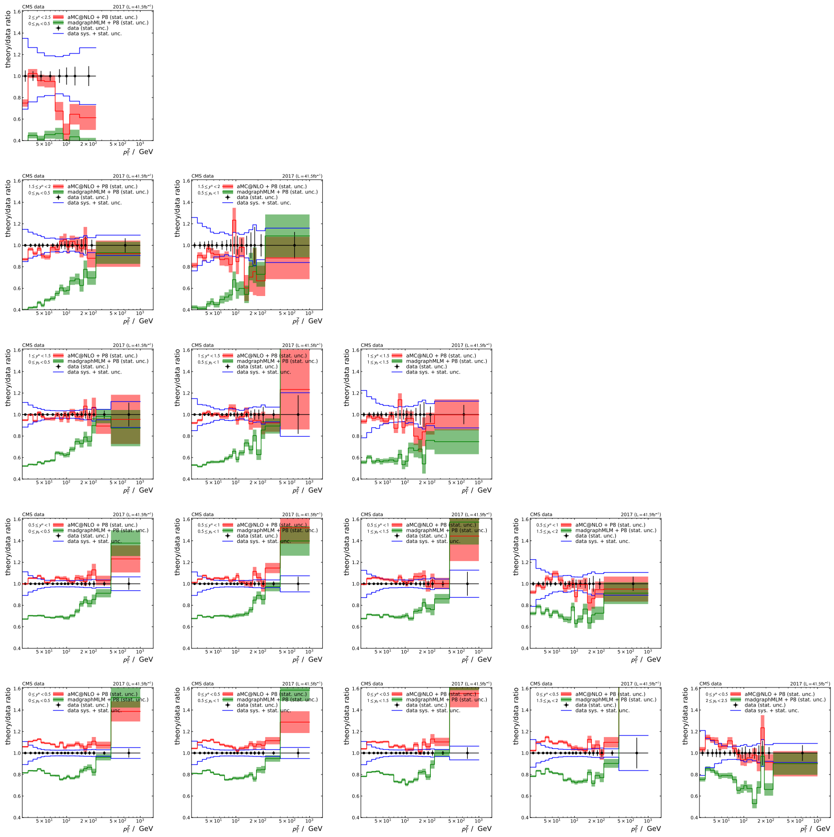


Figure B.2: Triple differential Z+jet cross-section ratio between 2017 data (black) with statistical and systematic uncertainties (blue) and simulated signal sample on generator level at LO (green) and NLO (red) accuracy is shown.

Appendix C

Resource Scheduling with TARDIS and HTCondor Configuration consider CPU-efficiency

Additional configuration for HTCondor to determine the average CPU-efficiency per HTCondor worker node (startd):

```
AverageCPUsUsage = Sum(My.ChildCPUsUsage)/Sum(My.ChildCPUs)
STARTD_PARTITIONABLE_SLOT_ATTRS = $(
    STARTD_PARTITIONABLE_SLOT_ATTRS), CPUsUsage
STARTD_ATTRS = $(STARTD_ATTRS), AverageCPUsUsag
```

Additional TARDIS configuration to consider network bandwidth via CPU-efficiency:

```
cpu_usage: IfThenElse(AverageCPUsUsage=?=undefined, 0, Real(
    AverageCPUsUsage))
```

List of Figures

List of Tables

Bibliography

[1] John C. Collins, Davison E. Soper and George F. Sterman, "Factorization of Hard Processes in QCD", *Adv. Ser. Direct. High Energy Phys.* **5** (1989) 1–91, DOI: 10.1142/9789814503266_0001, [arXiv:hep-ph/0409313].

[2] A. Einstein, "Über einen die Erzeugung und Verwandlung des Lichtes betreffenden heuristischen Gesichtspunkt", *Annalen der Physik* **322** (1905) 132–148, DOI: https://doi.org/10.1002/andp.19053220607, eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/andp.19053220607.

[3] C. S. Wu et al., "Experimental Test of Parity Conservation in Beta Decay", *Phys. Rev.* **105** (1957) 1413–1415, DOI: 10.1103/PhysRev.105.1413.

[4] Abdus Salam and John Clive Ward, "Electromagnetic and weak interactions", *Phys. Lett.* **13** (1964) 168–171, DOI: 10.1016/0031-9163(64)90711-5.

[5] P.A. Zyla et al., "Review of Particle Physics", *PTEP* (2020) p. 083C01, DOI: 10.1093/ptep/ptaa104.

