Machine Learning for a Model Unspecific Search in CMS

Masterarbeit in Physik

von

B. Sc. Ana Andrade

7th April 2023

Diese Arbeit wurde vorgelegt am

Fakultät der Mathematik, Informatik und Naturwissenschaften
der Rheinisch-Westfälischen Technischen Hochschule Aachen

angefertigt im

III. Physikalischen Institut A

Gutachter – Prof. Dr. rer. nat. Thomas Hebbeker
Zweitgutachter – Prof. Dr. rer. nat. Martin Erdmann

I dedicate this thesis to my parents,

Maria Madalena Barbosa Alves Andrade
&
António Agostinho Andrade.

I love you to the Moon and back.

Dedico esta tese aos meus pais,

Maria Madalena Barbosa Alves Andrade
&
António Agostinho Andrade.

Amo-vos daqui até à Lua.

Declaration of Authorship

I, B. Sc. Ana Andrade, declare that this thesis, titled *Machine Learning for a Model Unspecific Search in CMS*, and the work presented in it are my own. I hereby confirm that:

- This work was done wholly or mainly while in candidature for a Master of Science degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. Except for such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Aachen, 7th of April 2023

Signed: _____

(Ana Andrade)

Abstract

The MUSiC algorithm is a model unspecific search for new physics phenomena at the CMS experiment. It serves as a complementary analysis strategy to dedicated searches looking for physics beyond the Standard Model. So far, MUSiC analyses have not found deviations from Standard Model expectations. In this work, it is proposed that the MUSiC algorithm be modified to incorporate a newly published machine learning algorithm known as *New Physics Learning Machine* (NPLM). This algorithm is first tested on a case study containing data sets with different event weight distributions and varying percentages of negatively weighted events. NPLM is then employed for the $1e + 1\mu + p_T^{miss}$ MUSiC event class for 2017 data, at a centre of mass energy $\sqrt{s} = 13$ TeV. Three alternative hypothesis are introduced to test NPLM sensitivity, namely, the variation of the expected cross section of $t\bar{t}$ processes with $M_{t\bar{t}} > 700$ GeV, of Higgs processes and, lastly, of Multi-Boson processes. This work provided the first ever implementation of any machine learning approach to MUSiC analyses. It was concluded that NPLM is a promising candidate to such approach.

Zusammenfassung

Der MUSiC-Algorithmus ist eine modellunspezifische Suche nach Phänomenen der neuen Physik am CMS-Experiment. Er dient als ergänzende Analysestrategie zu speziellen Suchen, die nach Physik jenseits des Standardmodells suchen. Bisher haben die MUSiC-Analysen keine Abweichungen von den Erwartungen des Standardmodells gefunden. In dieser Arbeit wird vorgeschlagen, den MUSiC-Algorithmus so zu modifizieren, dass er einen neu veröffentlichten Algorithmus für maschinelles Lernen, bekannt als *New Physics Learning Machine* (NPLM), anwendet. Dieser Algorithmus wird zunächst an einer Fallstudie getestet, die Datensätze mit unterschiedlichen Ereignisgewichtsverteilungen und unterschiedlichen Anteilen negativ gewichteter Ereignisse enthält. NPLM wird dann für die MUSiC-Ereignisklasse $1e + 1\mu + p_T^{miss}$ für Daten aus dem Jahr 2017 bei einer Schwerpunktsenergie $\sqrt{s} = 13$ TeV eingesetzt. Drei alternative Hypothesen werden eingeführt, um die NPLM-Empfindlichkeit zu testen, namentlich die Variation des erwarteten Wirkungsquerschnitts von $t\bar{t}$-Prozessen mit $M_{t\bar{t}} > 700$ GeV, von Higgs-Prozessen und schließlich von Multi-Bosonen-Prozessen. Diese Arbeit war die erste Implementierung eines maschinellen Lernansatzes für MUSiC-Analysen überhaupt. Es wurde festgestellt, dass NPLM ein vielversprechender Kandidat für einen solchen Ansatz ist.

Acknowledgements

I want to first thank Prof. Thomas Hebbeker for welcoming me into the III A. Institute, allowing me to carry out my thesis work in such a positive and friendly research group. I would also like to thank Dr. Arnd Meyer for his guidance. Moreover, I express my gratitude to Lorenzo Vigilante, who thought me the inner workings of the MUSiC code, provided many fruitful discussions, and modified the MUSiC algorithm to accommodate a machine learning approach, allowing the development of the work in this thesis. I also thank the MUSiC team, Dr. Felipe Torres da Silva de Araújo and Yannik Kaiser, for their continued support, interesting discussions, and for providing such a pleasant working environment during the last months. I also thank Felipe for proofreading this work and providing many helpful comments. Moreover, I wish to thank Gaia Grosso, from the University of Padova, for guiding me through the NPLM implementation, which undoubtedly increased the quality of the work in this thesis. I also extend my gratitude to Francesco Ivone, who gave me my first ever tour of the CERN facilities, to Fabian Nowotny, for his naivety in falling for my lipstick related pranks, and to Valentina Sarkisovi, for her wonderful friendship and exam related advice. Lastly, I want to give a warm thank you to Sebastian Wiedenbeck, who provided me with unconditional emotional support, making my thesis writing process an easier and more enjoyable one. I also apologise to his brain cells for the awful dad jokes they had to endure.

Finally, I thank my parents, to whom this thesis is dedicated to, for always investing in my education and supporting me emotionally and financially. They are the only people who have seen the troubled journey that led to the completion of this thesis and without them I would not be writing these words today. Although their own education journey was a short one, stopping at primary school, they have always encouraged me to pursue a career in high energy physics. They have believed in me in the best and worst of times, giving me the strength to pursue my dream of becoming a particle physicist. I will spend the rest of my days trying to repay them, by answering their curiosity-driven questions regarding the turbulence of sea waves or the brightness of the Sun.

