Karlsruhe Institute of Technology

ETP-KA/2018-26

Automated Production of Interpolation Grids at NNLO for the Triple-Differential Z+Jet Cross Section Measurement at the LHC

MASTER'S THESIS

of

Miguel Santos Correa

at the Institute of Experimental Particle Physics (ETP)
of the Karlsruhe Institute of Technology (KIT)

Reviewer: Prof. Dr. Günter Quast
Second Reviewer: Priv.-Doz. Dr. Klaus Rabbertz

13th November 2018

Abstract

In this thesis, theory calculations for the triple-differential Z+jet cross section are presented that are suitable for PDF fits with CMS data from the LHC at $\sqrt{s} = 13\,\mathrm{TeV}$. These calculations are performed at NNLO and provided in the form of interpolation grids. An implementation of an automated pipeline for the creation of interpolation grids is offered and was used for this analysis. The cross section is presented as a function of the transverse Z boson momentum $p_{\mathrm{T}}^{\mathrm{Z}}$, two rapidity variables y_{b} and y^* of the Z+jet system, and an angular variable ϕ_η^* of the dimuon system. An analysis is performed on the resulting interpolation quality and the contribution of different partonic channels to the cross section.

Contents

CHAPTER 1

Introduction

Ever since its inception in the 19th century, particle physics has played an increasingly important role in our understanding of the laws of nature. This has led to the construction of big particle accelerators to probe and expand our knowledge of the smallest constituents of the universe. The Large Hadron Collider (LHC) is currently the largest particle accelerator worldwide that accelerates and collides protons at unprecedented energies. The LHC has four collision points, equipped with large particle detectors, one of which is the Compact Muon Solenoid (CMS) experiment. These detectors are designed to track and identify as many outgoing particles as possible. This information is then used in extensive analyses to reconstruct the events.

The best understood framework to describe the motion of particles and the outcome of particle collisions is Quantum Field Theory (QFT). Most predictions in QFT are done through the use of perturbation theory, however, some quantities cannot be calculated perturbatively. For instance, perturbative QFT is unable to provide a description of the proton structure, which is instead described by so called Parton Distribution Functions (PDFs). It is of crucial importance to determine the proton structure very accurately, since it affects most predictions for measurements at the LHC.

The analysis in this thesis focuses on the production of a Z boson in association with a hadronic jet to further constrain the uncertainties of the proton PDFs. Theory predictions for Z+jet events are stored in so-called interpolation grids to make them suitable for PDF fits. These grids will then be used in conjunction with the 2017 data from the CMS detector in a future analysis.

The production of interpolation grids at Next-to-Next-to-Leading-Order (NNLO) is a very resource heavy endeavor, costing hundreds of thousands of CPU hours on a computing cluster. A software pipeline was written as part of this thesis to automate the entire workflow of producing interpolation grids at NNLO. The pipeline handles the creation of interpolation grids and the creation of plots for cross checks of the interpolation quality. It is accompanied by an intuitive web user interface to present these automatically generated plots. The pipeline is used to create Z+jet interpolation grids as an example and will be used for other processes of interest in the future.

Outline Chapter 2 of this thesis outlines the theoretical foundations required for the study of Z+jet events at the LHC. Chapter 3 discusses the experimental setup of this analysis and its objectives. Chapter 4 gives a brief overview of the interpolation technique for cross section predictions. Chapter 5 describes in detail the implementation of a pipeline for the production of interpolation grids. The actual production of Z+jet interpolation grids is documented and analyzed in Chapter 6.

CHAPTER 2

Theoretical Framework

Notations and Conventions This thesis uses the natural unit notation in which the reduced Plank's constant and the speed of light are set to unity

$$\hbar = c = 1$$

According to the Einstein summation convention repeated indices in single terms that are not defined otherwise are summed over all possible values, e. g.,

$$a^i b^i = a^0 b^0 + a^1 b^1 + a^2 b^2$$

Greek indices run over 0, 1, 2, 3 and denote the four spacetime dimensions with

$$x_\mu = g_{\mu\nu} x^\nu$$

where $g_{\mu\nu}$ is the Minkowski metric tensor.

2.1 The Standard Model of Particle Physics

The physical laws of the universe are governed by four fundamental forces known as the electromagnetic, weak, strong, and gravitational force. The Standard Model of particle physics (SM) [1, 2] describes three of the four fundamental forces of nature, excluding gravity, and all known elementary particles. The SM is a quantum field theory, meaning that the Lagrangian consists of operators which are defined at all points in spacetime. Elementary particles are described as excitations of their respective particle field operators.

Figure 2.1 shows a list of all particles in the SM. They can be categorized into *bosons* with integer spin and *fermions* with half-integer spin. Fermions are the fundamental constituents of matter while bosons are responsible for mediating forces between fermions. There are two groups of fermions, the *quarks* who interact with the strong force and the *leptons* who do not interact strongly.

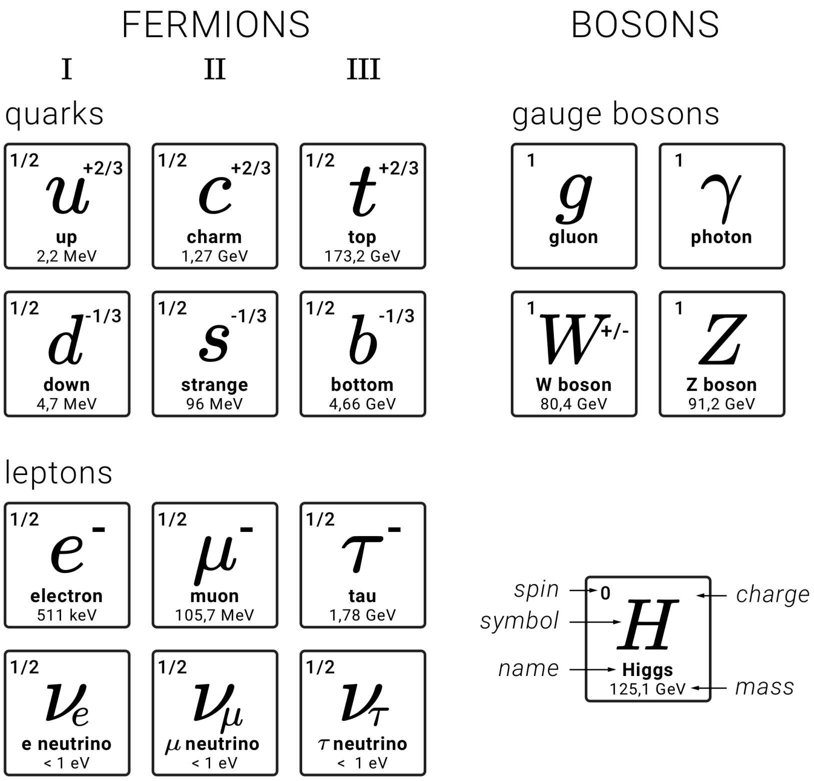


Figure 2.1: Illustration of all elementary particles of the SM with their respective properties. Particle mass data taken from [3]. Exact neutrino masses are unknown, but the sum of all three neutrino masses is constrained by $\sum m_\nu < 120\,\mathrm{meV}$ (95% C.L.) [4].

Gauge theory The SM Lagrangian is invariant under certain Lie groups of local transformations on fields, which can be written as

$$\psi(x) \rightarrow e^{i\alpha^a(x)t^a}\psi(x) \tag{2.1}$$

where ψ denotes a field operator, $\alpha^a(x)$ a set of real valued functions defined on all spacetime points x and t^a the generators of the Lie group. The symmetry group of the SM is the product of $\mathrm{U}(1) \times \mathrm{SU}(2)_\mathrm{L} \times \mathrm{SU}(3)$, where U(1) refers to the group of one-dimensional unitary matrices and SU(N) to the group of N-dimensional special unitary matrices. The SU(N) is defined by the commutation relations

$$\left[t^a, t^b\right] = if^{abc}t^c \tag{2.2}$$

with structure constants f^{abc}. This is also known as a non-abelian gauge theory, because of its non-commuting generators. Every symmetry of the Lagrangian is associated with a fundamental force and gives rise to corresponding gauge boson fields, which act as force carriers.

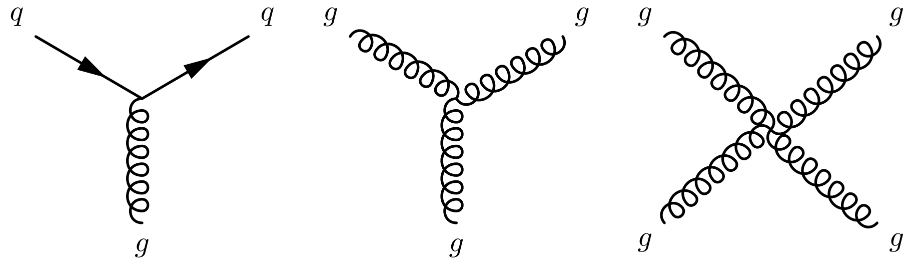


Figure 2.2: Fundamental interaction vertices of QCD. From left to right: quark-gluon coupling, trilinear gauge coupling, quartic gauge coupling.

2.2 Quantum Chromodynamics

Quantum Chromodynamics (QCD) is the theory of the strong interaction and is based on the SU(3) symmetry group. Starting from a Lagrangian of noninteracting free quark fields and enforcing gauge invariance one can derive the famous Yang-Mills Lagrangian of QCD

$$\mathcal{L}_{\text{QCD}} = \sum_f \bar{\psi}_f (i\gamma^\mu D_\mu - m_f)\psi_f - \frac{1}{4} G^a_{\mu\nu} G^{a\mu\nu} \tag{2.3}$$

where the sum goes over all quark flavors f. D_μ denotes the gauge covariant derivative and $G^a_{\mu\nu}$ the gluon field strength tensor with $a = 1, ..., 8$. These are defined as

$$D_\mu = \partial_\mu - igA^a_\mu t^a \tag{2.4}$$

$$G^a_{\mu\nu} = \partial_\mu A^a_\nu - \partial_\nu A^a_\mu + g f^{abc} A^b_\mu A^c_\nu \tag{2.5}$$

with g being a dimensionless coupling constant and A^a_μ the field operators for the eight gauge bosons of the SU(3), namely the gluons. The covariant derivative leads to an interaction between quarks and gluons, while the quadratic field strength tensor term in the Lagrangian leads to gluon-gluon interaction terms. A pictorial representation of these interactions in form of so-called *Feynman diagrams* is given by Figure 2.2.

Running coupling Applying perturbation theory to quantum field theories will often lead to infinite integrals. There is a technique called *renormalization* that can be used to control these infinities. In essence, the bare parameters of the Lagrangian are replaced with so-called *physical parameters* which are experimentally determined. A negative side effect of the renormalization of QCD is the introduction of a non-physical parameter, the *renormalization scale*.

The evolution of the strong coupling α_s as a function of the renormalization scale μ_r^2 is given at Leading-Order (LO) by

$$\alpha_\text{s}\left(\mu_r^2\right) = \frac{12\pi}{(33 - 2n_\text{f}) \ln(\mu_r^2/\Lambda^2)} \tag{2.6}$$

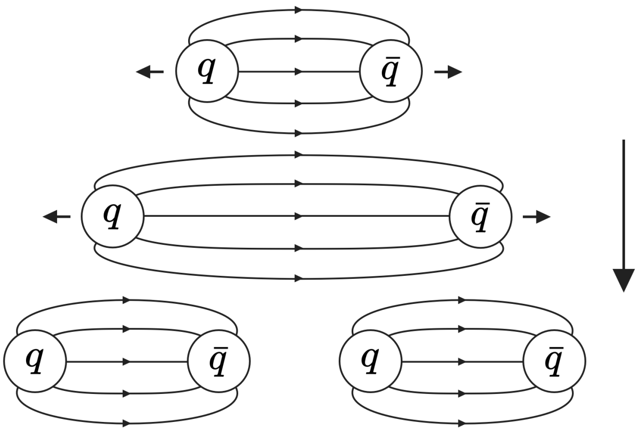


Figure 2.3: Gauge field configuration associated with the separation of color charges. It becomes energetically favorable to create a quark-antiquark pair at some point as the distance increases.

where n_f is the number of quark flavors and Λ^2 the momentum scale. Any observable that depends on α_s will also depend on the renormalization scale. However, since the scale is merely an artifact of the renormalization prescription, one has to conclude that observables cannot possibly depend on the scale. Any such scale dependence comes from the perturbative series only being evaluated for a fixed order. An appropriate choice for the scale has to be made in order to get good predictions.

Asymptotic freedom A natural scale choice for particle collisions is to set the scale equal to the momentum transfer Q^2. As the running coupling in equation 2.6 is a monotonically decreasing function, the interactions between quarks become asymptotically weaker as the momentum transfer increases. This effect is known as *asymptotic freedom*. It allows for accurate perturbative calculations at high energies and short distances.

Color confinement All bound states of QCD have to be color neutral, hence single quarks or gluons cannot be observed in nature as free particles. This phenomenon is known as *color confinement*. This property comes as a direct consequence of non-abelian gauge theories having charged gauge bosons. If one attempts to separate color neutral composites into colored components, the gluon field between the charges will form a tube of constant energy density. When the distance between the charges becomes too large, and with it the energy of the gluon field, it becomes energetically favorable to split the tube by creating a quark-antiquark pair. Figure 2.3 illustrates the separation of color charges in QCD. This phenomenon is observed in particle accelerators when individual quarks or gluons are produced in particle collisions. Due to color confinement, quark-antiquark pairs get created and form a collimated stream of color neutral hadronic particles, called a jet.

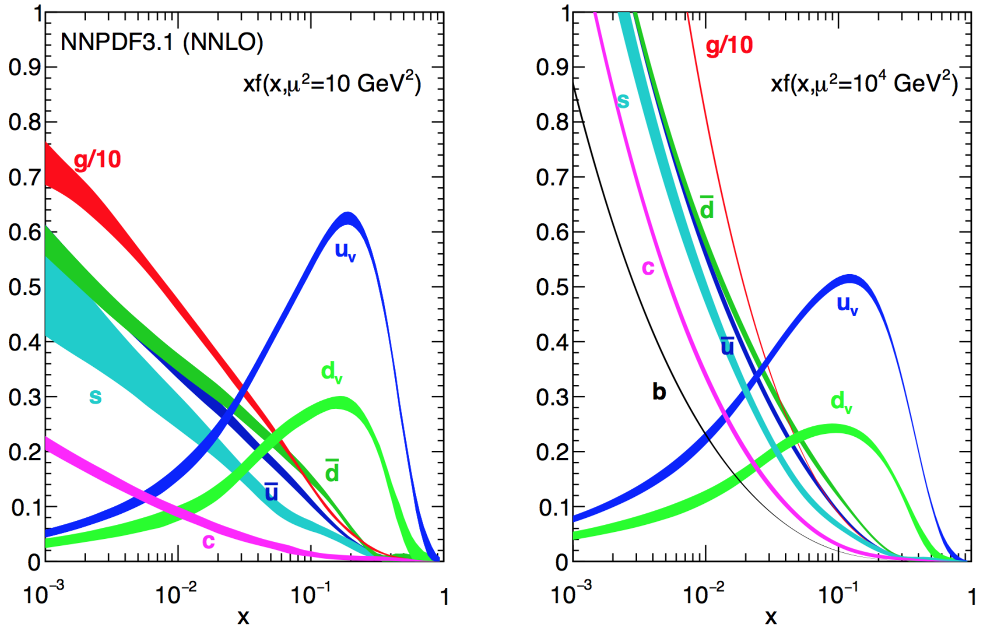


Figure 2.4: The NNPDF3.1 NNLO PDFs, evaluated at $\mu_f^2 = 10\,\text{GeV}^2$ (left) and $\mu_f^2 = 10^4\,\text{GeV}^2$ (right) from [5].

2.3 Parton Distribution Functions

The structure of hadrons, such as protons and neutrons, cannot be calculated perturbatively as they reside within the strongly coupled low-energy regime. It is, however, of essential importance to know the structure of hadrons in order to make accurate predictions for high-energy collisions between hadrons. The *parton model* for hadrons was introduced as a substitute for missing QCD predictions. In this model, hadrons are assumed to consist of a loosely bound assemblage of constituents, called *partons*. Hadrons consist of so-called *valence quarks* which give rise to its quantum numbers and a large number of gluons and quark-antiquark pairs formed by so-called *sea quarks*.

The Parton Distribution Functions (PDFs) $f_i(x, \mu_f^2)$ describe the probability of a parton i to carry a momentum fraction x of the total proton momentum. The *factorization scale* μ_f^2 is another non-physical parameter that impacts the perturbative expansion. It defines the dividing line between the process description through PDFs ($Q^2 < \mu_f^2$) and through a perturbative expansion ($Q^2 > \mu_f^2$). PDF sets are produced by various groups, e. g., NNPDF [5], HERA [6], CTEQ [7], ABM [8], MMHT [9]. Figure 2.4 shows the PDFs for two different choices of μ_f.