[6] Yuri L. Dokshitzer, "Calculation of the Structure Functions for Deep Inelastic Scattering and e+ e- Annihilation by Perturbation Theory in Quantum Chromodynamics.", *Sov. Phys. JETP* **46** (1977) 641–653.

[7] V.N. Gribov and L.N. Lipatov, "Deep inelastic e p scattering in perturbation theory", *Sov. J. Nucl. Phys.* **15** (1972) 438–450.

[8] G. Altarelli and G. Parisi, "Asymptotic freedom in parton language", *Nuclear Physics B* **126** (1977) 298–318, DOI: https://doi.org/10.1016/0550-3213(77)90384-4.

[9] Sayipjamal Dulat et al., "New parton distribution functions from a global analysis of quantum chromodynamics", *Phys. Rev. D* **93** (2016) p. 033006, DOI: 10.1103/PhysRevD.93.033006, [arXiv:1506.07443].

[10] L. Del Debbio et al., "The Neural network approach to parton distributions: HERA - LHC workshop proceedings", *HERA and the LHC: A Workshop on the Implications of HERA and LHC Physics (Startup Meeting, CERN, 26-27 March 2004; Midterm Meeting, CERN, 11-13 October 2004)*, 2005, [arXiv:hep-ph/0509059].

[11] Luigi Del Debbio, "Parton distributions in the LHC era", *EPJ Web of Conferences* **175** (2018) p. 01006, DOI: 10.1051/epjconf/201817501006.

[12] John Collins, *Foundations of perturbative QCD*, Cambridge monographs on particle physics, nuclear physics, and cosmology, New York, NY: Cambridge Univ. Press, 2011, DOI: 10.1017/CBO9780511975592.

[13] John C. Collins, Davison E. Soper and George Sterman, "Factorization of Hard Processes in QCD", *Perturbative QCD*, 1–91, DOI: 10.1142/9789814503266_0001, eprint: https://www.worldscientific.com/doi/pdf/10.1142/9789814503266_0001.

[14] Klaus Rabbertz et al., "High precision calculations of particle physics at the NEMO cluster in Freiburg" (2018), DOI: 10.15496/publikation-25203.

[15] Esma Mobs, "The CERN accelerator complex. Complexe des accélérateurs du CERN" (2016), URL: https://cds.cern.ch/record/2197559.

[16] Oliver Sim Brüning et al., *LHC Design Report*, CERN Yellow Reports: Monographs, Geneva: CERN, 2004, DOI: 10.5170/CERN-2004-003-V-1.

[17] Jorg Wenninger, "Operation and Configuration of the LHC in Run 2" (2019), URL: https://cds.cern.ch/record/2668326.

[18] G. Apollinari et al., "Chapter 1: High Luminosity Large Hadron Collider HL-LHC. High Luminosity Large Hadron Collider HL-LHC", *CERN Yellow Report* (2017) 1–19. 21 p, DOI: 10.5170/CERN-2015-005.1.

[19] HL-LHC project, accessed 05.11.2020, URL: https://hilumilhc.web.cern.ch/content/hl-lhc-project.

[20] The ALICE Collaboration, "The ALICE experiment at the CERN LHC", *Journal of Instrumentation* **3** (2008) S08002–S08002, DOI: 10.1088/1748-0221/3/08/s08002.

[21] The LHCb Collaboration, "The LHCb Detector at the LHC", *JINST* **3** (2008) S08005, DOI: 10.1088/1748-0221/3/08/S08005.

[22] The ATLAS Collaboration, "ATLAS: letter of intent for a general-purpose pp experiment at the large hadron collider at CERN" (1992), URL: http://cds.cern.ch/record/291061.

[23] The CMS Collaboration, "The CMS experiment at the CERN LHC", *Journal of Instrumentation* **3** (2008) S08004–S08004, DOI: 10.1088/1748-0221/3/08/s08004.

[24] David Barney and Sergio Cittolin, "CMS Detector Drawings" (2000), URL: https://cds.cern.ch/record/2629816.

[25] CMS Collaboration, *CMS Physics: Technical Design Report Volume 1: Detector Performance and Software*, Technical Design Report CMS, Geneva: CERN, 2006, URL: https://cds.cern.ch/record/922757.

[26] Izaak Neutelings, accessed 26.11.2020, 2017, URL: https://wiki.physik.uzh.ch/cms/latex:example_spherical_coordinates.