As a final acknowledgement, I would like to thank the nameless individuals who have fought for women's rights and women's education. Science is for everyone and I am grateful to have been born at a time which has not deprived me of the wonders of studying physics.

Contents

Chapter 1

The Standard Model

"My [algebraic] methods are really methods of working and thinking; this is why they have crept in everywhere anonymously."
– Emmy Noether

Fundamental physics searches for a complete understanding of the matter content in the universe, alongside with its associated interactions. To the point of writing, this search has culminated in the development of the Standard Model of Particle Physics (SM), describing all known particles and three of the four known forces, i.e. electromagnetism and the weak and strong nuclear forces (section 1.1). Gravity has not yet been successfully incorporated into the SM.

The SM is one of many possible Quantum Field Theories (QFTs). The primary motivation to use field theory is to describe *local* phenomena, i.e. to avoid that interactions happen instantaneously, which is not physical. Classically, General Relativity and Maxwell's Equations provide a field theory description for gravity and electromagnetism, respectively. It would be equally desirable to describe particle interactions *locally*. However, from Heisenberg's uncertainty principle, one can calculate the Compton wavelength, representing the scale at which the appearance of particle/anti-particle pairs becomes possible and the one-particle Schrödinger equation breaks down. Thus, a new mathematical formalism describing particle interactions is required. In particle physics, an equally important motivation for field theory lies in the identicalness of particles. A photon arising from a distant start, or one arising from a light bulb in a living room, share the same proprieties and are indeed indistinguishable without apparent reason. In a QFT framework, particles are seen as excitations of their respective dynamical fields, solving both the locality and identicalness problems.

To build a QFT, it is first necessary to postulate the symmetries of the system. To be compatible with Special Relativity, all QFTs must have global Poincaré symmetry. A QFT is then identified by its internal symmetry group, or gauge group. In the case of the SM, that group is $U(1)_Y \times SU(2)_L \times SU(3)_C$. Together, $U(1)_Y \times SU(2)_L$ describe Electroweak Theory (section 1.3), where Y stands for *weak hypercharge* and *left-handed fermions*. During symmetry breaking, particles in the SM acquire mass through the Higgs mechanism (section 1.4), and the electroweak force splits into the electromagnetic force, described by Quantum Electrodynamics (QED) (section 1.2), and the weak force. Finally, $SU(3)_C$, where C stands for *colour charge*, relates to the strong nuclear force, described by Quantum Chromodynamics (QCD) (section 1.5). The strong and electroweak forces have not yet been unified.

1.1 Particle Content

How particles behave under transformation groups will depend on their intrinsic spin, which may be an integer or half-integer value. Particles with spin 0 are denoted as *scalar* objects, spin $1/2$ as *spinors* and spin 1 as *vector* objects [1]. Spinors correspond to matter particles, also called *fermions*, while scalars and vectors correspond to force carriers, also called *bosons*.

Currently, the SM comprises of twelve fermions, four spin 1 *vector* or *gauge bosons* and one spin 0 *boson*. A summary table is found in Fig. 1.1. The fermions are split into *quarks* and *leptons*. Electrically charged fermions can interact electromagnetically, through the exchange of *photons*, and weakly. Weak interactions may happen through neutral currents, exchanging Z *bosons*, or through charged currents, exchanging $W^{\pm}$ *bosons*. Neutrinos are electrically neutral and only take part in weak interactions. Neutrinos are also taken to be massless in the SM, although this is known to not be the case. Note that $W^{\pm}$ *bosons* are also electrically charged and, consequently, interact with themselves. Quarks are the only fermions carrying *colour charge* and, consequently, the only ones interacting with carrier particles of the strong nuclear force, the *gluons*. Moreover, the gluons are the only bosons carrying colour charge and so, they also interact with themselves. Finally, all observed massive particles interact with the *Higgs boson*, which will be discussed later.

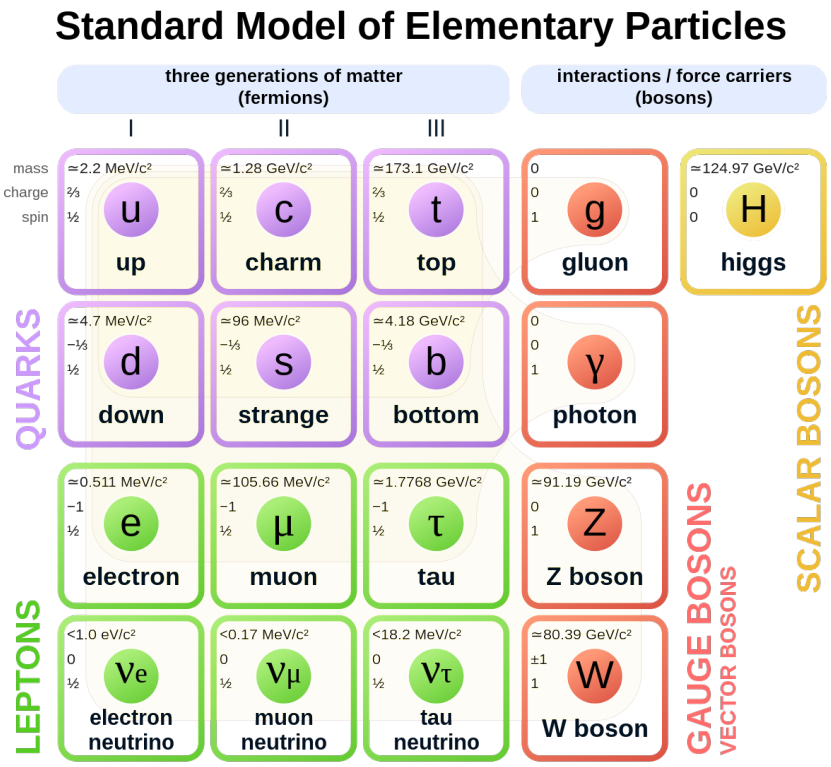


Figure 1.1: The Standard Model [1].