Factorization Theorem The cross section of proton-proton collisions can be described in terms of the simpler partonic cross section by using the *factorization theorem* [10, 11]. The hard process happens between two partons originating from the protons with a momentum fraction given by their respective PDFs. Integrating and summing over all

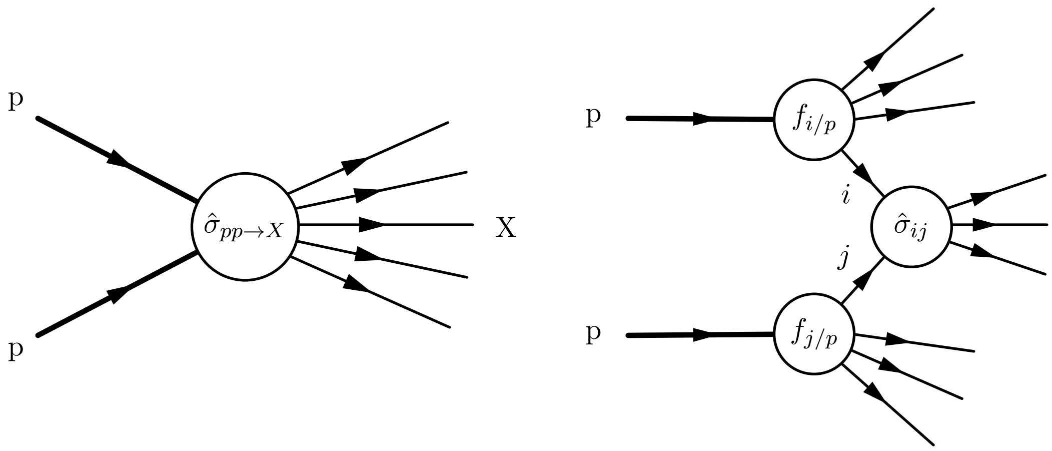


Figure 2.5: Proton-proton scattering process (left) factorized into hard parton-level cross section with PDFs (right).

possible partons and momenta one can derive the proton-proton cross section

$$d\hat{\sigma}_{pp\to X} = \sum_{i,j} \int dx_1 dx_2 f_{i/p}(x_1,\mu_f) f_{j/p}(x_2,\mu_f) \times d\hat{\sigma}_{ij\to X}(x_1,x_2,\mu_r,\mu_f,\alpha_s(\mu_r)) \quad (2.7)$$

This formula allows for quantitative predictions since the parton-level cross section can be calculated in perturbative Quantum Chromodynamics (pQCD) and the PDFs are known from experimental measurements. Figure 2.5 shows a pictorial illustration of the factorization theorem.

DGLAP Evolution Equations The dependence of PDFs $f_i\left(x,\mu_f^2\right)$ on the factorization scale μ_f^2 is described by the DGLAP [12–14] evolution equations

$$\frac{\partial f_i\left(x,\mu_f^2\right)}{\partial \ln\left(\mu_f^2\right)} = \frac{\alpha_\mathrm{s}(\mu_r^2)}{2\pi} \sum_j \int_x^1 \frac{d\xi}{\xi} P_{ij}(x/\xi) f_j\left(\xi,\mu_f^2\right) \quad (2.8)$$

with i, j going over all partons. The so-called *splitting functions* $P_{ij}(x/\xi)$ describe the probability of a parton j with momentum ξ to emit a parton i with momentum x. The evolution equations allow for one PDF set to be used for predictions with a dynamic scale. The splitting functions do not have to be measured, they can be calculated in pQCD.

CHAPTER 3

Z Boson Production at the Large Hadron Collider

3.1 The Large Hadron Collider

With a circumference of 27 km and a center of mass energy of up to 14 TeV the Large Hadron Collider (LHC) [15] is currently the most powerful particle accelerator worldwide. The LHC is located at the France-Switzerland border and was built by the European Organization for Nuclear Research (CERN) in a collaboration of thousands of scientists from over 100 countries. The construction started in 1998 and finished in 2008. The LHC started its first operation in 2009 with proton collisions with a center of mass energy of 7 TeV which was increased to 8 TeV in 2012. The LHC shut down in 2013 for maintenance and upgrades and was relaunched two years later in 2015 with a center of mass energy of 13 TeV.

Figure 3.1 depicts an illustration of the CERN complex with the LHC. The LHC is designed to collide beams of protons or heavy ions, which are pre-accelerated by a set of particle accelerators prior to entering the LHC. There are four interaction points with particle detectors in which the particle beams collide: ALICE [16], ATLAS [17], CMS [18] and LHCb [19].

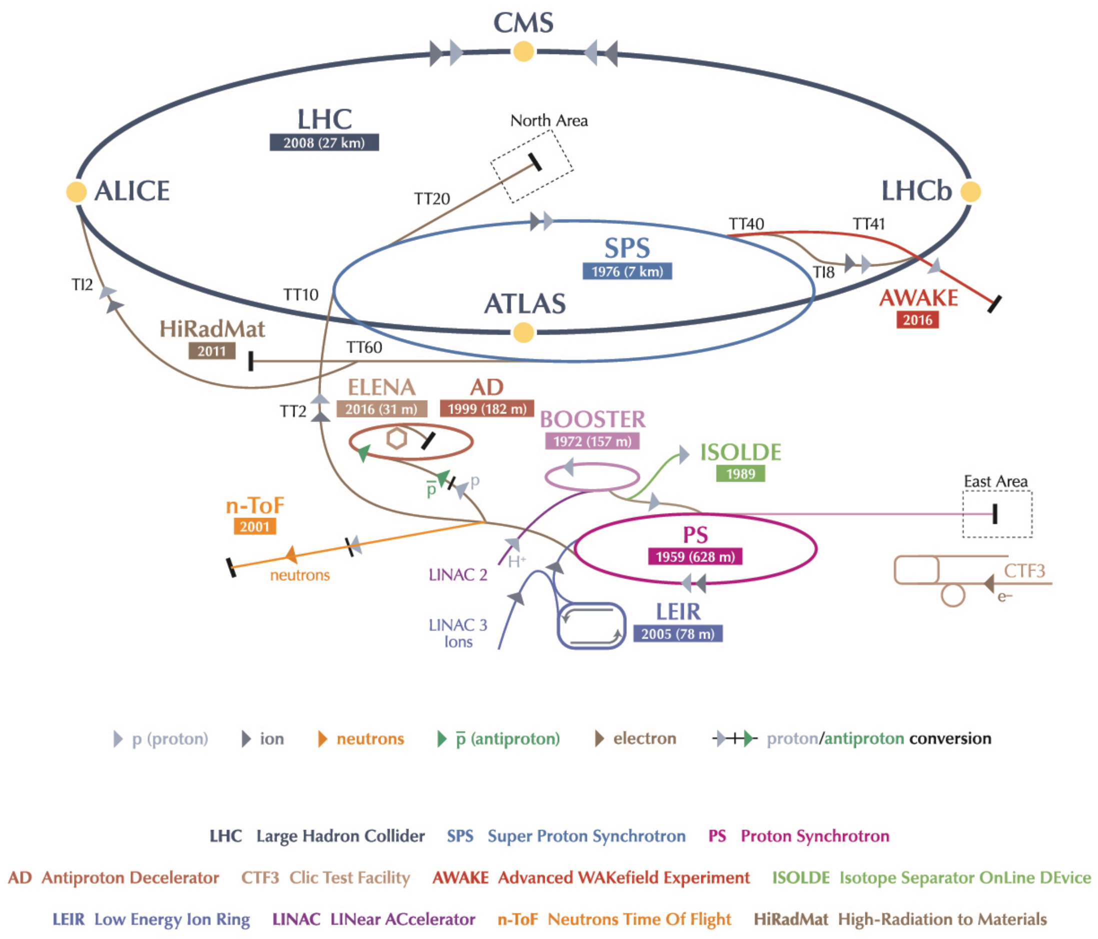


Figure 3.1: Illustration of the CERN accelerator complex. The LHC is the last ring (dark blue line) in a complex chain of particle accelerators. [20].

3.2 The Compact Muon Solenoid Experiment

Coordinate Conventions

The Compact Muon Solenoid (CMS) detector has a fixed and well defined coordinate system which is used throughout this thesis. The origin of the coordinate system is in the main collision point at the center of the detector. The cartesian axes are defined as

x	horizontal and pointing towards the center of the LHC
y	vertical and pointing upwards
z	horizontal and pointing along the beam axis westward

A visual representation of the coordinate system is illustrated in Figure 3.2.

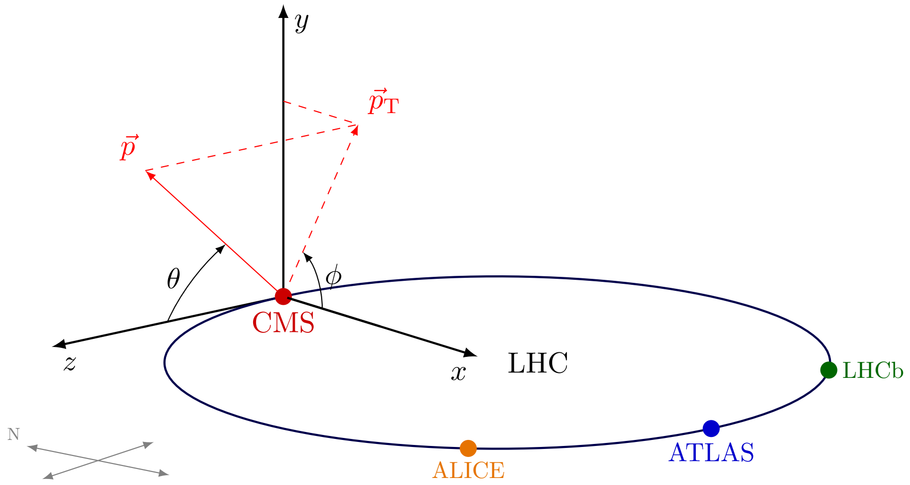


Figure 3.2: Coordinate system of CMS. Derivative of [21].

The transverse momentum p_{T} of a particle is defined as

$$p_{\mathrm{T}} = \sqrt{p_x^2 + p_y^2} \tag{3.1}$$

Rapidity and Pseudorapidity

In high energy physics it is common practice to use rapidity and pseudorapidity instead of a particle's polar angle. Differences in rapidities are invariant under Lorentz boosts along the beam axis z and offer therefore a better basis for studying collisions.

The rapidity y of a particle is defined as

$$y = \frac{1}{2} \ln\left(\frac{E + p_z}{E - p_z}\right) \tag{3.2}$$

where E is the total energy of the particle and p_z its momentum along the beam axis.

The pseudorapidity η is a function of the polar angle θ between the particle and the beam axis

$$\eta = -\ln\left[\tan\left(\frac{\theta}{2}\right)\right] \tag{3.3}$$

On first sight the rapidity and pseudorapidity of a particle seem quite different, but they are indeed closely related. The pseudorapidity can also be written down as

$$\eta = \frac{1}{2} \ln\left(\frac{|\vec{p}| + p_z}{|\vec{p}| - p_z}\right) \tag{3.4}$$

In the vicinity of a particle approaching the speed of light, or its mass being negligible ($E \approx |\vec{p}|$), the pseudorapidity will converge to the same value as the rapidity.

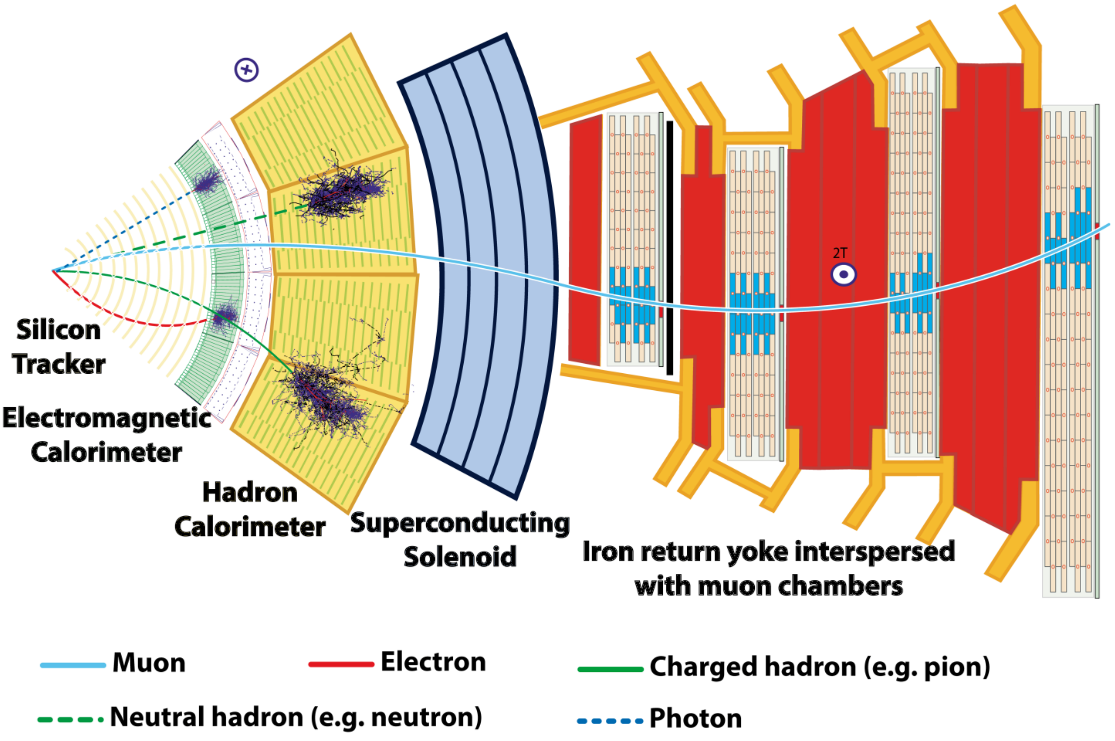

Silicon Tracker
Electromagnetic Calorimeter
Hadron Calorimeter
Superconducting Solenoid
Iron return yoke interspersed with muon chambers
2T
Muon
Electron
Charged hadron (e.g. pion)
Neutral hadron (e.g. neutron)
Photon

Figure 3.3: Slice showing CMS sub-detectors and how particles interact with them. Derivative of [22].

Detector

The CMS detector [18] is a general-purpose detector that sits at one of four collision points at the LHC. It has a cylindrical structure that is 15 meters high and 21 meters long. The collision point in the center of the detector is surrounded by layers of sub-detectors. Fig. 3.3 shows a slice of the CMS detector and how various particles interact with its components. The detector consists of the following components, starting from the center towards the outside:

Silicon Tracker The silicon tracker is composed of three layers of silicon pixel detectors, surrounded by ten layers of silicon strip detectors. The tracker has a full azimutal coverage within $|\eta| < 2.5$. The silicon pixel detector has 1440 modules that contain a total of 66 million pixels in an area of about $1\,\mathrm{m}^2$. It is surrounded by 15 148 strip detector modules with a total of 9.3 million strips and an active area of $198\,\mathrm{m}^2$. The tracker can reconstruct the paths of high-energy charged particles through the magnetic field, which allows for a precise measurement of particle momenta.

Electromagnetic Calorimeter The electromagnetic calorimeter consists of 61 200 lead tungstate ($PbWO_4$) crystals in the central barrel region with an additional 7324 crystals in each of the two endcaps. This allows for the measurement of the energy of charged particles with an excellent energy resolution. The barrel region covers the pseudorapidity range $|\eta| < 1.479$ and is complemented by the endcap region coverage in $1.479 < |\eta| < 3.0$.

Hadron Calorimeter The brass-scintillator sampling hadron calorimeter with coverage up to $|\eta| < 3.0$ is responsible for the energy measurement of hadrons. It is complemented by a different kind of hadron calorimeter in the forward region $3.0 < |\eta| < 5.2$ using a Cherenkov-based, radiation-hard technology.

Superconducting Solenoid The NbTi superconductor is operated at a temperature of 4 K and can generate a magnetic field strength of up to 4 T. The superconductor encloses the calorimeters and silicon tracker and bends the tracks of charged particles inside. The strong magnetic field allows for a high momentum resolution of charged particles in the tracking system.

Muon System The muon system is located outside the solenoid and consists of about $25\,000\,\mathrm{m}^2$ of detection planes that cover the pseudorapidity range $|\eta| < 2.4$. This system uses drift tubes and cathode strip chambers to track the spatial position of particles and resistive plate chambers as an additional trigger with fast readout times. This allows for the identification and measurement of muons with high precision.

3.3 Production of a Z Boson in Association with a Hadronic Jet

The Z boson is the neutral vector boson of the weak interaction. It is a heavy spin 1 particle with a mass of 91.187 GeV. The Z boson is very short-lived due to its heavy mass and has a half-life of 3×10^{-25} s. The detection of Z bosons is only done indirectly through its decay channels. Z bosons decay into all fermion anti-fermion pairs of the SM, except for the higher-mass top quarks. The branching ratios of the individual decay modes of the Z boson are shown in Table 3.1. The hadronic channel has the largest contribution to the Z boson decay. The summation over all color charges in the Feynman diagrams explains the higher branching ratio compared to leptons. The analysis in this thesis utilizes the CMS detector's ability to measure muons precisely and therefore focuses on the muon channel with a branching ratio of approximately 3.367 %.

Figure 3.4 shows the LO Feynman diagrams of the Z+jet channel. A virtual Z boson is created at the double quark vertex and decays subsequently into a muon anti-muon pair. The main channels for Z boson production are quark-gluon and quark-antiquark of the same flavor. The high quark-gluon cross section makes the Z boson a prime target for PDF studies, as the lesser known gluon PDF has a large impact on the cross section.