[27] *CMS subdetectors*, "accessed 17.11.2020", URL: https://twiki.cern.ch/twiki/bin/view/CMSPublic/WorkBookCMSExperiment?rev=40.

[28] accessed 18.11.2020, URL: https://twiki.cern.ch/twiki/bin/view/CMSPublic/LumiPublicResults.

[29] *The Phase-2 Upgrade of the CMS Tracker*, tech. rep., Geneva: CERN, 2017, URL: https://cds.cern.ch/record/2272264.

[30] Martina Ressegotti, "Overview of the CMS Detector Performance at LHC Run 2", *Universe* **5** (2019) p. 18, DOI: 10.3390/universe5010018.

[31] Thomas Berger, "Jet energy calibration and triple differential inclusive cross section measurements with Z ($\rightarrow \mu\mu$) + jet events at 13 TeV recorded by the CMS detector" (2019), DOI: 10.5445/IR/1000104286.

[32] The 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.

[33] CMS Physics Object Group, accessed: 28.1.2021, URL: https://twiki.cern.ch/twiki/bin/view/CMS/MuonReferenceEffsRun2.

[34] S. Dasu et al., "CMS. The TriDAS project. Technical design report, vol. 1: The trigger systems" (2000).

[35] Laurent Thomas, *Reweighting recipe to emulate Level 1 ECAL prefiring.* accessed 15. December 2020, URL: https://twiki.cern.ch/twiki/bin/view/CMS/L1ECALPrefiringWeightRecipe?rev=11.

[36] The 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.

[37] Matteo Cacciari, Gavin P Salam and Gregory Soyez, "The anti-ktjet clustering algorithm", *Journal of High Energy Physics* (2008) 063–063, DOI: 10.1088/1126-6708/2008/04/063.

[38] The 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.

[39] Taylor L. Alverson G, "Performance of the Particle-Flow jet identification criteria using proton-proton collisions at 13 TeV" (2016).

[40] *Pileup Jet Identification*, tech. rep., Geneva: CERN, 2013, URL: https://cds.cern.ch/record/1581583.

[41] J. Alwall et al., "The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations", *JHEP* **07** (2014) 079. 157 p, DOI: 10.1007/JHEP07(2014)079.

[42] Torbjörn Sjöstrand et al., "An introduction to PYTHIA 8.2", *Computer Physics Communications* **191** (2015) 159–177, DOI: 10.1016/j.cpc.2015.01.024.

[43] *CMS Luminosity Measurements for the 2016 Data Taking Period*, tech. rep., Geneva: CERN, 2017, URL: https://cds.cern.ch/record/2257069.

[44] https://twiki.cern.ch/twiki/bin/view/CMS/TWikiLUM [Online accessed: 29.11.2020], 2017.

[45] *CMS luminosity measurement for the 2017 data-taking period at* $\sqrt{s} = 13$ TeV, tech. rep., Geneva: CERN, 2018, URL: https://cds.cern.ch/record/2621960.

[46] The CMS collaboration, "Measurement of the production cross section for Z + b jets in proton-proton collisions at $\sqrt{s} = 13$ TeV" (2021), [arXiv:2112.09659].

[47] S Schmitt, "TUnfold, an algorithm for correcting migration effects in high energy physics", *Journal of Instrumentation* **7** (2012) T10003–T10003, DOI: 10.1088/1748-0221/7/10/t10003.

[48] Rene Brun et al., *root-project/root: v6.18/02*, version v6-18-02, 2019, DOI: 10.5281/zenodo.3895860.

[49] HEP Software Foundation, *HEP Software Foundation Community White Paper Working Group - Detector Simulation*, 2018, [arXiv:1803.04165].

[50] Odysseas I. Pentakalos, "An Introduction to the InfiniBand Architecture", *Int. CMG Conference*, 2002.

[51] T.A. Wassenaar et al., "WeNMR: Structural biology on the grid", *Journal of Grid Computing* **10** (2012) 743–767, DOI: 10.1007/s10723-012-9246-z.

[52] Ian Foster and Carl Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*, 2003.

[53] Alvise Dorigo et al., "XROOTD - A highly scalable architecture for data access", *WSEAS Transactions on Computers* **4** (2005) 348–353.

[54] W. Allcock et al., "The Globus Striped GridFTP Framework and Server", *SC '05: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing*, 2005, 54–54, DOI: 10.1109/SC.2005.72.