The dynamics of a single fermion or boson can be described by generalizing the Schrödinger equation. Let $\phi(x)$, $\psi(x)$ and $A^{\mu}(x)$ represent a scalar, spinor and vector field, where x is a

[1] There are also hypothetical spin 2 particles, like the graviton, which are *tensor* objects.

spacetime point. The fields then obey the Klein-Gordon (Eq. 1.1), Dirac (Eq. 1.2) and Proca (Eq. 1.3) equations, respectively,

$$(\partial^\mu \partial_\mu + m^2)\phi(x) = 0 \tag{1.1}$$

$$(i\boldsymbol{\gamma}^\mu \partial_\mu - m)\psi(x) = 0 \tag{1.2}$$

$$\partial_\mu(\partial^\mu A^\nu(x) - \partial^\nu A^\mu(x)) + m^2 A^\nu(x) = 0. \tag{1.3}$$

Note that m stands for the particle mass associated with each field; $\boldsymbol{\gamma}^\mu$ are the *Dirac matrices*, with anti-commutation relations $\{\boldsymbol{\gamma}^\mu, \boldsymbol{\gamma}^\nu\} = 2\boldsymbol{\eta}^{\mu\nu}$, where $\boldsymbol{\eta}^{\mu\nu}$ is the Minkowski metric tensor; and ∂_μ stands for the partial derivative over the four spacetime coordinates. Moreover, the Klein-Gordon equation results in two possible solutions, which are interpreted as particle and anti-particle fields. An anti-particle shares the same m as its partner particle, but all other associated quantum numbers are inverted. The Dirac equation also presents two possible solutions. This is seen more clearly when one assumes a plane wave solution to the derived relativistic energy momentum relation $E^2 = \boldsymbol{p}^2 + m^2$. As a final remark, note that for a massless vector field following the Proca equation, one recovers Maxwell's equations in a vacuum.

1.2 Quantum Electrodynamics

As previously stated at the beginning of the chapter, it would be desirable to describe quantum interactions *locally*. Focusing on a free fermion, e.g. the electron, one can write the Lagrangian density $\mathcal{L}$ from the Dirac equation (Eq. 1.2) as

$$\mathcal{L} = \bar{\psi}(x)(i\partial\!\!\!/ - m)\psi(x) \tag{1.4}$$

where the slashed notation was introduced, i.e. $\partial\!\!\!/ \equiv \gamma^\mu \partial_\mu$. Consider now an *abelian* unitary gauge group U of dimension d. When a given transformation $U(\theta^i)$, parametrized by θ^i, $i = (1, 2, ..., d)$, is applied to the field $\psi(x)$, it may be applied globally or locally. More precisely, the complex phase shift

$$\psi(x) \longrightarrow \psi'(x) = U(\theta^i)\psi(x) = e^{iq\theta^i \boldsymbol{T}^i}\psi(x), \tag{1.5}$$

may be the same everywhere in the field, or may vary for each spacetime point, i.e. $\theta^i \to \theta^i(x)$. Note that q denotes the coupling constant (the elementary electric charge) and T^i are complex hermitian matrices representing the corresponding group generators. Since elements in U commute with each other, the commutation relations $[\boldsymbol{T}^i, \boldsymbol{T}^j] = i\, f_{ijk}\, \boldsymbol{T}^k$ vanish, as all structure constants become 0. Indeed, for two successive field transformations one has that

$$e^{iq\chi(x)} e^{iq\chi'(x)}\psi(x) = e^{iq(\chi(x)+\chi'(x))}\psi(x) \tag{1.6}$$

where $\chi(x) \equiv \theta^i(x)\boldsymbol{T}^i(x)$. and similarly for $chi'(x)$. . The SM group $U(1)_Y$ is then represented by a group of 1×1 abelian matrices, i.e. the complex numbers.

Upon applying a local gauge transformation, it is important to note that the Lagrangian density (Eq. 1.4) is no longer gauge invariant. A neat trick to recover such invariance is to

introduce a vector field $A^\mu(x)$, whilst generalizing the definition of *derivative* ∂_μ to *covariant derivative* $D_\mu(x)$, where

$$A^\mu(x) \longrightarrow A'^\mu(x) = U(\theta^i)A^\mu(x) = A^\mu(x) + \frac{1}{q}\partial_\mu\theta^i \tag{1.7}$$

$$D_\mu(x) \equiv \partial_\mu + iqA_\mu(x). \tag{1.8}$$

It is the second term in the convariant derivative $D_\mu(x)$ which introduces the interaction between the vector and spinor fields, with coupling q. Using the Proca equation (Eq. 1.3), one can introduce a kinetic term for the free propagation of the gauge field $A^\mu(x)$. The Lagrangian density is then modified to

$$\mathcal{L} = \bar{\psi}(i\not{D} - m)\psi + \frac{1}{4}F^{\mu\nu}F_{\mu\nu} + \frac{1}{2}m_A^2A^\mu A_\mu \tag{1.9}$$

where $F^{\mu\nu} \equiv \partial^\mu A^\nu - \partial^\nu A^\mu$ is the electromagnetic field strength tensor. Since only the vector mass term breaks local gauge invariance, one postulates that the vector field $A^\mu(x)$ must be massless. Indeed, this is the observed massless photon field, which interacts with electrically charge particles! Since the photon field is itself electrically neutral, self-coupling does not occur. Considering electron - photon interactions with coupling $e = -1 \cdot q$, the QED Lagrangian density reads

$$\mathcal{L}_{QED} = \bar{\psi}(i\not{D} - m)\psi + \frac{1}{4}F^{\mu\nu}F_{\mu\nu} \tag{1.10}$$