Table 3.1: Z boson decay modes and branching ratios [3]

Decay mode	Branching ratio / %
Hadrons	69.91(6)
Down type	15.6(4)
Up type	11.6(6)
Charged leptons	10.097(3)
Electron	3.363(4)
Muon	3.366(7)
Tau	3.367(8)
Neutrinos	20.00(6)

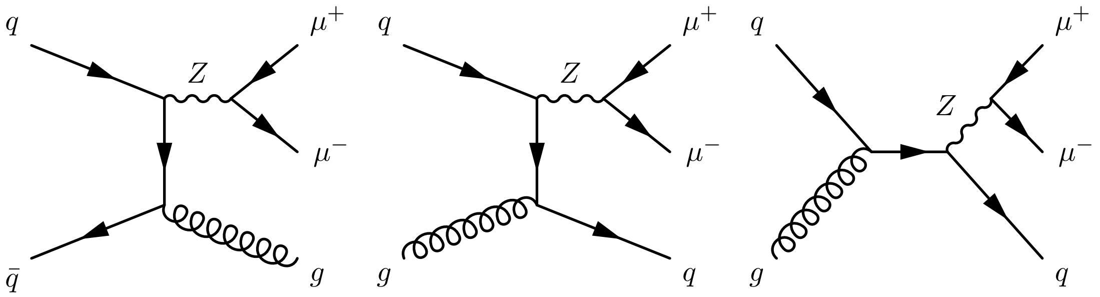


Figure 3.4: Feynman diagrams of the Z+jet channel in LO with the largest contribution to the cross section. An additional three diagrams in LO exist, which can be constructed by reversing the direction of the quark propagator.

3.4 Objectives of the Z+Jet Analysis

The objective of this analysis is to further constrain the PDF uncertainties, especially those of the gluon PDF. This thesis performs the necessary theory calculations for such an analysis and prepares them to be suitable for PDF fits. The evaluation of CMS data and the subsequent fitting of these data with the provided theory calculations is outside the scope of this thesis. This section describes the binned observables and selection cuts used in the analysis.

3.4.1 Triple-Differential Cross Section

The Z+jet analysis consists of two triple-differential studies of the cross section

$$\frac{d^3\sigma}{dy_\mathrm{b}dy^*dp_\mathrm{T}^\mathrm{Z}} \quad \text{and} \quad \frac{d^3\sigma}{dy_\mathrm{b}dy^*d\phi^*_\eta} \tag{3.5}$$

In both cases, the cross section σ is differentiated with respect to y_b and y^*, which are rapidity variables of the Z+jet system. The third differentiation is with respect to p_T^Z or ϕ^*_η, which only depend on the Z boson.

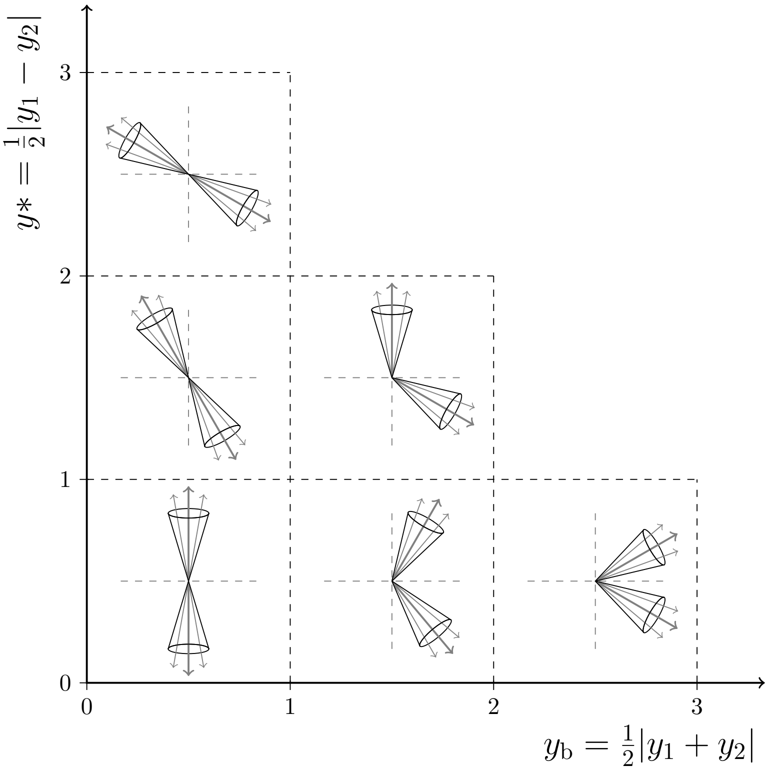


Figure 3.5: Illustration of event topologies of the Z boson and leading jet in the y_b and y^* kinematic plane. The original image [23] is from a dijet analysis, but the topology is the same with one jet replaced by a Z boson.

Rapidity Regions

The dependence of the cross section on the rapidity y^Z of the Z boson and the rapidity y^{jet} of the leading jet is expressed in terms of two different observables, y_b and y^*.

$$y_b = \frac{1}{2}\left|y^Z + y^{\text{jet}}\right| \tag{3.6}$$

$$y^* = \frac{1}{2}\left|y^Z - y^{\text{jet}}\right| \tag{3.7}$$

y_b is a measure of the boost of the Z+jet system and y^* a measure of the polar scattering angle in the center-of-mass system. Figure 3.5 illustrates the relationship of y_b, y^* and the geometry of the Z+jet system.

The ϕ^*_η Observable

The ϕ^*_η variable [24] is purely dependent on angular parameters. ϕ^*_η is described with the help of another variable θ^*_η which is defined as

$$\cos\left(\theta^*_\eta\right) = \tanh\left(\frac{\eta^- - \eta^+}{2}\right) \tag{3.8}$$

where η^- and η^+ refer to the pseudorapidities of the negatively and positively charged lepton of the decayed Z boson. With the addition of $\Delta\phi$, the azimuthal angle between the two leptons, one can write down ϕ^*_η as

$$\phi^*_\eta = \tan\left(\frac{\pi - \Delta\phi}{2}\right) \sin\left(\theta^*_\eta\right) \tag{3.9}$$

The reason for using ϕ^*_η over the more conventional p_T^Z is that ϕ^*_η can be measured more accurately, due to it being a purely angular quantity.

3.4.2 Event Selection

The selection of events is central to obtaining a meaningful dataset that is suited for cross section studies. The following selection criteria were chosen:

Jets Only Z+Jet inclusive events are selected and analyzed, meaning that the number of jets is restricted to be a minimum of 1.

$$N^\mathrm{jets} \geq 1 \tag{3.10}$$

Jet reconstruction becomes increasingly difficult with higher rapidities in the endcap regions due to tracker limitations. To mitigate this effect a cut on the pseudorapidity of the leading jet is used:

$$\left|\eta^\mathrm{jet}\right| < 2.4 \tag{3.11}$$

Low energy jets are difficult to calibrate and suffer from large systematic biases. Therefore, a lower limit on the transverse momentum of the leading jet is imposed:

$$p_\mathrm{T}^\mathrm{jet} > 20\,\mathrm{GeV} \tag{3.12}$$

Muons To achieve a high precision muon reconstruction, only muons in the pseudorapidity region

$$|\eta^\mu| < 2.4 \tag{3.13}$$

are selected. Additionally, a lower threshold for the muon transverse momenta is used:

$$p_\mathrm{T}^\mu > 25\,\mathrm{GeV} \tag{3.14}$$

Table 3.2: Triple-differential binning of y^*, y_b and $p_\mathrm{T}^\mathrm{Z}/\phi_\eta^*$

Observable	Bin edges
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
Standard binning (S)	
p_T^Z/GeV	25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 110, 130, 150, 170, 190, 220, 250, 400, 1000
ϕ_η^*	0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.2, 1.5, 2, 3, 4, 5, 7, 10, 15, 20, 30, 50
Edge binning (E)	
p_T^Z/GeV	25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 110, 130, 150, 170, 190, 250, 1000
ϕ_η^*	0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.2, 1.5, 2, 3, 5, 10, 50
Extreme binning (X)	
p_T^Z/GeV	25, 30, 40, 50, 70, 90, 110, 150, 250
ϕ_η^*	0.4, 0.6, 0.8, 1.0, 5

Z boson The dimuon invariant mass range is set to 20 GeV around the official Particle Data Group (PDG) Z boson mass to increase the probability that the muon anti-muon pair originated from a decayed Z boson.

$$\left|M^\mathrm{Z} - M^\mathrm{Z}_\mathrm{PDG}\right| < 20\,\mathrm{GeV} \tag{3.15}$$

Binning

Table 3.2 shows the standard binning used in this analysis. There are 5 bins of equal size for both y_b and y^* in the range of 0 to 2.5. This amounts to only 15 bins in total instead of 25, due to the chosen cuts in the phase space. The sum of the upper bin edges of y_b and y^* is restricted to not exceed the value 3.0, which results in only 15 bins. The standard binning for p_T^Z consists of 19 bins of various sizes in the range of 25 GeV to 1000 GeV. A different p_T^Z binning was chosen for rapidity bins that are less populated by events. The same is true for the ϕ_η^*, which has a standard binning of 18 bins in the range of 0.4 to 50. Table 3.3 shows a map of the binning from Table 3.2.

Table 3.3: Map of the triple-differential binning, as defined in 3.2.

$y^{*\mathrm{max}}$						
2.5	X					
2.0	E	E				
1.5	S	S	E			
1.0	S	S	S	E		
0.5	S	S	S	S	E	
0.0	0.5	1.0	1.5	2.0	2.5	$y_\mathrm{b}^\mathrm{max}$

CHAPTER 4

Theory Predictions for Z+Jet Events

4.1 Next-to-Next-to-Leading-Order Calculations

Next-to-Leading-Order (NLO) QCD predictions for Z+jet production have existed for a long time now, and are e. g. implemented in the Monte-Carlo event generator SHERPA [25] in combination with BLACKHAT [26] or OPENLOOPS [27] for loop diagrams. The predecessor studies [28–30] to this thesis have used NLO predictions from SHERPA for PDF fits of CMS data at $\sqrt{s} = 13\,\text{TeV}$. In recent developments, the Next-to-Next-to-Leading-Order (NNLO) corrections to the Z+jet production became available and were implemented in the parton-level Monte Carlo generator NNLOJET [31–33]. The NNLO calculation in [31] yielded a 1 % increase to the NLO cross section and significantly reduced scale variation uncertainty. Similar results are to be expected for the phase space defined in this thesis.

4.2 Interpolation Technique for Cross Sections

The calculation of cross sections in hadron collisions are increasingly demanding in time consumption, taking up to several years of computing time for high precision results. However, the fitting procedure used in PDF fits requires repeated calculations of the cross section for varying PDFs. An efficient technique for repeated calculations with varying PDFs is therefore required. This technique is implemented by two projects, FASTNLO [34–36] and APPLGRID [37, 38].

The cross section is a convolution integral over α_s, the PDFs and the hard scattering process, which is time-consuming to calculate.

$$\sigma_{pp\to X} = \sum_{abn} \int dx_1 dx_2 \alpha_s^n(\mu_r) f_{a/p}(x_1, \mu_f) f_{b/p}(x_2, \mu_f) \sigma^n_{ab\to X}(x_1, x_2, \mu_r, \mu_f) \tag{4.1}$$

The use of interpolation techniques enables the separation of the α_s and PDF dependence from the hard scattering process. The PDFs are approximated as a sum of eigenfunctions E_i on a grid in x.

$$f_{a/p}(x) \approx \sum_i f_{a/p}(x_i) E_i(x) \tag{4.2}$$

The set of eigenfunctions constitutes an orthonormal and complete basis, fulfilling the relations

$$E_i(x_k) = \delta_{ik} \tag{4.3}$$

$$\sum_i E_i(x) = 1 \tag{4.4}$$

The same procedure is used to extract the scale dependence of the hard scattering process.

$$\sigma^n_{ab\to X}(\mu) \approx \sum_k \sigma^n_{ab\to X}(\mu_k) E_k(\mu) \tag{4.5}$$

The time consuming part of the cross section is calculated once, using the formula

$$\widetilde{\sigma}_{n,i,j,k,a,b} = \int dx dx' E_i(x) E_j(x') E_k(\mu) \sigma^n_{ab\to X}(x, x', \mu) \tag{4.6}$$

and stored in so-called *interpolation grids*. Finally, the cross section can be expressed by a sum

$$\sigma_{pp\to X} = \sum \alpha^n_s(\mu_k) f_{a/p}(x_i, \mu_k) f_{b/p}(x_j, \mu_k) \widetilde{\sigma}_{n,i,j,k,a,b} \tag{4.7}$$

over α_s, the PDFs and the interpolation grid coefficients. These interpolation grid coefficients will only have to be recalculated if either the hard scattering process or the observable definition or binning changes.

4.2.1 Scale Flexibility

Scale flexibility [39] allows for a change in scale without redoing the time consuming calculation. The perturbative expansion of the cross section at NNLO is given by

$$\sigma = \underbrace{\sigma_0}_{\text{LO}} + \underbrace{\log\left(\mu_r^2\right)\sigma_r + \log\left(\mu_f^2\right)\sigma_f}_{\text{NLO}} + \underbrace{\log^2\left(\mu_r^2\right)\sigma_{rr} + \log^2\left(\mu_f^2\right)\sigma_{ff} + \log\left(\mu_r^2\right)\log\left(\mu_f^2\right)\sigma_{rf}}_{\text{NNLO}} \tag{4.8}$$

The scale is expressed as a generic function of m variables

$$\mu = \mu(O_1, ..., O_m) \tag{4.9}$$

Instead of performing the interpolation on the scale directly, the interpolation is now performed on the m variables. After calculating the interpolation grid coefficients, the cross section can be calculated for any given function $\mu(O_1, ..., O_m)$. A grid is introduced for every Observable $O_i \to O_{i,o_i}$, with

$$\mu_{o_1,...,o_m} = \mu(O_{1,o_1}, ..., O_{m,o_m}) \tag{4.10}$$

This leads to the introduction of a set of m eigenfunctions:

$$\sigma^n_{ab\to X}(\mu) \approx \sum_{o_1,...,o_m} E_{o_1}(O_1) \cdot ... \cdot E_{o_m}(O_m) \cdot \sigma^n_{ab\to X}(\mu_{o_1,...,o_m}) \tag{4.11}$$

The approximation of the cross section is then given by

$$\sigma_{pp\to X} = \sum \alpha^n_s(\mu_{o_1,...,o_m}) f_{a/p}(x_i, \mu_{o_1,...,o_m}) f_{b/p}(x_j, \mu_{o_1,...,o_m}) \widetilde{\sigma}_{n,i,j,a,b,o_1,...,o_m} \tag{4.12}$$

4.3 NNLOJET Workflow

The production of interpolation grids with NNLOJET is split into so-called *channels*. These channels correspond to different sets of Feynman diagrams in the perturbative expansion. The NNLOJET workflow with interpolation grids consists of the following steps:

Pre-processing The format and parameterization of the interpolation grids has to be established. Small test runs with a limited number of events are performed and analyzed in order to determine the optimal number of interpolation nodes, the interpolation order, etc.

Vegas integration The Vegas integration is used to optimize the event generation. It measures the accessed phase space and creates an approximate probability distribution function for each channel, that is later used to generate events.

Phase space determination The phase space is measured in terms of the momentum fraction x and the scale μ^2. This information is required in order to calculate the optimal placement of interpolation grid nodes in x and μ^2. The phase space calculation is reasonably fast, as the CPU expensive weight calculation is not required for this step.

Interpolation grid production The interpolation grid production is split into many parallel jobs. This step consumes the most resources and can take years of CPU time to complete.

Merging process The interpolation grids that were created by the previous step are combined into a final set of grids. After the merging process is complete, the output of all production jobs can be in principal deleted, thus greatly reducing the output size. For the case of a later statistical analysis, however, the individual results must be kept.

CHAPTER 5

Developing a Pipeline for Automated Production of Interpolation Grids

Creating interpolation grids with NNLOJET is tedious and very time consuming. The desire to reduce this workload has prompted the development of a pipeline that automates and streamlines the production of interpolation grids as a central part of this thesis. This has led to a significant increase in productivity and a reduction of human error. The pipeline also handles the creation of plots for cross checks of the interpolation quality and is accompanied by an intuitive web User Interface (UI) to evaluate these plots. This chapter contains a detailed description of the design and functionality of the pipeline.

5.1 Software Framework

LUIGI

LUIGI [40] is a Python framework that was developed by Spotify and is designed to build complex pipelines of batch jobs. LUIGI utilizes a central scheduler that schedules *tasks* to a number of worker nodes. A task may have dependencies on other tasks to be completed before it can be executed. Upon requesting a specific task to be run, the central scheduler will resolve all dependencies and schedule the necessary tasks to its worker nodes. This architecture allows for complex non-linear dependency graphs, since one does not have to compute the entire pipeline all at once.