[55] *Folding@Home projekt*, accessed 28.2.2021, URL: https://foldingathome.org/.

[56] *Folding@Home: available resources 12. April 2020*, accessed 28.2.2021, URL: https://archive.vn/20200412111010/https://stats.foldingathome.org/os#selection-199.0-219.6.

[57] *Top500*, accessed 17.02.2021, URL: https://www.top500.org/.

[58] Johannes Albrecht et al., "A Roadmap for HEP Software and Computing R \& D for the 2020s", *Comput. Softw. Big Sci.* **3** (2019) p. 7, DOI: 10.1007/s41781-018-0018-8, [arXiv:1712.06982].

[59] *Estimated CPU requirements for the CMS collaboration*, https://twiki.cern.ch/twiki/bin/view/CMSPublic/CMSOfflineComputingResults?rev=12 [Online accessed: 19.11.2020], 2020.

[60] I Bird et al., *Update of the Computing Models of the WLCG and the LHC Experiments*, tech. rep., 2014, URL: https://cds.cern.ch/record/1695401.

[61] Burt Holzman et al., "HEPCloud, a New Paradigm for HEP Facilities: CMS Amazon Web Services Investigation", *Comput. Softw. Big Sci.* **1** (2017) p. 1, DOI: 10.1007/s41781-017-0001-9, [arXiv:1710.00100].

[62] Schnepf, Matthias J. et al., "Dynamic Integration and Management of Opportunistic Resources for HEP", *EPJ Web Conf.* **214** (2019) p. 08009, DOI: 10.1051/epjconf/201921408009.

[63] *ATLAS site setup and configuration*, accessed 17.01.2021, URL: https://twiki.cern.ch/twiki/bin/view/AtlasComputing/SitesSetupAndConfiguration?rev=58.

[64] *New CMS compute site*, accessed 17.01.2021, URL: https://twiki.cern.ch/twiki/bin/view/CMSPublic/NewComputingSite?rev=8.

[65] *OpenScienceGrid Worker Node Overview*, accessed 17.01.2021, URL: https://opensciencegrid.org/docs/worker-node/using-wn/.

[66] http://linux.web.cern.ch/linux/scientific6/ [Online accessed: 24.11.2016], 2016.

[67] *Announcement of the End of support for SLC6 (Scientific Linux CERN 6)*, accessed 17.01.2021, URL: https://linux.web.cern.ch/#announcement-of-the-end-of-support-for-slc6-scientific-linux-cern-6.

[68] https://www.openstack.org/ [Online accessed: 02.01.2017], 2017.

[69] Konrad Meier, "Infrastrukturkonzepte für virtualisierte wissenschaftliche Forschungsumgebungen" (2017), DOI: 10.6094/UNIFR/14873.

[70] http://www.docker.com [Online accessed: 04.01.2019], 2020.

[71] Matthias Jochen Schnepf, "Calculation of cross-section limits for the production of single top quarks in association with a Higgs boson using container technologies", Karlsruhe Institute of Technology (KIT), 2017.

[72] Gregory M. Kurtzer, Vanessa Sochat and Michael W. Bauer, “Singularity: Scientific containers for mobility of compute”, *PLOS ONE* **12** (2017) 1–20, DOI: 10.1371/journal.pone.0177459.

[73] http://wlcg-public.web.cern.ch/about [Online accessed: 25.09.2020], 2020.

[74] K. Bos et al., *LHC computing Grid: Technical Design Report. Version 1.06 (20 Jun 2005)*, Technical Design Report LCG, Geneva: CERN, 2005, URL: https://cds.cern.ch/record/840543.

[75] *GridKa website*, accessed 17.11.2020, URL: http://www.gridka.de.

[76] “Facilities & Services, Documentation” (), URL: https://twiki.cern.ch/twiki/bin/view/CMSPublic/FacilitiesServicesDocumentation?rev=53.

[77] http://www.scc.kit.edu/en/aboutus/13531.php, 2020.

[78] I Sfiligoi, “glideinWMS—a generic pilot-based workload management system”, *Journal of Physics: Conference Series* **119** (2008) p. 062044, DOI: doi:10.1088/1742-6596/119/6/062044.

[79] F. Berghaus et al., “High-Throughput Cloud Computing with the Cloudscheduler VM Provisioning Service”, *Computing and Software for Big Science* **4** (2020) p. 4, DOI: 10.1007/s41781-020-0036-1.