It is remarkable that purely from mathematical arguments, i.e. the requirement that the Lagrangian density be locally invariant to the introduction of a complex phase in the spinor field, one finds a justification for the existence of photons! As a final remark, note that one may use Noether's Theorem to show that a consequence of this particular Lagrangian symmetry is the conservation of electric charge.

1.3 Electroweak Theory

In the previous section, the discussion focused on the electromagnetic force, associated with $U(1)$ transformations. In this section, the discussion is turned to the weak force. In 1956, Chien-Shiung Wu showed experimentally that the weak force, unlike electromagnetism, differentiates between particle chiralities [2]. Only left-handed fermions, or right-handed anti-fermions, participate in weak interactions. Thus, the weak force is associated with transformations from the *non-abelian* group $SU(2)_L$, capable of making such distinction. The left-handed fermions are organized into doublets, or isospinors, $\psi_L^{1/2}$, where each component is a regular Dirac spinor. The right-handed anti-fermions are described as singlets $\bar{\psi}_R$. Analogously to Eq. 1.5, an isospinor transforming under $SU(2)_L$ can be written as

$$\psi^{1/2}(x) \longrightarrow \psi'^{1/2}(x) = SU(\alpha^a)\psi^{1/2}(x) = e^{\frac{iI}{2}\alpha^a\boldsymbol{\tau}^a}\psi^{1/2}(x), \tag{1.11}$$

where $\alpha^a = (\alpha^1, \alpha^2, \alpha^3)$ are real parameters and $\boldsymbol{\tau}^a = (\boldsymbol{\tau}^1, \boldsymbol{\tau}^2, \boldsymbol{\tau}^3)$ are the *Pauli matrices*, i.e. the generators of the group. In this case, the coupling charge I is called *weak isospin*. Redefining $\frac{\boldsymbol{\tau}}{2} \equiv \boldsymbol{T}^{1/2}$, the angular momentum relations from quantum mechanics become apparent

$$[\boldsymbol{T}_i^{1/2}, \boldsymbol{T}_j^{1/2}] = i\epsilon_{ijk}\boldsymbol{T}_k^{1/2} \tag{1.12}$$

by identifying $\boldsymbol{T}_i^{1/2}$ with the angular momentum operators $\boldsymbol{J}_i$.

To describe weak interactions locally, the transformation parameter α^a should become dependent on space-time position, $\alpha^a(x)$, as it was done with θ^i in the previous section. Since there are now three $SU(2)_L$ group generators, as opposed to just one in the $U(1)$ case, the analogous covariant derivative to equation 1.8 will now introduce three new vector fields

$$D_\mu(x)^a \equiv \partial_\mu + \frac{iI}{2}\boldsymbol{\tau^a} \cdot \boldsymbol{W}_\mu^a(x) \tag{1.13}$$

where $\boldsymbol{W}_\mu^a = (\boldsymbol{W}_\mu^1, \boldsymbol{W}_\mu^2, \boldsymbol{W}_\mu^3)$ will correspond to three new gauge bosons. Before attempting to write a Lagrangian density, it is important to note that due to the non-commutativity of the group, the field strength tensor must now include the commutation relations, thus becoming

$$W_{\mu\nu}^a = \partial_\mu W_\nu^a - \partial_\nu W_\mu^a + iI\left[W_\mu^a, W_\nu^a\right]. \tag{1.14}$$

The Lagrangian density now explicitly reflects the distinction in chiralities, with the covariant derivative in Eq. 1.13 only being applied to the left-handed fermionic field,

$$\mathcal{L}_{SU(2)_L} = i\bar{\psi}_L\gamma^\mu D_\mu\psi_L + i\bar{\psi}_R\gamma^\mu\partial_\mu\psi_R - \frac{1}{2}\text{Tr}\left(W_{\mu\nu}^a W^{a\mu\nu}\right). \tag{1.15}$$

1.4 Higgs Mechanism

The introduction of massive gauge bosons in Electroweak theory implies a spontaneous gauge symmetry breaking. The mechanism to explain this symmetry breaking was independently proposed, in 1964, by Peter Higgs [3] and François Englert et. al. [4]. The idea consists in introducing a two-component complex scalar field ϕ, known as the Higgs field, introducing four new degrees of freedom. This results in a non-vanishing ground state of the system. The introduced degrees of freedom give mass to the three electroweak gauge bosons, $W^\pm$ and Z^0, and to an additional massive spin-0 particle H^0, known as the Higgs boson. The Higgs boson was experimentally discovered in 2012 [5] [6], confirming the theoretical prediction.