Figure 5.1 outlines the basic structure of a task in LUIGI. Every task consists of three main methods: *requires*, *output* and *run*. The requires method returns all other tasks that are required to be completed before the run method can be executed. The output method defines the specific filename or directory in the local filesystem in which the output of this task will be stored. The run method contains the code to be executed by a worker node. LUIGI does not utilize a database to determine completed tasks, but will call the output method of a task to look up whether the specified target exists in your filesystem. The output of any task can therefore be supplied from external sources as well without having to run the specified task.

```
class MyTask(luigi.Task):
  # parameters for this task
  param = luigi.Parameter()

  def requires(self):
    # dependencies on other tasks
    return SomeOtherTask(self.param)

  def output(self):
    # output filename
    return luigi.LocalTarget('out.txt')

  def run(self):
    # business logic of the task
```

Figure 5.1: Outline of a basic Luigi task

law

law (Luigi analysis workflow) [41] offers a layer of abstraction on top of Luigi for run locations, storage locations and software environments. Of special interest for this work is law's built in support for large scale computations on the Worldwide LHC Computing Grid (WLCG). law can automatically submit to batch systems with HTCondor [42] and utilize remote dCache targets with all WLCG protocols (srm, xrootd, . . .).

Additionally, law makes use of so called *workflows*. A workflow consists of a number of branches of the same Luigi task. They are well suited for tasks that need to be run many times in parallel with different parameters. A workflow is considered successful when all of its branches completed successfully, or when the number of failed branches is below a specified threshold.

5.2 Design and Execution

This section describes the inner workings of the pipeline and how to use it.

5.2.1 Project Structure and Setup

The *nnlo-law-analysis* package uses Git version-control and comes with law, Luigi and six [43] (Python 2 and 3 compatibility library) as Git submodules. Instructions to install the project and run a test production can be found in Appendix A.1.

Required Software

- fastNLO— code for creation and evaluation of interpolation grids
- NNLOjet rev. 5088 — parton-level Monte Carlo generator
- NNLO-Bridge 0.0.40 — interface between fastNLO and NNLOjet
- HTCondor 8.6.5 — batch system
- GFAL2 [44] 2.15.4 — grid file access library
- Python 2.7 & 3.5

Required Analysis Files

The nnlo-law-analysis package requires a modified NNLOjet runcard and fastNLO steering files to run an analysis. The repository comes with an example runcard and steering file that can be used to make a simple test run.

The modified NNLOjet runcard can be created by taking a runcard that follows the NNLOjet specifications and inserting a few substitution strings into it. The syntax for these substitution strings is a name in capital letters surrounded by "@", e. g. *@CHANNEL@*.

The complete list of substitution strings is: @CHANNEL@, @EVENTS@, @ITERATIONS@, @SEED@, @WARMUP@, @PRODUCTION@, @UNIT_PHASE@, @REGION@

These strings denote values within the runcard that vary in different stages of the interpolation grid production. The example runcard that comes with the nnlo-law-analysis package already has these substitution strings in the correct positions and can be used as a template for other runcards. Alternatively, the runcard used for the Z+jet analysis in this thesis is in Appendix A.3 and also contains those strings. The fastNLO steering files require no such substitution, they only have to be compatible with the runcard.

5.2.2 Configuration

The Luigi configuration file is the centerpiece of the pipeline. Every task and workflow has a set of so-called *Luigi parameters* that can be set inside the configuration file. This allows for a lot of flexibility while making the results of an analysis reproducible, since the behavior of the pipeline is entirely determined by the configuration file. An example of a configuration file is shown in Figure 5.2.

The file is made up of different sections. Every parameter can be set inside of the [DEFAULT] section and every task will inherit these defaults. Parameters that have the same value for every task are supposed to be defined inside of that section, e. g. name, process, channels. Additionally, every task has a dedicated [TaskName] section for itself, in which parameters can be defined. Should a certain parameter be defined in both the default section and a task section, then the definition inside of the task section will take precedence. For instance, default HTCondor parameters should be configured inside

```
[DEFAULT]
name = ZJtriple
process = ZJ
channels = LO R V RRa RRb RV VV
wlcg_path = srm://cmssrm-kit.gridka.de:8443/srm/...
htcondor_accounting_group = cms.jet
htcondor_requirements = (TARGET.ProvidesCPU==true)
htcondor_request_cpus = 1
htcondor_request_memory = 4096
...

[Warmup]
warmup_events = 200000 10000 40000 5000 5000 10000 20000
warmup_iterations = 10 10 10 10 10 10 10
starting_seed = 0
htcondor_request_cpus = 20
htcondor_request_memory = 16384
...
```

Figure 5.2: Example of a Luigi configuration file.

of the [DEFAULT] section, and then overridden for individual tasks that have other requirements.

5.2.3 Scheduling and Resubmission of Workflows

The pipeline is composed of a total of 17 tasks and workflows. Each task constitutes a valid computation that can be executed by running

```
$ law run TaskName
```

Luigi takes care of required dependencies and schedules all missing tasks up to the requested task. The option flag *–print-status* prints out a list of required tasks and the status of their outputs.

```
$ law run TaskName --print-status -1
```

The number in the print-status argument denotes the depth of task dependencies that gets printed out, where 0 stands for the task itself and -1 for no depth limit.

Luigi does not have a database to keep track of task completions. Rather, it makes an inspection of the defined output of a task, and its existence on the filesystem defines whether the task has already been run successfully. In order to rerun a certain task, its output has to be deleted or moved. This can be done with:

```
$ law run TaskName --remove-output 0
```

5.2.4 Progressive Increase of the Number of Interpolation Grids

A full production at NNLO requires a large amount of computing resources, usually requiring over 100 000 CPU hours. Therefore, it is essential to be able to run the production with a smaller amount of events beforehand. The chosen solution to this problem is to set up a production with only a few jobs in each channel and to allow a progressive increase of that number after the workflow has already run. This also solves the problem of not knowing beforehand how many events are required for a statistically significant production.

Recommended Mode of Operation

1. Configure the pipeline with a few production jobs in each channel
2. Run the full pipeline, including the merging process and plotting
3. Evaluate the results and possibly start over with new adjustments
4. Increase the number of production jobs and rerun the production step, existing interpolation grids will not be resubmitted
5. Remove or relocate the output of the merging process and all plots in order to rerun this step
6. Evaluate the results and go to step 4. to further increase the number of production jobs

5.2.5 Limitations and Opportunities for Improvements

One of the limitations of the current design are the existence of untracked files. It is impossible to track every file individually, since the number of files can potentially exceed 100 000 files and that would lead to too many system calls. The natural solution is to store multiple output files of one task in a directory and only track the directory. However, the merging process and certain plotting scripts require a specific directory structure. These untracked files can only be removed manually, since law has no knowledge of their existence. This makes the pipeline harder to use and can cause problems if output is not removed correctly. Existing scripts that depend on a specific directory structure have to be rewritten to solve this issue.

Another issue that causes problems is the submission file in HTCondor jobs. Its only purpose is to track job IDs, but a bug in law causes the submission file not to be removed with the remove-output flag. The file has to be removed manually if the user wants to re-run an HTCondor workflow.

5.3 Implemented Workflows and Tasks

Workflows and tasks can be categorized into three main parts. The first one is the production of interpolation grids with NNLOjet over HTCondor. The second part manages the merging of interpolation grids created by the previous step. The last part of the pipeline consists of various plotting scripts.

A workflow in law is a task with a number of branches that can be computed in parallel. There are two types of workflows used in this project: *LocalWorkflow* does the computation of branches on the same computer as the scheduler, whereas *HTCondorWorkflow* sends the individual branch tasks over HTCondor for them to be computed on a batch system.

Table 5.1 shows a complete list of all the tasks and Figure 5.3 illustrates them in a graph with their dependencies. The following subsections give a more detailed description of the implemented tasks.

5.3.1 Interpolation Grid Production

The production of interpolation grids is done on a distributed computing cluster over HTCondor and its output is stored on a dCache fileserver. The production consists of four main steps:

Warmup *HTCondorWorkflow*
The Warmup workflow performs the Vegas integration for the defined phase space in the runcard. The number of events and iteration steps are defined for each channel. One multicore job per channel is submitted via HTCondor. The output of each branch is a tar archive containing the result of the Vegas integration.

FastWarm *HTCondorWorkflow*
The FastWarm workflow is responsible for the determination of the accessed phase space in x and Q^2. The resulting warmup table contains the minimum and maximum value of x and Q^2 in each bin. The number of events and the number of jobs for each channel can be specified. The output of each branch is a tar archive containing fastNLO warmup tables.

MergeFastWarm *Task*
The MergeFastWarm task manages the merging of all warmup tables created by the FastWarm workflow. It unzips all FastWarm tar files inside a temporary directory and then it merges the warmup tables for every observable. The merging procedure takes a

Table 5.1: List of implemented tasks and workflows

#	Name	Type	Req.	# Branches	Output
1	BaseRuncard	Task	—	—	The base runcard
2	Steeringfiles	Task	—	—	Tar archive with steering files
3	Warmup	Condor	1	N_{ch}	Tar archive with Vegas integration results
4	FastWarm	Condor	1,2,3	$\sum N_{jobs}$	Tar archive with warmup tables
5	MergeFastWarm	Task	4	—	Tar archive with merged warmup tables
6	FastProd	Condor	1,2,3,5	$\sum N_{jobs}$	Tar archive with interpolation grids
7	CopyTables	Local	6	$N_{FastProd}$	Merge directory with interpolation grids
8	Combine	Task	7	—	Weight files
9	MergeFastProd	Condor	8	$N_{ch} \cdot N_{obs}$	Merged channel interpolation grids
10	MergeFinal	Local	9	$N_{final} \cdot N_{obs}$	Final interpolation grids
11	FnloCppread	Local	7	$N_{FastProd}$	Interpolation grid log files
12	FnloCppreadFinal	Local	10	$N_{fin+ch} \cdot N_{obs}$	Final interpolation grid log files
13	SingleScalecheck	Local	11	$N_{ch} \cdot N_{obs}$	Single grid scalecheck plots
14	Approxtest	Local	8,11	$N_{ch} \cdot N_{obs}$	Multi grid scalecheck plots
15	Absolute	Local	12	$N_{final} \cdot N_{obs}$	Cross section plots
16	AbsoluteAll	Local	12	N_{obs}	Cross section plot with all orders
17	AllPlots	Task	13-16	—	—

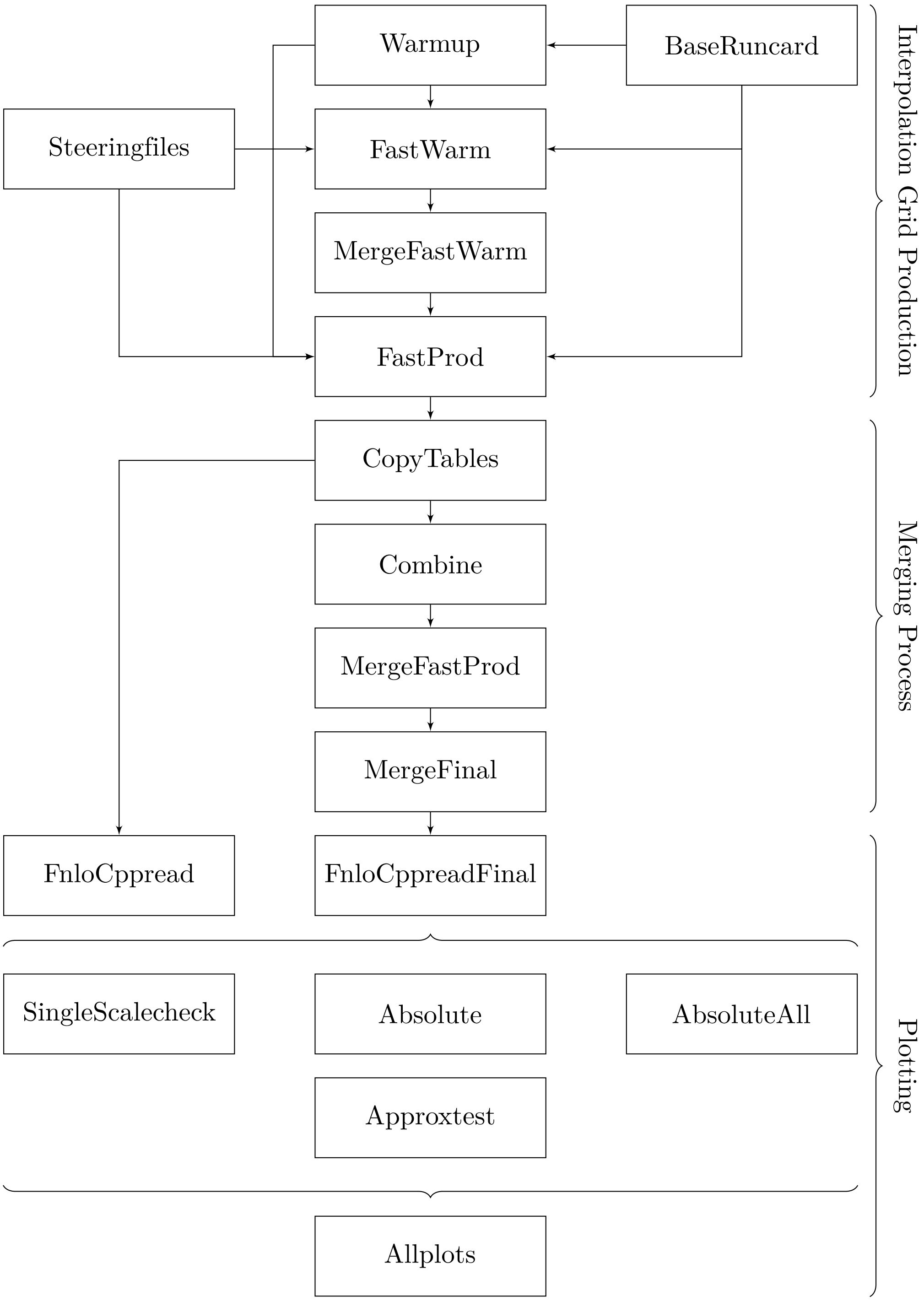


Figure 5.3: Task dependency graph. An arrow pointing from task A to task B signifies that task B requires the output of task A in order to run.

set of tables and calculates the minimum and maximum value for x and Q^2 in each bin. The output of this task is a tar archive containing the merged warmup tables.

FastProd *HTCondorWorkflow*
The FastProd workflow handles the production of interpolation grids. The number of events and jobs for each channel can be set for this workflow in the configuration file. The output of this task is a tar archive containing the interpolation grid files.

5.3.2 Merging Process

The merging process takes the large number of interpolation grids produced by the previous step and combines them into a single grid for each channel and observable. The merging of interpolation grids requires more effort than the merging of warmup tables and is therefore split into four tasks.

CopyTables *LocalWorkflow*
In order to be able to merge the interpolation grids created by the FastProd workflow, they have to be transferred to the local storage. The CopyTables workflow copies the tar archives from the FastProd workflow storage to the merging directory and unzips them. The interpolation grids of each channel are put in a separate folder named after the channel.

Combine *Task*
The Combine task executes the *nnlojet-combine.py* script from the NNLOJET repository on the ensemble of interpolation grids. In it, various weight files are calculated that are necessary for the next steps to correctly merge everything.

MergeFastProd *HTCondorWorkflow*
The MergeFastProd workflow merges all interpolation grids of a given observable and channel together with the weight files from the Combine task.

MergeFinal *Task*
The MergeFinal task is the last step in the merging process and creates the final set of interpolation grids. As its input it takes the interpolation grids created by the MergeFastProd workflow and combines multiple channels into one grid, as defined in the configuration file.

5.3.3 Plotting

As a part of this project, various plotting scripts from the FASTNLO repository have been integrated into the pipeline.

SingleScalecheck *LocalWorkflow*
Creates a scalecheck plot for every observable in every channel of the FastProd workflow. This scalecheck is performed on the interpolation grid with the first seed in each channel. An example of a Scalecheck plot is shown in Figure 5.4. It shows the ratio and the asymmetry of the cross sections from the interpolation grid and from NNLOJET. Ideally this would be exactly 1, but there will always be a small bias, because the interpolation technique is an approximation.

Approxtest *LocalWorkflow*
The Approxtest workflow creates plots on the ensemble of all interpolation grids of a given channel and observable. Figure 5.5 and 5.6 are examples of these plots. They show the ratio and the asymmetry of the interpolation grid cross section and the NNLOJET cross section.

Absolute *LocalWorkflow*
The Absolute workflow creates a plot for every merged interpolation grid. Figure 5.7 is an example for a plot that was created by this workflow. It shows the absolute cross section of a given interpolation grid that was evaluated with a given PDF. This cross section is compared to the original cross section calculated by NNLOJET. The ratio of the two cross sections is shown in the lower plot.