[80] G. Erli et al., “roced-scheduler/ROCED 1.1.0”, version 1.1.0 (2018), DOI: 10.5281/zenodo.1888310.

[81] Sciacca, F G and Weber, M, “Production experience and performance for ATLAS data processing on a Cray XC-50 at CSCS”, *EPJ Web Conf.* **214** (2019) p. 03023, DOI: 10.1051/epjconf/201921403023.

[82] Sven Lange, “Analyse und Vorhersage von Nutzungsverhalten und Ressourcenverbrauch in einer wissenschaftlichen Cloud-Umgebung” (2019).

[83] Max Fischer et al., *MatterMiners/cobald: v0.10.0*, version v0.10.0, 2019, DOI: 10.5281/zenodo.3469929.

[84] HTCondor Team, *HTCondor*, version 8.9.5, 2020, DOI: 10.5281/zenodo.3595387.

[85] Fernandes, João et al., “HNSciCloud, a Hybrid Cloud for Science”, *EPJ Web Conf.* **214** (2019) p. 09006, DOI: 10.1051/epjconf/201921409006.

[86] *Exoscale*, commercial cloud provider, accessed 10.10.2020, URL: https://www.exoscale.com/.

[87] Rohit Yadav et al., *Apache CloudStack*, 2018, URL: https://github.com/apache/cloudstack.

[88] Robert Barthel and Simon Raffeiner, "ForHLR: a New Tier-2 High-Performance Computing System for Research", *Proceedings of the 3rd bwHPC-Symposium* (2017) 73–75, DOI: doi:10.11588/heibooks.308.418.

[89] Hannes Hartenstein, Thomas Walter and Peter Castellaz, "Aktuelle Umsetzungskonzepte der Universitäten des Landes Baden-Württemberg für Hochleistungsrechnen und datenintensive Dienste", *PIK - Praxis der Informationsverarbeitung und Kommunikation* **36** (2013) 99–108, DOI: https://doi.org/10.1515/pik-2013-0007.

[90] Morris Jette and Mark Grondona, "SLURM: Simple Linux Utility for Resource Management" (2003).

[91] F. Fischer, "Large-scale dynamic Provisioning of Compute Resources for High Energy Physics using Cloud Technology" (2017), URL: https://publish.etp.kit.edu/record/21314.

[92] *MOAB*, URL: https://adaptivecomputing.com/.

[93] P Calafiura et al., "The ATLAS Event Service: A new approach to event processing", *Journal of Physics: Conference Series* **664** (2015) p. 062065, DOI: 10.1088/1742-6596/664/6/062065.

[94] B Bockelman et al., "Commissioning the HTCondor-CE for the Open Science Grid", *Journal of Physics: Conference Series* **664** (2015) p. 062003, DOI: 10.1088/1742-6596/664/6/062003.

[95] *NorduGrid ARC Compute Element*, accessed 07.10.2020, URL: http://www.nordugrid.org/arc/ce/.

[96] Oliver Freyermuth et al., *Operating an HPC/HTC Cluster with Fully Containerized Jobs using HTCondor, Singularity, CephFS and CVMFS*, accepted for publication in Computing and Software for Big Science, 2021.

[97] Max Fischer et al., "Effective Dynamic Integration and Utilization of Heterogenous Compute Resources", *EPJ Web of Conferences* **245** (2020) p. 07038, ed. by C. Doglioni et al., DOI: 10.1051/epjconf/202024507038.

[98] Caspart, Rene et al., "Advancing throughput of HEP analysis work-flows using caching concepts", *EPJ Web Conf.* **214** (2019) p. 04007, DOI: 10.1051/epjconf/201921404007.

[99] J. Dugan et al., *iperf*, version 2.0, URL: https://iperf.fr/.

[100] http://www.spec.org/spec/ [Online accedded: 19.03.2020], 2020.

[101] "White Paper Open Telekom Cloud" (2018), URL: https://open-telekom-cloud.com/resource/blob/data/305476/3433c472384be87e7406b8b80048f683/open-telekom-cloud-white-paper.pdf.