1.5 Quantum Chromodynamics

The introduction of QCD was motivated by the discovery of the Δ^{++} resonance, since all three up quarks in the composite particle have the same spin $s = +\frac{1}{2}$ [7]. This property apparently violates the Pauli exclusion principle and so, a new quantum number needed to be introduced. This new variable is now known as *colour charge* and only applies to elementary particles interacting via the strong nuclear force. Colour charge can take the values of *red*, *green* or *blue* and thus, the strong nuclear force is associated to transformations from the group $SU(3)$, with three complex dimensions. Unlike QED where there was only one group generator, the $SU(3)$ is associated to 8 generators, since there are 8 independent directions in the matrix space. These generators represent the transformations carried out by the *gluons* in colour space, and so, there are indeed 8 gluons in the SM. The Lagrangian of QCD can then be constructed as

$$\mathcal{L}_{SU(3)} = \bar{\psi}_q^i(\gamma^\mu D_\mu)_{ij}\psi_q^j + m_q\bar{\psi}_q^i\psi_{qi} - \frac{1}{4}G_{\mu\nu}^a G^{a\mu\nu}. \tag{1.16}$$

where ψ_q^i represents a quark field with colour i, $\psi_q = (\psi_q^{red}, \psi_q^{green}, \psi_q^{blue})^T$, m_q introduces a non-zero quark mass via the Higgs mechanism and $G_{\mu\nu}^a$ is strength tensor of the gluon field, with an adjoint colour index $a \in \{1, 2, 3, 4, 5, 6, 7, 8\}$, given by

$$G_{\mu\nu}^a = \partial_\mu G_\nu^a - \partial_\nu G_\mu^a - g_s f_{bc}^a G_\mu^b G_\nu^c \tag{1.17}$$

where g_s stands for the strong coupling constant and f^{abc} are the structure constants of $SU(3)$. Lastly, the covariant derivative D_μ is given by

$$(D_\mu)_{ij} = \delta_{ij}\partial_\mu - ig_s t_{ij}^a G_\mu^a \tag{1.18}$$

with t_{ij}^a representing a matrix proportional to the Gell-Mann matrices λ_{ij}^a, which are the higher dimensional analog to the Pauli matrices. The usual convention is to take $t_{ij}^a = \frac{1}{2}\lambda_{ij}^a$.

Chapter 2

CMS Experiment at the LHC

"Lots of data gets collected through the latest technology today,
and not all of it is about people's consumer preferences."
– Lisa Randall

2.1 The Large Hadron Collider

The Large Hadron Collider (LHC) was established to deliver high energy collision data, allowing for the study of particle physics at the TeV scale. It is the largest, and most energetic, particle accelerator in the world, designed to deliver proton-proton collisions up to a centre-of-mass energy of $\sqrt{s} = 14$ TeV, at a luminosity of 10^{34} cm^{-2} s^{-1}. The LHC can also collide heavy ions, however that is outside of the scope of the present work. It is located at the European Organization for Nuclear Research (CERN), at the border between France and Switzerland, near Geneva. The LHC was built inside the 26.7 km circular tunnel previously hosting the Large Electron-Positron Collider (LEP), which finished operations in 2000. It is located at a mean depth of 100 meters underground. Superconducting electromagnets, cooled to 1.9 K through a liquid-helium system, keep the beams in a circular trajectory. The two proton beams are then made to collide at four different points, each one hosting a different experiment, namely:

- **A Toroidal LHC Apparatus (ATLAS)** – Multi-purpose experiment.
- **Compact Muon Solenoid (CMS)** – Multi-purpose experiment (section 2.2).
- **A Large Ion Collider Experiment (ALICE)** – Heavy ion collisions.
- **The Large Hadron Collider beauty Experiment (LHCb)** – Specializes in *b*-physics and matter-antimatter asymmetry.

To achieve the high energy particle beams, the LHC relies on a pre-acceleration system. The protons are sourced from hydrogen atoms, which have been stripped of their electrons. They are accelerated to 160 MeV at the Linear Accelerator 4 (LINAC4), which then injects them into the Proton Synchrotron Booster (PSB). In the PSB, the protons are accelerated up to 2 GeV and are subsequently injected into the Proton Synchrotron (PS). The PS further accelerates the protons to 25 GeV. Finally, the PS injects the beam into the Super Proton Synchrotron (SPS), which in turn increases their energy to 450 GeV. This is the required injection energy for the LHC, where the protons will go through a last acceleration phase, bringing their energy to a theoretical maximum of 7 TeV. The CERN accelerator complex is schematically shown in Fig. 2.1.