AbsoluteAll *LocalWorkflow*

The AbsoluteAll workflow creates a similar output as the Absolute workflow, but it combines the contributions of all channels into one plot. It also shows the *K factor*, which is a measure for the relative size of the contributions:

$$\frac{\sigma_{\mathrm{NLO}}}{\sigma_{\mathrm{LO}}} \text{ and } \frac{\sigma_{\mathrm{NNLO}}}{\sigma_{\mathrm{NLO}}} \tag{5.1}$$

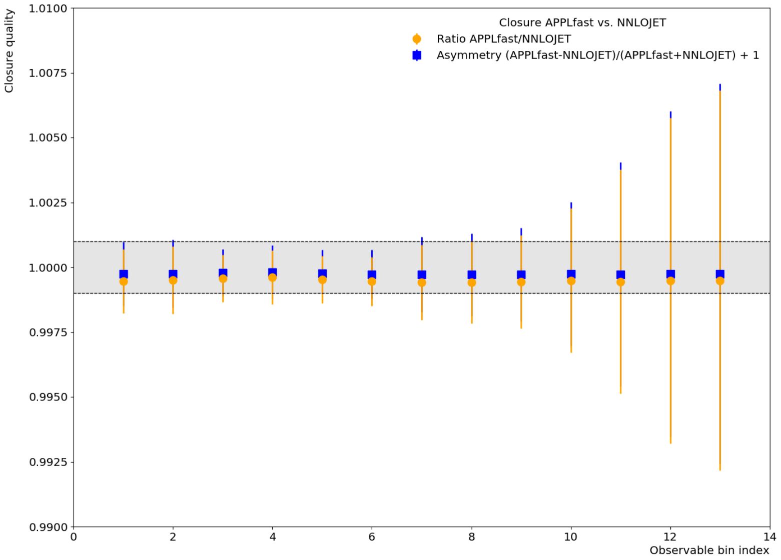


Figure 5.4: Example SingleScalecheck plot. Comparison between a single interpolation grid and the NNLOJET data file. It shows the ratio (yellow) and the asymmetry (blue) of the two cross sections. The vertical bars denote the statistical uncertainty of the NNLOJET calculation. However, the interpolation bias is not affected by the statistical uncertainty, because the comparison is between the same events in both cases.

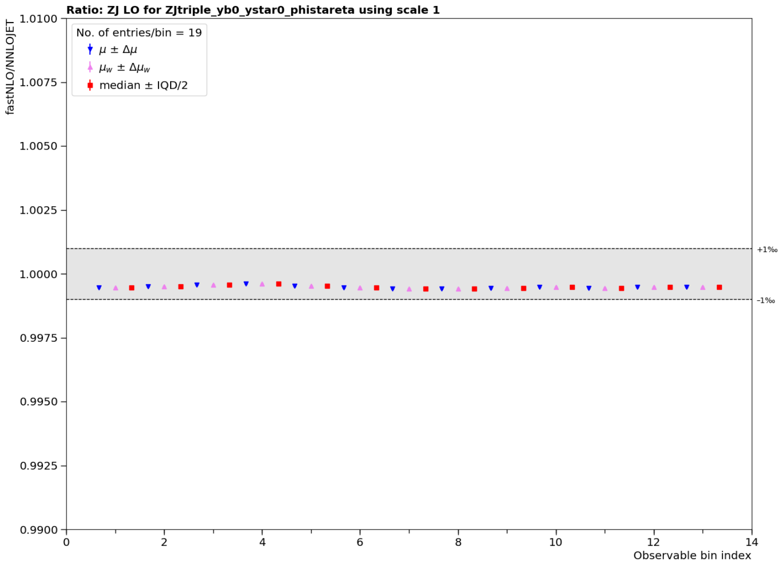


Figure 5.5: Example of a ratio plot created by the Approxtest workflow. The plot shows the mean (blue), the weighted mean (pink) and the median (red) of the ratio of the interpolation grid cross section and the NNLOJET cross section. These are calculated from the ensemble of all interpolation grids of a given channel and observable.

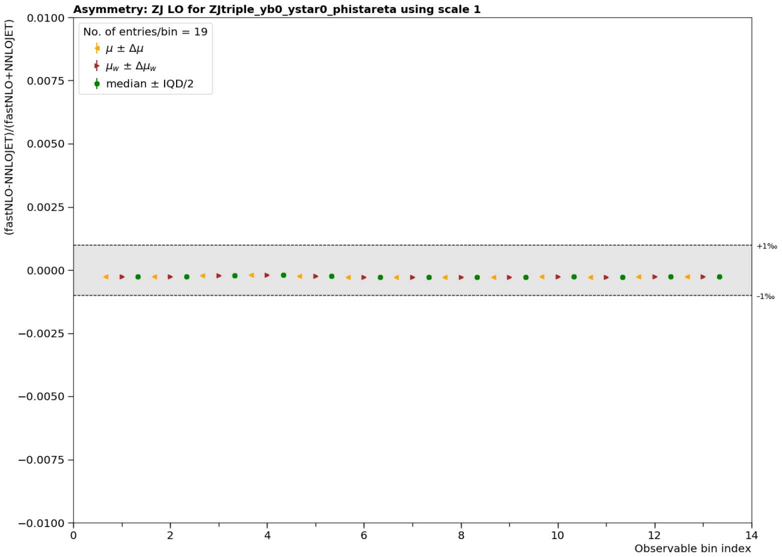


Figure 5.6: Example of an asymmetry plot created by the Approxtest workflow. Same as Figure 5.5, but it shows the asymmetry instead of the ratio.

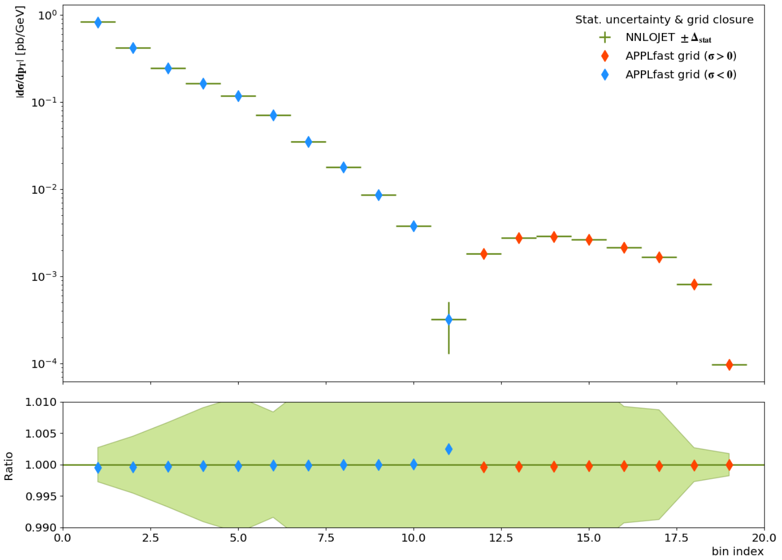


Figure 5.7: Example plot created by the Absolute workflow. The absolute value of the calculated cross section by NNLOJET (green) is plotted with the cross section as predicted by the interpolation grid (orange - positive, blue - negative). The ratio of the two cross sections is shown in the lower plot. The interpolation quality is worse at zero-crossings, but these have a smaller effect on the result because of the small cross section value. The interpolation grids are in good agreement with the NNLOJET data in this example.

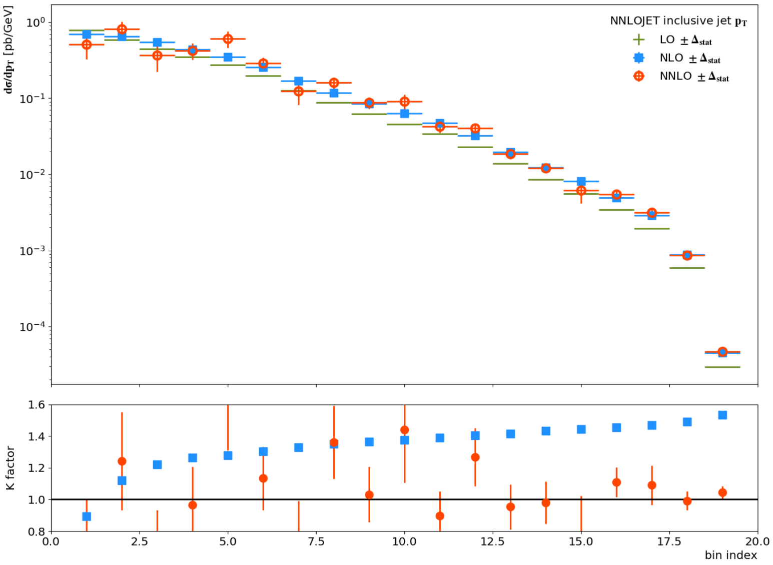


Figure 5.8: Example plot created by the AbsoluteAll workflow. The cross section (top) and the K factors (bottom) $\sigma_{\mathrm{NNLO}}/\sigma_{\mathrm{NLO}}$ (red) and $\sigma_{\mathrm{NLO}}/\sigma_{\mathrm{LO}}$ (blue) are shown.

5.4 Web Application for Generated Plots

Because of the sheer amount of plots automatically generated by the pipeline, one cannot go through every plot one by one. Depending on how many interpolation grids exist in a given production, the number of plots may potentially be in the thousands. An application has therefore been developed in order to navigate through the vast number of plots. The advantage of using web based technologies for this use case is that the plots will be accessible from everywhere. This makes it particularly easy to share and discuss the content with colleagues.

Although it can be used seamlessly with pipeline results, the web application is not explicitly part of the pipeline, but a standalone application. This design decision comes with some major advantages. Further development is made substantially easier this way and the application can be still used without the pipeline. Additionally, it can be used for completely unrelated analyses in which the user needs to filter through a large number of plots. The application is written in JavaScript and divided into two parts, which will be discussed in detail in the following two subsections. Instructions to install and run the project can be found in Appendix A.2.

5.4.1 Front-End User Interface

The UI of this application is coded in React [45], a JavaScript library for dynamic websites. Figure 5.9 shows a screenshot of the UI.

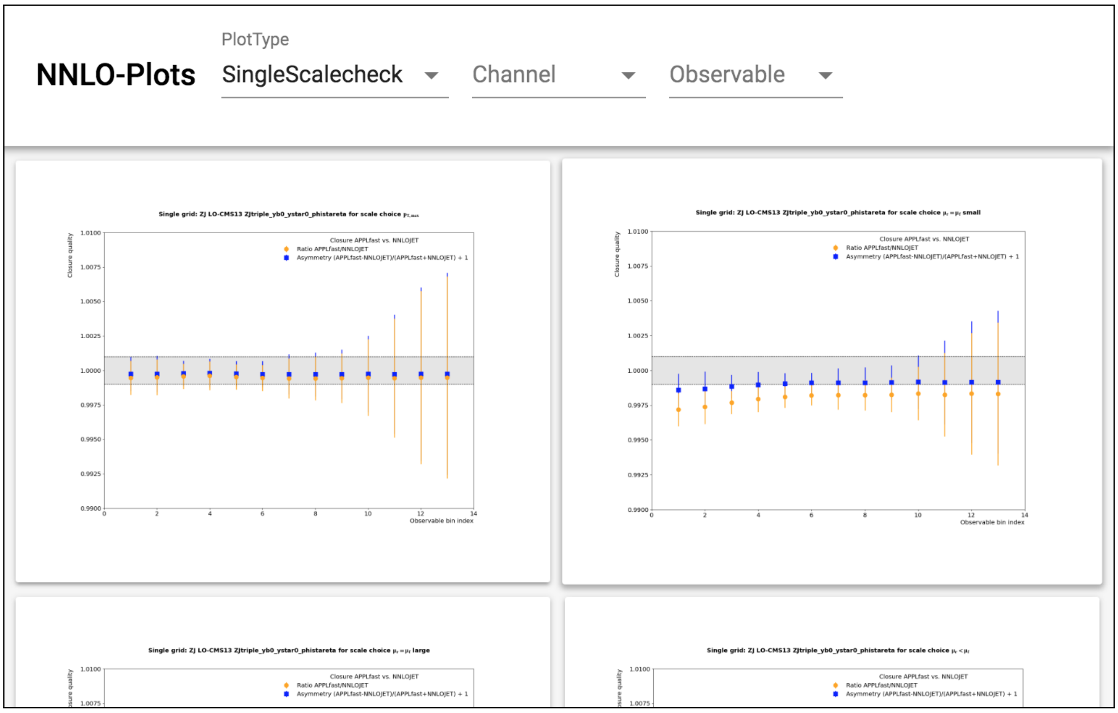
Upon starting the website the client will fetch a list of all available plots and then display them to the user inside a so-called *infinite image masonry*. It is basically a grid of images that also allows for variable image aspect ratios without creating whitespaces between them. Images are fetched progressively as the user scrolls down. The bar at the top contains text fields to filter for certain content.

5.4.2 Back-End Server

Node [46] is a JavaScript run-time environment that allows for server-side scripting and was used to write the back-end. The back-end serves two main functions for the application.

static content The first one is to serve static content, which is located in the subdirectory *public*. The plots created by the pipeline are supposed to be dumped into *public/plots* and can then be fetched by the front-end by a simple https get request.

list of all available plots In principle, serving plots to the client could have been done without the need of a dedicated back-end server, however it would be impossible to know the list of all available plots. This functionality was added to the back-end and can be accessed through a get request with the path */plots*. Upon requesting it, the server will then recursively scan the plots directory and return a list of paths to all the files



NNLO-Plots

PlotType: SingleScalecheck

Channel
Observable
Single grid: ZJ LO-CMS13 ZJtriple_yb0_ystar0_phistareta for scale choice $p_{T,\max}$
Single grid: ZJ LO-CMS13 ZJtriple_yb0_ystar0_phistareta for scale choice $\mu_r = \mu_f$ small
Single grid: ZJ LO-CMS13 ZJtriple_yb0_ystar0_phistareta for scale choice $\mu_r = \mu_f$ large
Single grid: ZJ LO-CMS13 ZJtriple_yb0_ystar0_phistareta for scale choice $\mu_r < \mu_f$
Closure quality
Closure APPLfast vs. NNLOJET
Ratio APPLfast/NNLOJET
Asymmetry (APPLfast-NNLOJET)/(APPLfast+NNLOJET) + 1
1.0100
1.0075
1.0050
1.0025
1.0000
0.9975
0.9950
0.9925
0.9900
0
2
4
6
8
10
12
14
Observable bin index

Figure 5.9: Screenshot of the web UI. Plots are displayed in an infinite layout. The bar at the top contains text fields to filter for certain content.

inside it. The client can then use this information to fetch any plot inside that list by simply requesting it by its path.

CHAPTER 6

Z+Jet Interpolation Grid Production

The pipeline from Chapter 5 is used to perform the example Z+jet analysis from Chapter 3. This chapter describes the steps performed for the production of Z+jet interpolation grids and gives a detailed look into the results.

6.1 Pre-Processing

The choice for an appropriate interpolation grid format is a balancing act between the desire to store as much information as possible and the constraint to stay within the resource limits. After extensive testing, a flexible-scale format has been defined for this production. The two scales chosen are the transverse energy E_T^Z and the total activity H_T^Z.

$$E_T^Z = \sqrt{(M^{\mathrm{Z}})^2 + (p_{\mathrm{T}}^{\mathrm{Z}})^2} \tag{6.1}$$

$$H_T^Z = E_T^Z + \sum_i^{\text{partons}} p_T^i \tag{6.2}$$

The next step is to create a range of interpolation grids with different settings for the scale and x-nodes. These are compared against the original cross section results from NNLOJET. Figure 6.1 shows two scalecheck plots of one rapidity bin in LO with a different number of x-nodes, but otherwise identical setup. The interpolation bias becomes smaller as more nodes are used, but the required disk space gets larger. After evaluating all plots, with the available resources in mind, the final number of nodes is set to 4 scale nodes and 30 x-nodes.

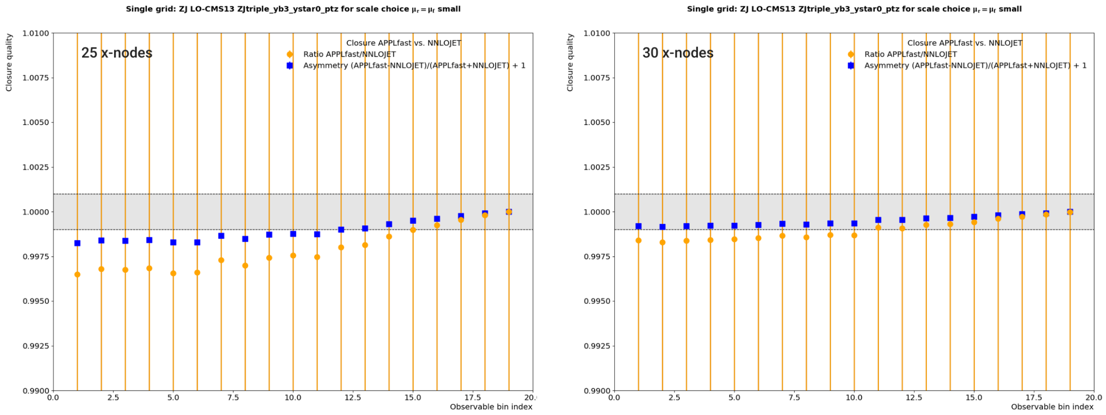


Figure 6.1: Comparison of interpolation quality with 25 x-nodes (left) and 30 x-nodes (right).

6.2 Resource Consumption

The following computing resources have been used to perform the Z+jet interpolation grid production:

- BwForCluster NEMO [47], University of Freiburg (bwHPC)
- Helix Nebula Science Cloud [48]
- ETP Resources, Karlsruhe Institute of Technology (KIT)
- Grid Computing Centre Karlsruhe (GridKa) [49]

Table 6.1 contains a detailed breakdown of the computing resources that were consumed in each step of the interpolation grid production. The $p_{\mathrm{T}}^{\mathrm{Z}}$ and ϕ_{η}^{*} grids were calculated separately in the production step, because of memory constraints, resulting in a runtime that is twice as high. In total, 8.4 billion events were generated during the production step, consuming roughly 480 days of CPU time with an output of 947 GB.