Acronyms

ROCED Responsive On-Demand Cloud-enabled Deployment. 55, 56

glideinWMS *glideinWMS*. 55

COBalD COBalD - the opportunistic Balancing Deamon. 58, 59, 62, 63, 65, 66, 68–70, 92, 110

TARDIS **T**ransparent **A**daptive **R**esource **D**ynamic **I**ntegration **S**ystem. 58–60, 62, 63, 65–70, 78–80, 92, 107, 110, 111

docker docker. 51, 74

drones drones. 59, 70, 110

drone drone. 51, 54–56, 58–60, 63–70, 78–80, 91, 92, 110, 113

ALICE A Large Ion Collider Experiment. 15

AMS Alpha-Magnet-Spectrometer. 61

ATLAS A Toroidal LHC ApparatuS. 15, 16, 69–71, 110

Belle II Belle II. 61, 70, 71, 92, 110

CERN Conseil européen pour la recherche nucléaire. 13, 14

CHS *charged hardon subtraction*. 25

CMS Compact Muon Solenoid. 3, 4, 13, 16–19, 21, 24–26, 30, 33, 35, 42, 44, 49, 52, 55, 61, 63, 70, 71, 75, 91, 109, 110

CSCS Swiss Centre for Scientific Computing. 56

CVMFS Cern Virtual Machine File System. 50

DGLAP Dokshitzer-Gribov-Lipatov-Altarelli-Parisi. 10

ECAL electromagnetic calorimeter. 18, 24, 25, 42

ETP Institute of Experimental Particle Physics. 44, 50, 51, 57, 61–65, 67–69, 74, 84, 92, 110, 113

ForHLR II ForHLR II. 51, 56, 63, 70, 78–80

GridKa GridKa. 52, 56, 61–63, 69–71, 74, 88, 92, 110

HCAL hardon calorimeter. 18, 25

HDD Hard Drive Disk. 62

HEP High Energy Physics. 3, 4, 13, 47, 49–56, 61, 63, 69, 73, 83, 91, 92

HL-LHC High Luminosity LHC. 13, 49, 56, 83, 91

HLT High-Level-Trigger. 19

HPC High-Performance Computing. 47, 48, 50, 51, 53, 57, 58, 63, 67, 77, 91, 92, 127

HS06 HEP-SPEC06. 83–89, 111, 113

HTC High-Throughput Computing. 47, 48, 63

JEC *jet energy correction*. 25, 26, 33, 34, 42–44

JER *jet energy resolution*. 25, 26, 43

LHC Large Hadron Collider. 3–6, 8, 12–15, 17, 25, 49, 52, 91, 92, 109

LHCb Large Hadron Collider beauty. 15

LO leading order. 6, 27, 43, 44, 91, 105, 106

NLO next-to-leading order. 6, 27, 35, 43, 44, 91, 105, 106

NNLO next-to-next-leading order. 10, 12, 21, 44

OBS overlay batch system. 52–60, 62–65, 67–70, 78, 110, 128

OTC Open Telekom Cloud. 62, 81, 85, 86

PDF parton distribution function. 3, 5, 6, 8–12, 21, 22, 44, 45, 91

PF particle-flow. 24

QCD quantum chromodynamics. 7, 10, 21

RMS root mean square. 38, 39, 43, 101, 102

SM Standard Model of particle physics. 3, 5, 7, 10, 15, 16, 109

SPEC Standard Performance Evaluation Corporation. 83

SSD Solid State Disk. 62, 73

WLCG Worldwide LHC Computing Grid. 49, 50, 52–56, 61, 69, 73, 83, 84, 88, 127

WMS workload management system. 55

Glossary

Cloudscheduler V2 Resource manager developed for opportunistic resources. 55, 56

Computing Element An entry point that accepts jobs from outside a site and submit these to the batch system. Thereby the job gets site specific modification, such as certificate to local user mapping.. 53, 69, 70, 110

`HTCondor` `HTCondor` is an open source batch system mainly developed by Center for High Throughput Computing at UW-Madison.. 51, 52, 61, 62, 74, 77, 79, 107

`Helix Nebula Science Cloud` The Helix Nebula Science Cloud was an EU-project. In this project, scientic communities and research centers can gain experiments and develop tools to use commertial cloud provdier. Furthermore, the commercial cloudprovider develop further tools and infrastrucutre to fulfill the needs of the scientific communities.. 62, 85, 92

`TOpAS` **T**hroughput **Op**timized **A**nalysis **S**ystem is a cluster at GridKa designed for end-user analysis. Worldwide LHC Computing Grid (WLCG) jobs are backfilled for high utilization.. 79, 80