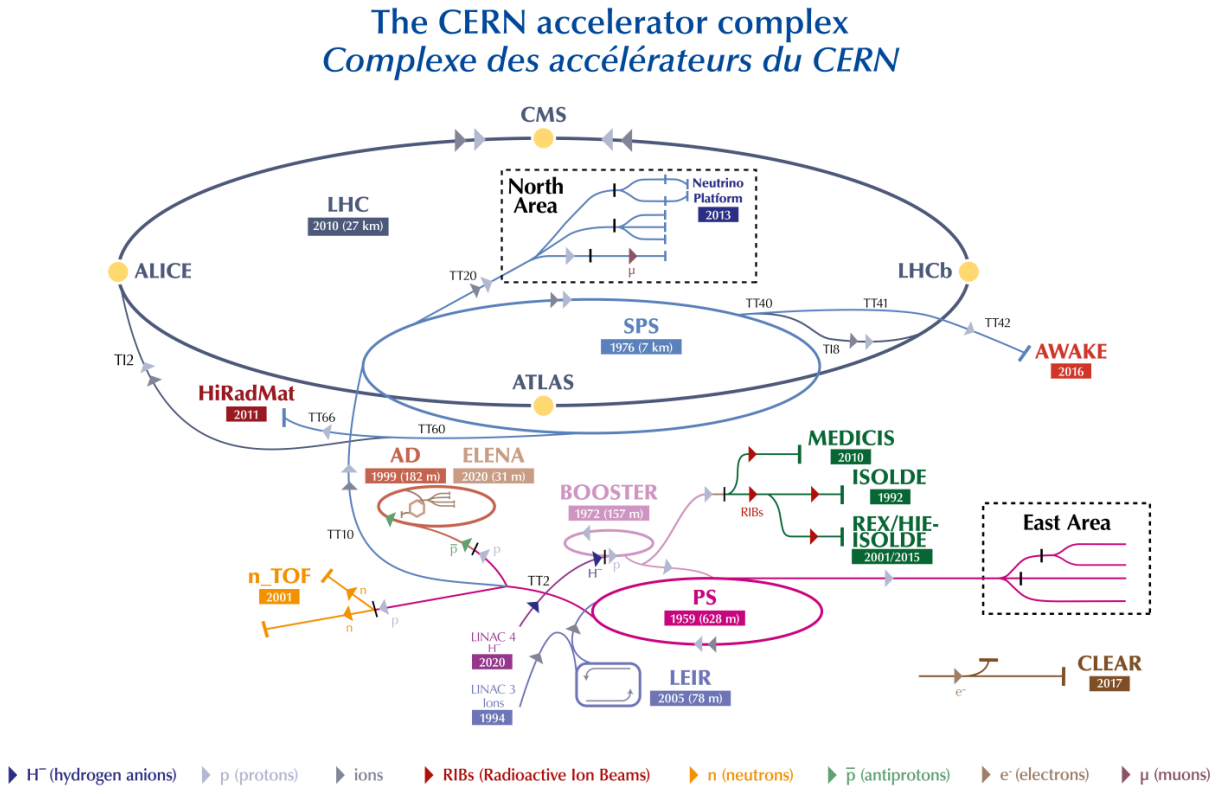


Figure 2.1: The CERN accelerator complex [8].

The LHC became operational in 2008, with the first data taking period, Run 1, starting in 2011, at centre-of-mass energies of $7-8$ TeV. After an upgrade period where the beam is turned off, known as a long shutdown (LS) period, the Run 2 data taking period started in 2015, at 13 TeV. At the point of writing, Run 3 is currently underway, having started at the end of 2022 at a centre-of-mass energy of 13.6 TeV.

The centre-of-mass energy when the beams collide is not enough to quantize the number of collisions in a given time period dt. For that, the cross-section σ of the process must be taken into account, and multiplied by the integrated luminosity $\mathcal{L}_{int}$ (Eq. 2.1),

$$N = \sigma \, \mathcal{L}_{int} = \sigma \int \mathcal{L} \, \mathrm{d}t. \tag{2.1}$$

The cross-section is a physical constant, predicted by the SM, representing the probability that a given process occurs. The luminosity, on the other hand, depends purely on experimental parameters and is calculated as

$$\mathcal{L} = \frac{N_1 N_2 N_b f}{4\pi\sigma_x\sigma_y} F, \tag{2.2}$$

where N_1, N_2 and N_b are the number of particles in each bunch, and the total number of bunches, respectively; f is the revolution frequency; σ_x and σ_y are the distribution widths of the particles in the $x-y$ plane; and F is a correction factor to account for the crossing angle when the bunches collide.

2.2 The Compact Muon Solenoid Experiment

Located near the French village Cessy, the CMS experiment hunts for new physics in proton-proton, or heavy ion, collision data. Although both the CMS and ATLAS collaborations share the same purpose, the two experiments have employed different detector designs. In the present chapter, the CMS design is briefly presented, from the inner- to the outermost subsystems. A schematic of the CMS detector is shown in Fig. 2.2. A detailed description of the full experimental apparatus can be found in [9].

Coordinate System – Both Cartesian and cylindrical coordinate systems are used at CMS. The origin is set at the interaction point (IP), the x axis points towards the centre of the LHC, the y axis towards the surface and the z axis towards the counterclockwise beam direction. The azimuthal angle ϕ is defined from the x to the y axis, while the polar angle θ is defined from the z to the y axis. In hadron colliders, the polar angle commonly is expressed through the *pseudo-rapidity*[1] η defined in Eq. 2.3 . The coordinate system, as well as the relationship between θ and η, are shown in Fig. 2.3.