Table 6.1: Resource consumption of the Z+jet grid production

				NLO		NNLO				
Step	Stat	Unit	LO	R	V	RRa	RRb	RV	VV	Total
Warmup	Iterations		10	10	10	10	10	10	13	—
	Events	M	50	25	25	4	4	4	4	—
	Threads		24	24	24	24	24	24	24	—
	Runtime	h	4	5	5	46	46	72	80	258
FastWarm	Jobs		10	10	10	10	10	10	10	70
	Events/Job	M	200	200	200	100	100	100	100	—
	Events	G	2	2	2	1	1	1	1	10
	Runtime	d	10	10	10	10	10	10	10	70
FastProd	Jobs		20	50	50	10	10	50	50	240
	Events/Job	M	250	45	22	0.8	0.7	0.25	0.8	—
	Events	G	5	2.25	1.1	0.008	0.007	0.012	0.04	8.4
	Output/Job	GB	0.08	4.1	3.5	4.3	3.8	5.5	4.2	—
	Output	GB	1	205	175	43	38	275	210	947
	Runtime	d	40	100	100	20	20	100	100	480

6.3 Interpolation Quality

The resulting interpolation quality at NLO is excellent, however at NNLO it needs improvement. For this reason, only a relatively small number of jobs have been computed for the NNLO channels. The final production in these channels is delayed until some further improvements have been implemented.

Moreover, the RR channels suffer from fluctuating Vegas integration results. The calculated probability function in the Vegas integration does not converge after 10, or even 13 steps. This issue is likely to be resolved by tweaking the parameters for the Vegas integration until an acceptable result is computed.

The VV channel has a large interpolation bias in the central scale. Figure 6.2 shows the ratio of the interpolation over the NNLOjet cross section in one of the rapidity bins. The ratio exceeds the 1 % mark, rendering it useless for precision fits with experimental data. Further studies are required to identify the cause of this discrepancy in the VV channel.

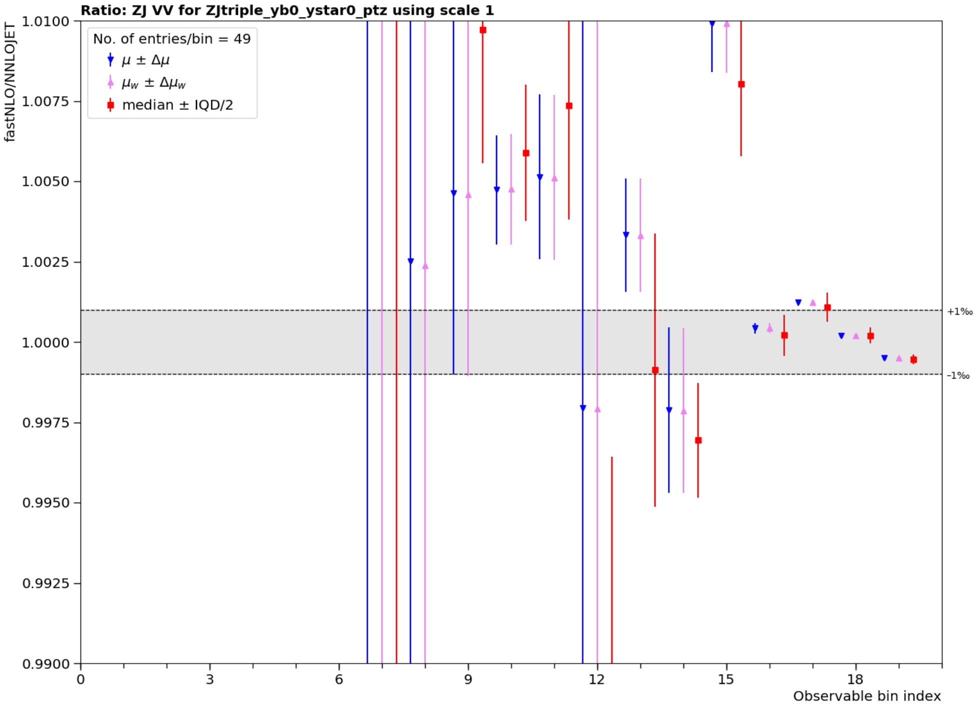


Figure 6.2: Problematic interpolation results in the VV channel. The ratio of the interpolation over the NNLOJET cross section is plotted for the central rapidity bin.

6.4 Analysis of Interpolation Grids at NLO

6.4.1 Partonic Subprocesses

A subject of interest for the analysis of the Z+jet cross section is the composition of partonic subprocesses. The results are presented in the following four plots: Figure 6.3 shows the subprocess composition at LO in ϕ^*_η, Figure 6.4 at NLO in ϕ^*_η, Figure 6.5 at LO in p_T^Z and Figure 6.6 at NLO in p_T^Z.

Dominant contribution As expected from theoretical arguments, the quark-gluon channel has the largest contribution to the cross section, followed by the antiquark-gluon and quark-antiquark channel. These channels are already present at LO and continue to dominate at NLO, while others are only present at NLO, namely quark-quark, antiquark-antiquark and gluon-gluon. The largest contribution out of those channels comes from the quark-quark channel, followed by a negative contribution from the gluon-gluon channel and a barely visible antiquark-antiquark contribution.

Distribution in rapidity bins The relative size of the NLO contribution increases with higher y^* bins, but stays constant across y_b bins. This effect is largely due to the increase of quark-quark processes in higher y^* bins. The event topology allows for higher parton momenta in high y^* bins, thus increasing the occurrence of quark-quark processes.

Distribution in p_T^Z/ ϕ^*_η Events in higher p_T^Z/ ϕ^*_η bins originate from partons with higher momenta. This leads to an increase in quark-quark processes and therefore a larger NLO contribution.

6.4.2 Triple-Differential Cross Section

Figure 6.7 shows a plot of the triple-differential Z+jet cross section. The cross sections that were calculated by NNLOJET are drawn by lines, while the markers represent the evaluated interpolation grids. The value in each of the rapidity bins is multiplied by a different decimal power to improve readability. This plot highlights the good agreement of the cross sections from NNLOJET with those from the interpolation grids, indicating a negligible bias of less than 1 ‰.

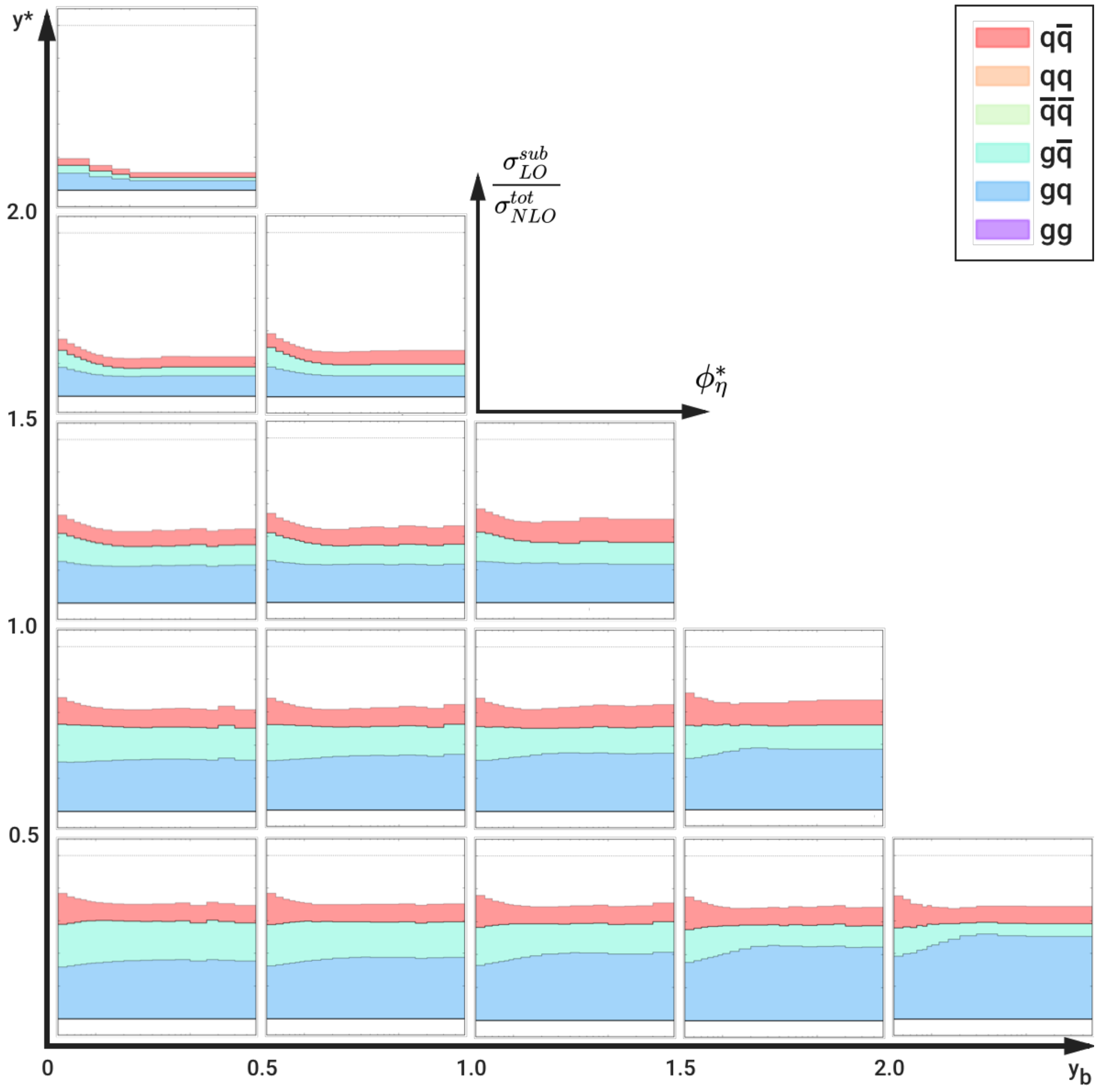


Figure 6.3: Subprocess composition of the LO cross section in ϕ_η^*, normalized to the total NLO cross section. Note: top y^* bin has a smaller maximum ϕ_η^*.

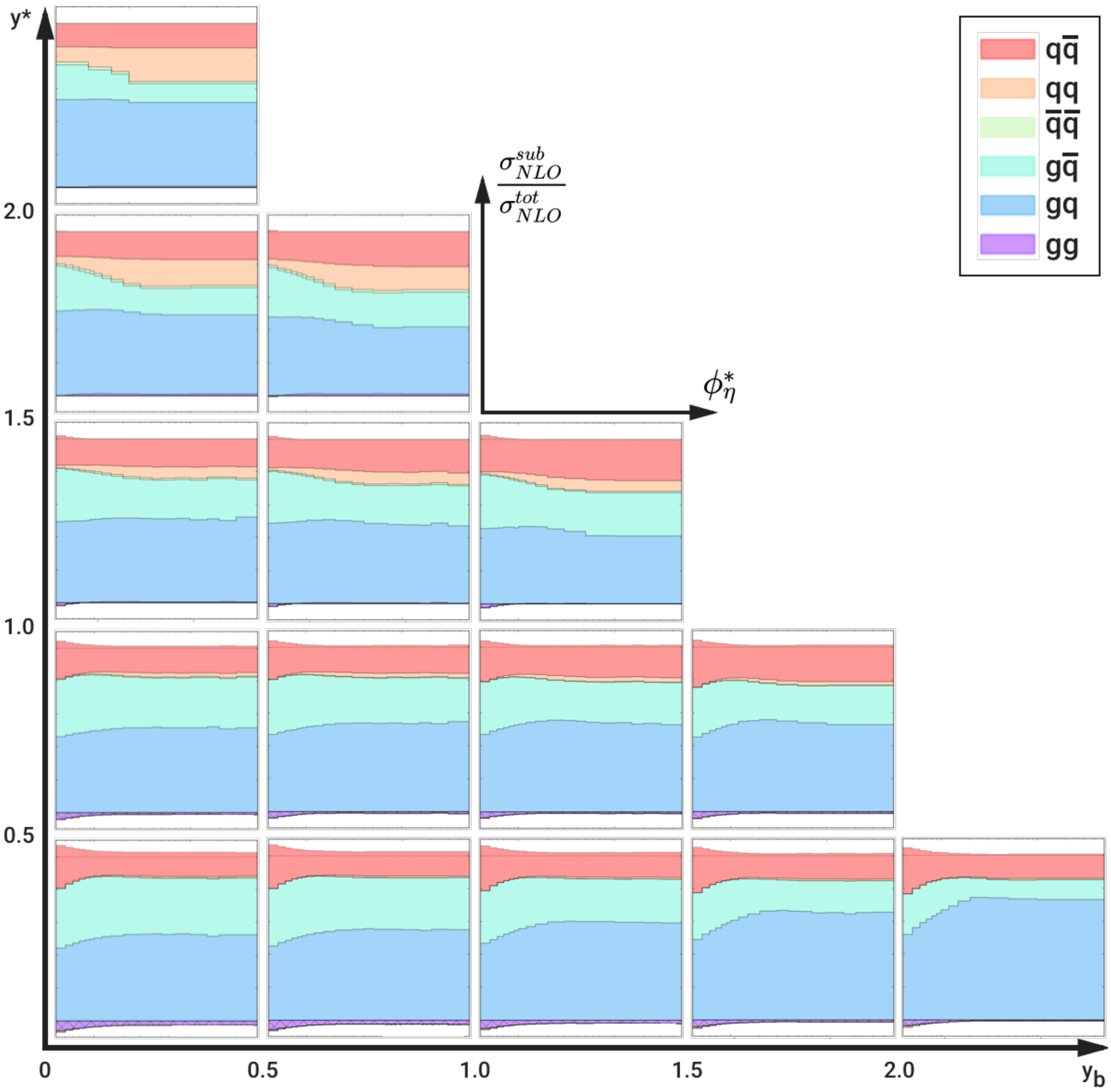


Figure 6.4: Subprocess composition of the NLO cross section in ϕ_η^*, normalized to the total NLO cross section. Note: top y^* bin has a smaller maximum ϕ_η^*.

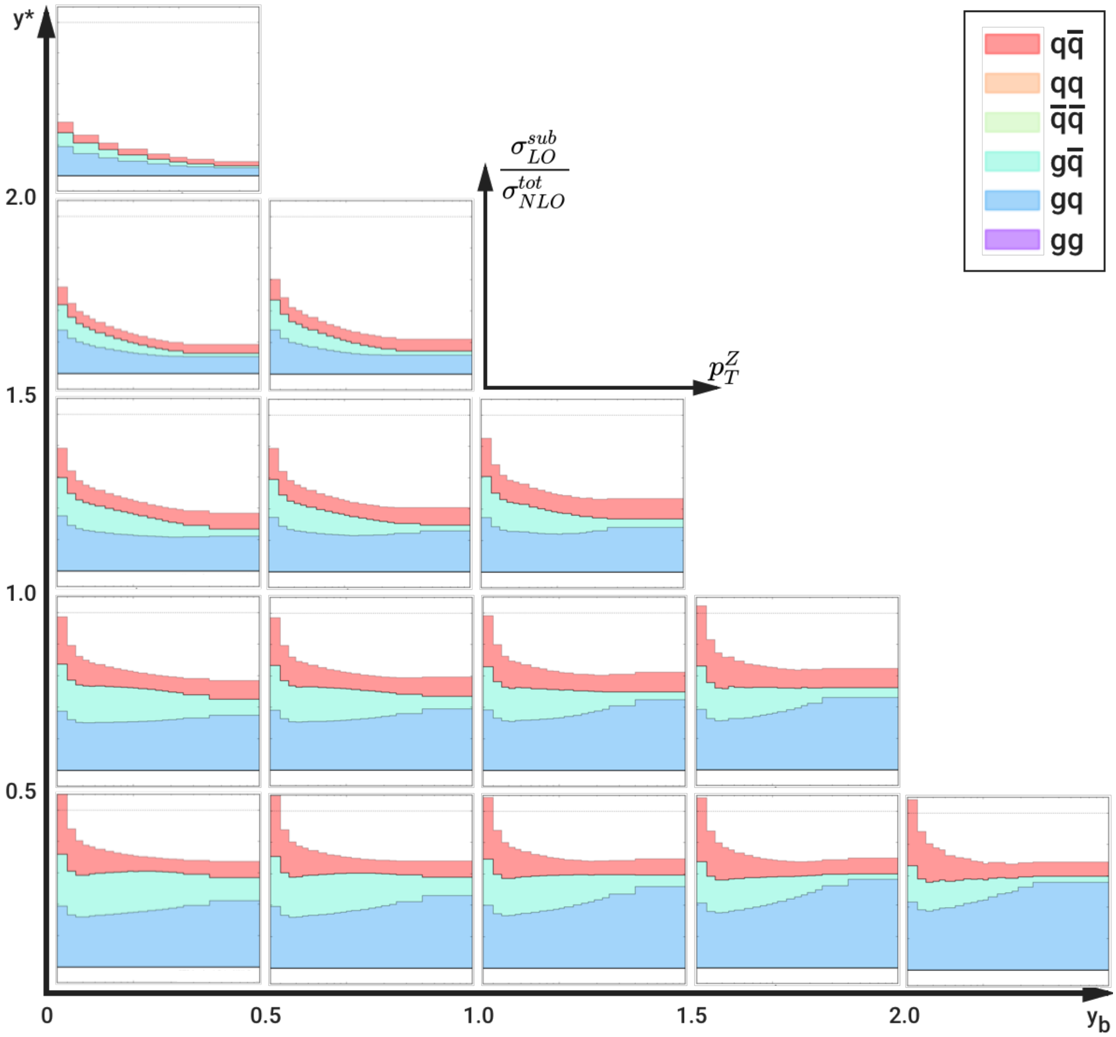


Figure 6.5: Subprocess composition of the LO cross section in $p_{\mathrm{T}}^{\mathrm{Z}}$, normalized to the total NLO cross section. Note: top y^* bin has a smaller maximum $p_{\mathrm{T}}^{\mathrm{Z}}$.