`XRootD` XRootD is a file transfer protocol mainly used by HEP collaborations. It enables to stream files and transparent caching of files. 73

`bwForCluster NEMO` The bwForCluster NEMO is one of the HPC clusters of the bwHPC-C5 project project. This cluster provides resources to the neuroscience, elementary particle physics, and microsystems engineering community at Baden-Württemberg.. 50, 63, 66, 76, 84, 85

`demand` One metric used by COBalD to describe how many resources are needed.. 59, 60

`supply` One metric used by COBalD to describe how many resources are currently provided.. 59, 60

bwHPC-C5 project The bwHPC-C5 project provides state wide computing resources for scientific communities in the state of Baden-Württemberg. Therefore, several HPC cluster where build at some universities of Baden-Württemberg.. 63, 127

cloud One or more data centers which provide the infrastructure for several services.. 50

cloud site A cloud site is a set of worker nodes within an OBS which looks similar inside the OBS in term of resources, environment provion, and resource handling.. 64, 110

Event Service A service which manages monte carlo production and analyses on event level instead on file or dataset level. 69

occupancy A metric to define how well a resource is utilized.. 59, 60, 65–68, 78–80, 110

parton Partons are the constituents of a proton, namely quarks and gluons. 5, 6, 8–10, 21, 22, 39

suitability A metric to define how well a resource fits the current demand.. 59, 60, 65–68, 78, 110

Danksagung

Ohne Hilfe und Unterstützung ist eine Promotion kaum möglich. Deswegen möchte ich noch allen Danken, die mich in den vergangenen Jahren unterstützt haben.

Besonders danken möchte ich Prof. Günter Quast, dass ich bei Ihm promovieren durfte und er mir die Möglichkeit gab im Bereich Computing tätig zu werden.

Prof. Achim Streit möchte ich danken, dass er mir diese interdisziplinäre Promotion ermöglicht hat. Danke auch für sein Verständnis und die Unterstützung während der Arbeit.

Zusätzlich möchte ich der Karlsruher Schule für Elementarteilchen- und Astroteilchenphysik (KSETA) für die Förderung meiner Arbeit danken. Mein Dank gilt auch der Deutschen Forschungssegelschaft (DFG) un dem Land Baden-Württemberg für die Förderung des HPC-cluster NEMO in Freiburg über INST 39/963-1 FUGG und bwHPC. Ebenso geht mein Dank an das Betriebsteam des HPC-Clusters NEMO für den reibungslosen Betrieb und die gute Zusammenarbeit.

Der Europäischen Kommission möchte ich für die Förderung des Helix Nebula Science Cloud Projektes und den damit verbundenen Ressourcen danken.

Dem GridKa-Team und dem Steinbuch Centre for Computing (SCC) möchte ich für die offene und freundliche Aufnahme in das Team danken.

Den Arbeitskollegen am Institut für Experimentelle Teilchenphysik danke ich für die gute Zusammenarbeit und Unterstützung sowie für die angenehme Arbeitsatmosphäre. Besonderen Dank gilt dem Admin Team für die Betreuung und Verwaltung der IT-Infrastruktur. Der QCD-Gruppe danke ich für die Unterstützung, Diskussion und Hilfestellung bei dem Physikteil meiner Arbeit. Dem Computing Team, vor allem Christoph Heidecker, R. Florian von Cube und Maximilian Horzela danke ich für die Unterstützung und die angeregten Diskussionen sowie die angenehme Arbeitsatmosphäre.

Danken möchte ich auch meinen Freunden, Bekannten und meiner Familie für die Unterstützung und das Verständnis während der Promotion.

Mein besonderer Dank geht vor allem an Manuel Giffels, Max Fischer, Eileen Kühn und René Caspart für die großartige Unterstützung während der vergangenen Jahre.

Erklärung der selbständigen Anfertigung meiner Dissertationsschrift

Hiermit erkläre ich, dass ich die Dissertationsschrift mit dem Titel

Dynamic Provision of Heterogeneous Computing Resources for Computation- and Data-intensive Particle Physics Analyses

selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht habe, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

Ich versichere außerdem, dass ich die Dissertation nur in diesem und keinem anderen Promotionsverfahren eingereicht habe und dass diesem Promotionsverfahren keine endgültig gescheiterten Promotionsverfahren vorausgegangen sind.

Karlsruhe, den 31. 03. 2021
Matthias J. Schnepf