$$\eta = -\ln\left[\tan\left(\frac{\theta}{2}\right)\right] \tag{2.3}$$

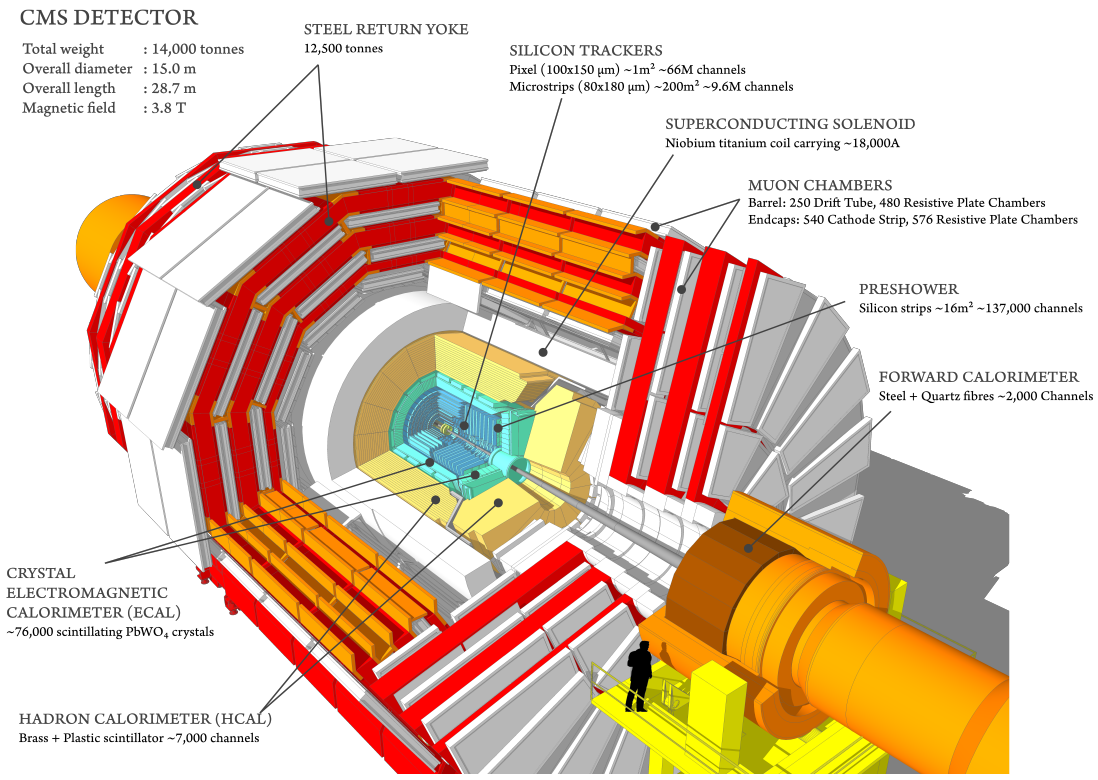


Figure 2.2: Schematic of the CMS detector components [10].

[1]Pseudo-rapidity is a useful parameter because it allows the definition of the Lorentz invariant distance (in the relativistic regime) $\Delta R \equiv \sqrt{\Delta\eta^2 + \Delta\phi^2}$.

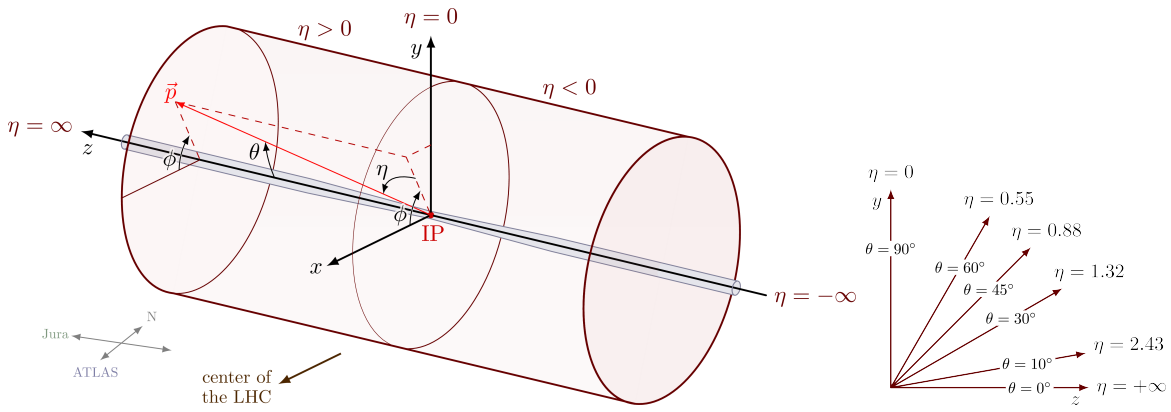


Figure 2.3: (left) CMS coordinate system showing the interaction point (IP); the Cartesian x axis, pointing to the centre of the LHC, the y axis pointing towards the surface and the z axis pointing in the counter-clockwise beam direction; the polar angle θ, defined from the $z$to the y axis, and the azimuthal angle ϕ, defined from the x to the y axis. The values of pseudo-rapidity η associated with each axis and coordinate region are also shown. (right) Polar angle and pseudo-rapidity equivalence [11].

2.2.1 Silicon Trackers

When a collision happens, the product particles start their outward journey through the CMS detector components. After moving through the ultra-high vacuum inside the beam pipe, and exiting the pipe itself, the first component that the particles encounter is the silicon tracking system. Besides identifying interaction and decay vertices, the tracker measures the momentum of electrons, muons and hadrons. This is done by measuring the position of the particles with a precision $\pm 10 \mu$m. The more the particles bend in the magnetic field produced by the solenoid magnet (see 2.2.4.), the higher their momentum. This allows to discriminate between particles. To measure the high particle flux close to the interaction point, the tracking system must be spatially compact but still present a good enough spatial resolution. These requirements imply that a highly granular structure is necessary. Silicon detectors were chosen for their granular module feasibility, but also for their fast response. Moreover, to protect the materials from radiation damage, the tracker is operated at -20C. The tracking system can measure the properties of particles within a pseudo-rapidity range of $|\eta| < 2.5$, achieving a 90% and 98% efficiency for electrons and muons, respectively. For hadrons, the efficiency ranges between $85 - 95\%$, depending on the energy. The tracker is composed of the following concentric subsystems:

- **PIXEL** – The pixel detector is the innermost subsystem of the tracker. It contains four 2D–layers of silicon sensors (the *pixels*) at approximately 3, 7, 11 and 16 cm from the interaction point [2] [12]. This layout allows for 3D track reconstruction by taking several measurements of the particles' position. Each pixel measures 150×100 μm^2 and every time a charged particle passes through it, enough energy is deposited to eject an electron from its silicon atoms. These electrons are collected through an applied voltage and converted into an electric signal. A readout chip is located beneath each pixel, which then amplifies this electric signal.

[2]The particle flux at a 3 cm radius is about 6×10^8 particles per cm^{2}s^{-1}.