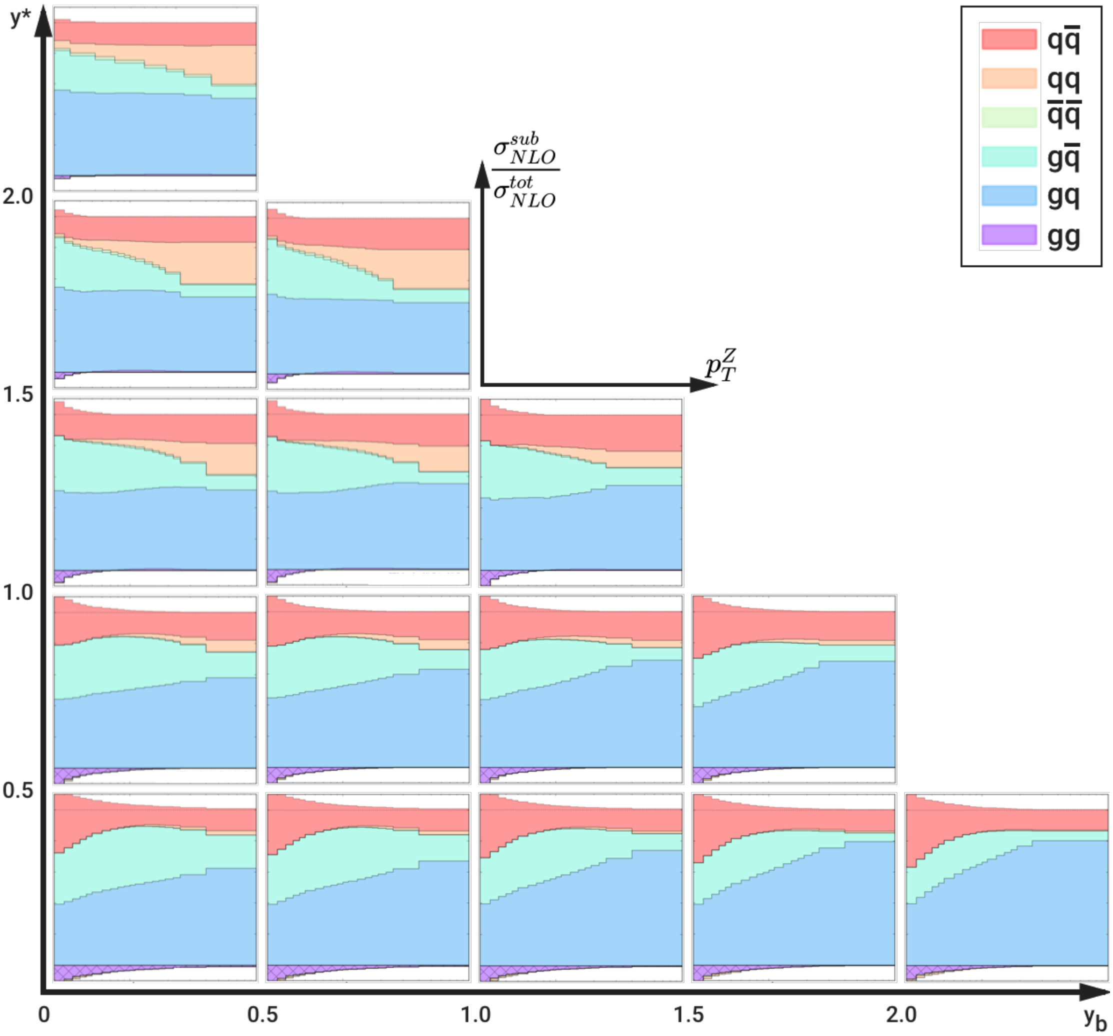


Figure 6.6: Subprocess composition of the NLO cross section in $p_{\mathrm{T}}^{\mathrm{Z}}$, normalized to the total NLO cross section. Note: top y^* bin has a smaller maximum $p_{\mathrm{T}}^{\mathrm{Z}}$.

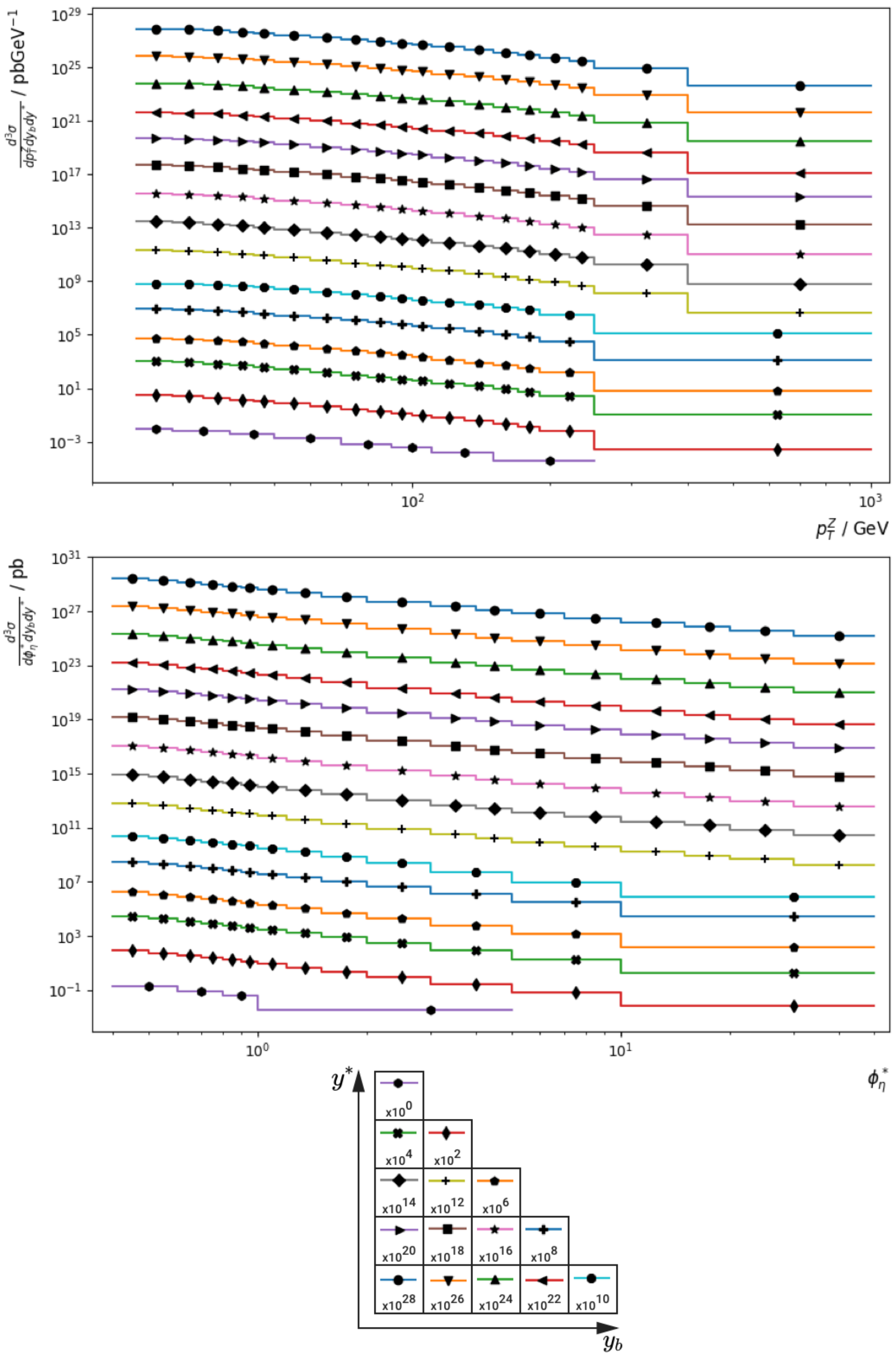


Figure 6.7: Plot of the triple-differential Z+jet cross section at NLO. The results from NNLOJET (lines) are compared to the cross sections from evaluated interpolation grids (markers). The cross section of each rapidity bin is multiplied by a different decimal power to improve readability.

CHAPTER 7

Conclusion and Outlook

In this thesis, an automated workflow for the production of interpolation grids has been designed and implemented in the form of a pipeline. This has led to an improvement in productivity. The setup is compatible with any NNLOJET runcard and can be applied to a wide range of processes. It handled the production of interpolation grids for the Z+jet analysis in this thesis successfully, thus demonstrating a working proof of concept.

Interpolation grids for Z+jet events at $\sqrt{s} = 13\,\mathrm{TeV}$ have been filled at NNLO. Since some further studies are required for the NNLO part, however, these grids were analyzed at NLO, due to a better interpolation quality. The interpolation grids are in agreement with the NNLOJET results with an interpolation bias of less than 1 ‰.

The NNLO contribution requires some tweaking before interpolation grids with a large number of events can be filled. Improvements are necessary in the Vegas integration in the RR channels and the interpolation quality of the VV channel. Once the NNLO setup has been improved, a final set of interpolation grids will be produced. These will then be combined with experimental data from the CMS detector to give new insights into the proton structure.

The pipeline is open to further testing and development. New tasks and workflows can easily be added to accommodate a wider range of needs. The production of interpolation grids of various different processes with this pipeline is possible already now. The use of the pipeline will lead to a reduction in manpower when running large-scale productions of interpolation grids at NNLO for various processes.

APPENDIX A

Appendix

A.1 nnlo-law-analysis

Project setup

To clone the project, including its submodules, execute the following command:

```
$ git clone https://github.com/miguel-sc/nnlo-law-analysis.git
   --recursive
```

Edit setup.sh, analysis/bootstrap.sh and analysis/multicore_bootstrap.sh to source the correct paths.

```
$ source setup.sh
```

Open luigi.cfg and edit wlcg_path and htcondor_user_proxy.

To enable the built in auto completion tools from LAW, execute this command once:

```
$ law db
```

Interpolation grid creation

Test htcondor setup by first running:

```
$ law run Warmup
```

Recommended to run the grid production up to Combine task in single threaded mode:

```
$ law run Combine
```

Run the rest multithreaded:

```
$ law run AllPlots --workers 10
```

A.2 nnlo-law-website

Project setup

Clone the project with:

```
$ git clone https://github.com/miguel-sc/nnlo-law-website.git
```

Open two shells and go to each of the subdirectories, client and server. Then install the dependencies in both directories:

```
$ npm install
```

Compiles and hot-reloads for development

Run development server in both subdirectories with:

```
$ npm start
```

The website can be accessed under port 3000 by default and makes requests to the node server under port 3001.

Populate the server with plots

Create a new directory called:

```
server/public/plots
```

Add your png images to this directory with the following naming scheme:

```
process/plottype/channel/observable/name.png
```

The default naming scheme can be changed and is defined inside:

```
client/src/constants.js
```

Compiles and minifies for production

Build the client application for production:

```
$ npm run build
```

The build folder can then be served by any http server.

A.3 NNLOJET Runcard

```
NNLOJET_RUNCARD
@CHANNEL@-CMS13       ! Job name id
ZJ                    ! Process name
@EVENTS@              ! Number of events
@ITERATIONS@          ! Number of iterations
@SEED@                ! Seed number
.@WARMUP@.            ! Warmup
.@PRODUCTION@.        ! Production
CT14nnlo              ! PDF set
0                     ! PDF member
antikt                ! Jet algorithm, accepts ’kt’, ’cam’ or ’antikt’
0.4d0                 ! Rcut
.false.               ! exclusive
1                     ! Heavy particle decay type
1d-8                  ! Technical cutoff y0
.false.               ! angular averaging of the phase space, default to true
2                     ! Virtual Integration method, use 2 if you are not sure
@REGION@              ! RR a/b region flag. Accepts ’a’,’b’ or ’all’.
0                     ! set to zero for MC, 2 for point test
.false.               ! print max weight flag
.true.                ! momentum mapping and PDF storage flag
.false.               ! colour sampling flag
```

```
.false.               ! explicit pole check flag, stops integration when set to true

PHYSICAL_PARAMETERS
13000d0               ! roots
125.09d0              ! Mass of the Higgs Boson
0.004029643852d0      ! Width of the Higgs Boson
91.1876d0             ! Mass of the Z Boson
2.4952d0              ! Width of the Z Boson
80.385d0              ! Mass of the W Boson
2.085d0               ! Width of the W Boson
173.21d0              ! Mass of the Top Quark
1.41d0                ! Width of the Top Quark
4.18d0                ! Mass of the Bottom Quark
0d0                   ! Width of the Bottom Quark
1.275d0               ! Mass of the Charm Quark
0d0                   ! Width of the Charm Quark
1.777d0               ! Mass of the Tau lepton
0d0                   ! Width of the Tau lepton

@UNIT_PHASE@

SELECTORS

        !----- jet veto
        select jets_abs_eta max = 2.4
        select jets_pt min = 20

        !----- for Z production we have two leptons (l^[+-] = l[pm])
        select abs_ylp max = 2.4
        select abs_ylm max = 2.4
        select ptlp min = 25
        select ptlm min = 25

        !----- V = {lp, lm}
        select mll min = 71.1876 max = 111.1876

        !----- for inclusive ptz
        select njets min = 1

END_SELECTORS


```

```

HISTOGRAMS

        phi_star > ZJtriple_yb0_ystar0_phistareta [0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.2, 1.5, 2.0, 3.0, 4.0, 5.0, 7.0, 10, 15, 20, 30, 50] grid = ZJtriple_yb0_ystar0_phistareta.fast
        HISTOGRAM_SELECTORS
                select yboost_Zj max = 0.5
                select ystar_Zj max = 0.5
        END_HISTOGRAM_SELECTORS

        ptz > ZJtriple_yb0_ystar0_ptz [25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 110, 130, 150, 170, 190, 220, 250, 400, 1000] grid = ZJtriple_yb0_ystar0_ptz.fast
        HISTOGRAM_SELECTORS
                select yboost_Zj max = 0.5
                select ystar_Zj max = 0.5
        END_HISTOGRAM_SELECTORS

        phi_star > ZJtriple_yb0_ystar1_phistareta [0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.2, 1.5, 2.0, 3.0, 4.0, 5.0, 7.0, 10, 15, 20, 30, 50] grid = ZJtriple_yb0_ystar1_phistareta.fast
        HISTOGRAM_SELECTORS
                select yboost_Zj max = 0.5
                select ystar_Zj min = 0.5 max = 1.0
        END_HISTOGRAM_SELECTORS

        ptz > ZJtriple_yb0_ystar1_ptz [25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 110, 130, 150, 170, 190, 220, 250, 400, 1000] grid = ZJtriple_yb0_ystar1_ptz.fast
        HISTOGRAM_SELECTORS
                select yboost_Zj max = 0.5
                select ystar_Zj min = 0.5 max = 1.0
        END_HISTOGRAM_SELECTORS

        ...

END_HISTOGRAMS



SCALES
```

```

        muf = z_ht mur = etz
        muf = 2.718281828 mur = 2.718281828
        muf = 4.48168907 mur = 4.48168907
        muf = 4.48168907 mur = 2.718281828
        muf = 2.718281828 mur = 4.48168907
        muf = 12.18249396 mur = 2.718281828
        muf = 2.718281828 mur = 12.18249396

END_SCALES



REWEIGHT ht_part**2



CHANNELS

        @CHANNEL@

END_CHANNELS

SETUP
END_SETUP
```

A.4 fastNLO Steering File

```
ScaleDescriptionScale1          "etz_[GeV]"
ScaleDescriptionScale2          "zht_[GeV]"

ScenarioName                    ZJ.ZJtriple
PublicationUnits                12

GlobalVerbosity                 INFO
FlexibleScaleTable              true

X_NNodes                        30
Mu1_NNodes                      4
Mu2_NNodes                      4
```

A.5 Luigi Configuration File

```
[core]
```

```

no_lock = True

[worker]

keep_alive = False
ping_interval = 20
wait_interval = 20
max_reschedules = 0

[DEFAULT]

# name of your analysis
name = ZJtriple

# NNLOJET process
process = ZJ

# NNLOJET channels (append "a", "b" for RR region flag)
channels = LO R V RRa RRb RV VV

# merged grids (make sure it’s compatible with your combine.ini config)
final_tables = {
    "NLO": ["LO", "R", "V"],
    "NLO_only": ["R", "V"],
    "NNLO_only": ["RRa", "RRb", "RV", "VV"],
    "NNLO": ["LO", "R", "V", "RRa", "RRb", "RV", "VV"]
  }

# list of all observables
observables = [
    "ZJtriple_yb0_ystar0_phistareta",
    "ZJtriple_yb0_ystar1_phistareta",
    "ZJtriple_yb0_ystar2_phistareta",
    "ZJtriple_yb0_ystar3_phistareta",
    "ZJtriple_yb0_ystar4_phistareta",
    "ZJtriple_yb1_ystar0_phistareta",
    "ZJtriple_yb1_ystar1_phistareta",
    "ZJtriple_yb1_ystar2_phistareta",
    "ZJtriple_yb1_ystar3_phistareta",
    "ZJtriple_yb2_ystar0_phistareta",
    "ZJtriple_yb2_ystar1_phistareta",
    "ZJtriple_yb2_ystar2_phistareta",
    "ZJtriple_yb3_ystar0_phistareta",
    "ZJtriple_yb3_ystar1_phistareta",
    "ZJtriple_yb4_ystar0_phistareta",
    "ZJtriple_yb0_ystar0_ptz",
    "ZJtriple_yb0_ystar1_ptz",
    "ZJtriple_yb0_ystar2_ptz",
    "ZJtriple_yb0_ystar3_ptz",
    "ZJtriple_yb0_ystar4_ptz",
    "ZJtriple_yb1_ystar0_ptz",
    "ZJtriple_yb1_ystar1_ptz",
    "ZJtriple_yb1_ystar2_ptz",
    "ZJtriple_yb1_ystar3_ptz",
    "ZJtriple_yb2_ystar0_ptz",
    "ZJtriple_yb2_ystar1_ptz",
    "ZJtriple_yb2_ystar2_ptz",
    "ZJtriple_yb3_ystar0_ptz",
    "ZJtriple_yb3_ystar1_ptz",
    "ZJtriple_yb4_ystar0_ptz"
  ]

# dcache path, output will be created inside $wlcg_path/$name
wlcg_path = srm://cmssrm-kit.gridka.de:8443/srm/managerv2?SFN=/pnfs/gridka.de/cms/disk-only/store/user/msantosc

# htcondor job config
htcondor_accounting_group = cms.jet
htcondor_requirements = (TARGET.ProvidesCPU==true)
htcondor_remote_job = True
htcondor_user_proxy = /tmp/x509up_u12225
htcondor_walltime = 84000
htcondor_request_cpus = 1
htcondor_request_memory = 4096
htcondor_universe = docker
htcondor_docker_image = mschnepf/slc6-condocker

# create log files in htcondor jobs
transfer_logs = True

# set local scheduler
local_scheduler = True

# set tolerance for workflow success with failed branches
tolerance = 0

```

```
# submit only missing htcondor workflow branches (should always be true)
only_missing = True

# bootstrap file to be sourced at beginning of htcondor jobs
bootstrap_file = bootstrap.sh

# directory for merging of grids inside $merge_dir/$name
merge_dir = /ceph/mcorrea/mergedgrids

# fastNLO cppread
pdf = CT14nnlo
scalecombs = -6
ascode = LHAPDF
norm = no
scale = scale12

# directory for plots
plots_dir = /ceph/mcorrea/plots

[BaseRuncard]

# path to base runcard file
source_path = /ceph/mcorrea/analysis/ZJtriple/ZJ.ZJtriple.run

[Steeringfiles]

# directory with all steering files
source_path = /ceph/mcorrea/analysis/ZJtriple/steeringfiles

[Runcard]

[Warmup]

# override default htcondor config
htcondor_request_cpus = 24
bootstrap_file = multicore_bootstrap.sh
htcondor_request_memory = 35000
htcondor_walltime = 180000

# NNLOJET event count and integration steps for every channel (must be same length as channels)
warmup_events = 50000000 25000000 25000000 4000000 4000000 4000000 4000000
```

```
warmup_iterations = 10 10 10 10 10 10 13

# first seed for first branch, counting upwards
starting_seed = 0

[FastWarm]

# override tolerance to 5% failed jobs
# tolerance = 0.05
htcondor_requirements = (TARGET.ProvidesEkpResources==true)

# NNLOJET event count and number of jobs for each channel
fastwarm_events = 200000000 200000000 200000000 100000000 100000000 100000000 100000000
fastwarm_jobs = 10 10 10 10 10 10 10
starting_seeds = 1000 2000 3000 4000 5000 6000 7000

[MergeFastWarm]

[FastProd]

htcondor_request_memory = 15000

# NNLOJET event count and number of jobs for each channel
fastprod_events = 250000000 45000000 22000000 800000 700000 250000 800000
fastprod_jobs = 20 50 50 10 10 50 50
starting_seeds = 10000 20000 30000 40000 50000 60000 70000

[CopyTables]

[Combine]

# path to combine.ini config
combine_ini = /ceph/mcorrea/analysis/ZJtriple/combine.ini

# number of cores for NNLOJET combine script
cores = 20

[MergeFastProd]

htcondor_requirements = (TARGET.ProvidesEkpResources==true)

# execute workflow as local workflow instead of htcondor
```

```
   workflow (useful for merging small amount of grids, to be
   removed later)
workflow = local

[MergeFinal]

[FnloCppread]

[FnloCppreadFinal]

[SingleScalecheck]

[Approxtest]

fscl = 7

[Absolute]

[AbsoluteAll]
```

APPENDIX B

List of Figures

APPENDIX C

List of Tables

APPENDIX D

Bibliography

[1] Michael E. Peskin and Daniel V. Schroeder. “An Introduction to quantum field theory”. Reading, USA: Addison-Wesley, 1995. ISBN: 9780201503975, 0201503972.

[2] L. H. Ryder. “QUANTUM FIELD THEORY”. Cambridge University Press, 1996. ISBN: 9780521478144, 9781139632393, 9780521237642.

[3] Particle Data Group Collaboration. “Review of Particle Physics”. *Chin. Phys.* C40.10 (2016), p. 100001.
DOI: 10.1088/1674-1137/40/10/100001.

[4] Nathalie Palanque-Delabrouille et al. “Neutrino masses and cosmology with Lyman-alpha forest power spectrum”. *JCAP* 1511.11 (2015), p. 011.
DOI: 10.1088/1475-7516/2015/11/011. arXiv: 1506.05976 [astro-ph.CO].

[5] NNPDF Collaboration. “Parton distributions from high-precision collider data”. *Eur. Phys. J.* C77.10 (2017), p. 663.
DOI: 10.1140/epjc/s10052-017-5199-5. arXiv: 1706.00428 [hep-ph].

[6] ZEUS Collaboration. “Combination of measurements of inclusive deep inelastic $e^{\pm}p$ scattering cross sections and QCD analysis of HERA data”. *Eur. Phys. J.* C75.12 (2015), p. 580.
DOI: 10.1140/epjc/s10052-015-3710-4. arXiv: 1506.06042 [hep-ex].

[7] Sayipjamal Dulat et al. “New parton distribution functions from a global analysis of quantum chromodynamics”. *Phys. Rev.* D93.3 (2016), p. 033006.
DOI: 10.1103/PhysRevD.93.033006. arXiv: 1506.07443 [hep-ph].

[8] S. Alekhin, J. Blumlein, and S. Moch. “The ABM parton distributions tuned to LHC data”. *Phys. Rev.* D89.5 (2014), p. 054028.
DOI: 10.1103/PhysRevD.89.054028. arXiv: 1310.3059 [hep-ph].

[9] L. A. Harland-Lang et al. “Parton distributions in the LHC era: MMHT 2014 PDFs”. *Eur. Phys. J.* C75.5 (2015), p. 204.
DOI: 10.1140/epjc/s10052-015-3397-6. arXiv: 1412.3989 [hep-ph].

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

[11] John Collins. “Foundations of perturbative QCD”. *Camb. Monogr. Part. Phys. Nucl. Phys. Cosmol.* 32 (2011), pp. 1–624.

[12] V. N. Gribov and L. N. Lipatov. “Deep inelastic e p scattering in perturbation theory”. *Sov. J. Nucl. Phys.* 15 (1972). [Yad. Fiz.15,781(1972)], pp. 438–450.

[13] Guido Altarelli and G. Parisi. “Asymptotic Freedom in Parton Language”. *Nucl. Phys.* B126 (1977), pp. 298–318.
DOI: 10.1016/0550-3213(77)90384-4.

[14] 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). [Zh. Eksp. Teor. Fiz.73,1216(1977)], pp. 641–653.

[15] Lyndon Evans and Philip Bryant. “LHC Machine”. *JINST* 3 (2008), S08001.
DOI: 10.1088/1748-0221/3/08/S08001.

[16] ALICE Collaboration. “The ALICE experiment at the CERN LHC”. *JINST* 3 (2008), S08002.
DOI: 10.1088/1748-0221/3/08/S08002.

[17] ATLAS Collaboration. “The ATLAS Experiment at the CERN Large Hadron Collider”. *JINST* 3 (2008), S08003.
DOI: 10.1088/1748-0221/3/08/S08003.

[18] CMS Collaboration. “The CMS Experiment at the CERN LHC”. *JINST* 3 (2008), S08004.
DOI: 10.1088/1748-0221/3/08/S08004.

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

[20] Cinzia De Melis. *The CERN accelerator complex. Complexe des accélérateurs du CERN.* https://cds.cern.ch/record/2119882. [Online; accessed 12-June-2018]. 2016.

[21] Izaak Neutelings. *Simple example of 3D axes with spherical coordinates.* https://wiki.physik.uzh.ch/cms/latex:example_spherical_coordinates. [Online; accessed 3-July-2018]. 2017.

[22] David Barney. *CMS Detector Slice.* https://cds.cern.ch/record/2120661. [Online; accessed 12-June-2018]. 2016.

[23] CMS Collaboration. “Measurement of the triple-differential dijet cross section in proton-proton collisions at $\sqrt{s} = 8\,\text{TeV}$ and constraints on parton distribution functions”. *Eur. Phys. J.* C77.11 (2017), p. 746.
DOI: 10.1140/epjc/s10052-017-5286-7. arXiv: 1705.02628 [hep-ex].

[24] A. Banfi et al. “Optimisation of variables for studying dilepton transverse momentum distributions at hadron colliders”. *Eur. Phys. J.* C71 (2011), p. 1600.
DOI: 10.1140/epjc/s10052-011-1600-y. arXiv: 1009.1580 [hep-ex].

[25] T. Gleisberg et al. “Event generation with SHERPA 1.1”. *JHEP* 02 (2009), p. 007. DOI: 10.1088/1126-6708/2009/02/007. arXiv: 0811.4622 [hep-ph].

[26] C. F. Berger et al. “One-Loop Calculations with BlackHat”. *Nucl. Phys. Proc. Suppl.* 183 (2008), pp. 313–319. DOI: 10.1016/j.nuclphysbps.2008.09.123. arXiv: 0807.3705 [hep-ph].

[27] Fabio Cascioli, Philipp Maierhofer, and Stefano Pozzorini. “Scattering Amplitudes with Open Loops”. *Phys. Rev. Lett.* 108 (2012), p. 111601. DOI: 10.1103/PhysRevLett.108.111601. arXiv: 1111.5206 [hep-ph].

[28] Dominik Haitz. “Precision Studies of Proton Structure and Jet Energy Scale with the CMS Detector at the LHC”. PhD thesis. Karlsruhe Institute of Technology (KIT), 2016.

[29] Anna Friedel. “Measurement of the Z($\rightarrow \mu\mu$) + Jets Cross Section at $\sqrt{s} = 13$ TeV with the CMS Experiment and Studies of the Proton Structure”. Master’s thesis. Karlsruhe Institute of Technology (KIT), 2017.

[30] Bettina Schillinger. “Untersuchung des dreifach differentiellen Wirkungsquerschnitts der Z+Jet Produktion auf PDF-Sensitivität”. Bachelor’s thesis. Karlsruhe Institute of Technology (KIT), 2016.

[31] A. Gehrmann-De Ridder et al. “Precise QCD predictions for the production of a Z boson in association with a hadronic jet”. *Phys. Rev. Lett.* 117.2 (2016), p. 022001. DOI: 10.1103/PhysRevLett.117.022001. arXiv: 1507.02850 [hep-ph].

[32] Aude Gehrmann-De Ridder et al. “NNLO QCD corrections for Z boson plus jet production”. *PoS* RADCOR2015 (2016), p. 075. DOI: 10.22323/1.235.0075. arXiv: 1601.04569 [hep-ph].

[33] A. Gehrmann-De Ridder et al. “Z+jet production at NNLO”. *PoS* LL2016 (2016), p. 056. DOI: 10.22323/1.260.0056. arXiv: 1607.01749 [hep-ph].

[34] T. Kluge, K. Rabbertz, and M. Wobisch. “FastNLO: Fast pQCD calculations for PDF fits”. *Deep inelastic scattering. Proceedings, 14th International Workshop, DIS 2006, Tsukuba, Japan, April 20-24, 2006.* 2006, pp. 483–486. DOI: 10.1142/9789812706706_0110. arXiv: hep-ph/0609285 [hep-ph].

[35] fastNLO Collaboration. “Theory-Data Comparisons for Jet Measurements in Hadron-Induced Processes” (2011). arXiv: 1109.1310 [hep-ph].

[36] fastNLO Collaboration. “New features in version 2 of the fastNLO project”. *Proceedings, 20th International Workshop on Deep-Inelastic Scattering and Related Subjects (DIS 2012): Bonn, Germany, March 26-30, 2012.* 2012, pp. 217–221. DOI: 10.3204/DESY-PROC-2012-02/165. arXiv: 1208.3641 [hep-ph].

[37] Tancredi Carli, Gavin P. Salam, and Frank Siegert. “A Posteriori inclusion of PDFs in NLO QCD final-state calculations”. *HERA and the LHC: A Workshop on the Implications of HERA for LHC Physics (Startup Meeting, CERN, 26-27 March 2004; Working Group Meeting, CERN, 17-21 January 2005; Final Meeting 21-24 Mar 2005) CERN, Geneva, Switzerland, October 11-13, 2004*. 2005. arXiv: hep-ph/0510324 [hep-ph].

[38] Tancredi Carli et al. “A posteriori inclusion of parton density functions in NLO QCD final-state calculations at hadron colliders: The APPLGRID Project”. *Eur. Phys. J.* C66 (2010), pp. 503–524. DOI: 10.1140/epjc/s10052-010-1255-0. arXiv: 0911.2985 [hep-ph].

[39] Daniel Andreas Britzger. “Regularized Unfolding of Jet Cross Sections in Deep-Inelastic ep Scattering at HERA and Determination of the Strong Coupling Constant”. PhD thesis. University of Hamburg, 2013.

[40] Spotify. *Luigi*. https://github.com/spotify/luigi. [Online; accessed 25-October-2018]. 2012.

[41] M. Erdmann et al. “Design and Execution of make-like, distributed Analyses based on Spotify’s Pipelining Package Luigi”. *J. Phys. Conf. Ser.* 898.7 (2017), p. 072047. DOI: 10.1088/1742-6596/898/7/072047. arXiv: 1706.00955 [physics.data-an].

[42] Douglas Thain, Todd Tannenbaum, and Miron Livny. “Distributed computing in practice: the Condor experience”. *Concurrency and Computation: Practice and Experience* 17.2-4 (2005), pp. 323–356. DOI: 10.1002/cpe.938.

[43] Benjamin Peterson. *Six*. https://github.com/benjaminp/six. [Online; accessed 11-November-2018].

[44] CERN. *GFAL2*. https://dmc.web.cern.ch/projects/gfal-2/home. [Online; accessed 25-October-2018]. 2014.

[45] Jordan Walke. *React*. https://reactjs.org/. [Online; accessed 25-October-2018]. 2013.

[46] Ryan Dahl. *Node.js*. https://nodejs.org. [Online; accessed 25-October-2018]. 2009.

[47] bwHPC initiative. *bwForCluster NEMO*. https://www.hpc.uni-freiburg.de/nemo. [Online; accessed 8-November-2018]. 2013.

[48] *Helix Nebula Science Cloud*. https://www.hnscicloud.eu/. [Online; accessed 8-November-2018].

[49] Karlsruhe Institute of Technology (KIT). *Grid Computing Centre Karlsruhe (GridKa)*. www.gridka.de. [Online; accessed 8-November-2018].

Acknowledgements

I dedicate this chapter to the people who contributed to this thesis. Thank You.

- Prof. Günter Quast for giving me the opportunity to work with him and his team.
- Dr. Klaus Rabbertz who came up with the idea for this thesis and also acted as my personal mentor throughout the entire thesis. This work would not have been possible without his help.
- Thomas Berger for the close collaboration on the Z+jet analysis.
- Bettina Schillinger for her great work on plotting scripts that were used in this thesis.
- The generous computing resources supplied by the Institute for Experimental Particle Physics (ETP) at the KIT, the Helix Nebula Science Cloud, the Grid Computing Centre Karlsruhe (GridKa) and the BwForCluster NEMO through bwHPC and the German Research Foundation (DFG).
- The friendly computing experts Florian von Cube, Christoph Heidecker and Matthias Schnepf who helped me with all of my technical problems.
- Daniel Savoiu for proof reading my thesis.
- Last but not least, a thank you to my parents Carlos and Ivonne for their continued support throughout my entire studies in physics.

Erklärung der selbständigen Anfertigung der Masterarbeit

Hiermit erkläre ich, dass ich die Masterarbeit mit dem Titel

»*Automated Production of Interpolation Grids*
at NNLO for the Triple-Differential Z+Jet
Cross Section Measurement at the LHC«

selbständig und unter ausschließlicher Verwendung der angegebenen Hilfsmittel angefertigt habe.

Miguel Santos Correa
Karlsruhe, den 13. November 2018