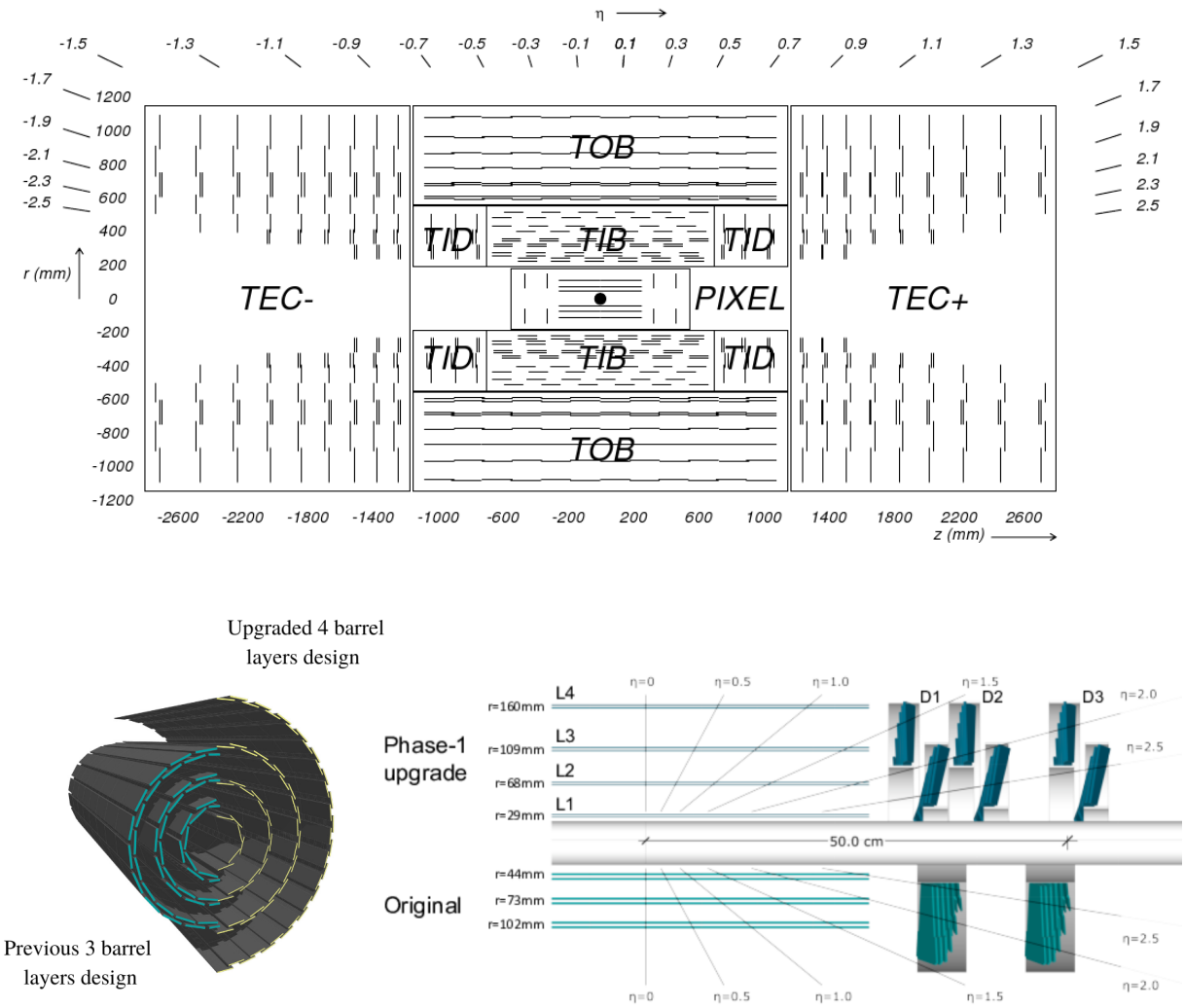


Figure 2.4: (top) The CMS Silicon Tracking system [9] showing, in the centre, the collision point as a black dot; the PIXEL subsystem (PIXEL), which is enclosed by the Tracker Inner Barrel (TIB) and the Tracker Inner Disks (TID); the whole subsystems are then enclosed in by the Tracker Outer Barrel (TOB) and the Tracker EndCaps (TEC $\pm$). The y axis shows the distance r from the collision point in millimeters; the z axis (at the bottom) shows the longitudinal distance z from the collision point in millimeters; and the upper part of the figure shows the corresponding pseudo-rapidity. (bottom) Upgraded Pixel detector subsystem, highlighting the design differences between the previous three layer design of silicon sensors to the current four layer design [12].

- **TIB** & **TID** (*Tracker Inner Barrel & Tracker Inner Disk*) – At radii larger than 20 cm, the particle flux decreases and silicon micro-strips can be used instead of the silicon pixels. The TIB is composed of four concentric layers of silicon micro-strips, with a width between $80 - 180$ μm. The micro-strips work similarly to the pixels, where excited electrons are captured and the electrical signal is amplified by readout chips. There are two TID, one at each end of the TIB. They are composed of three small disks each, allowing to reconstruct particle tracks with higher pseudo-rapidity.

- **TOB** & **TEC**$\pm$ (*Tracker Outer Barrel & Tracker EndCaps*) – The TOB surrounds the TIB and both TIDs, and is composed of six concentric layers of silicon micro-strips. The TEDs are located at each end of TOB, closing the tracker system. Similarly to the TIDs, the TECs allows for the reconstruction of tracks with higher pseudo-rapidity.

A schematic of the Silicon Tracker systems is shown in Fig. 2.4.

2.2.2 Electromagnetic Calorimeter

Besides momentum, total energy of particles must also be measured. Depending on the nature of the particle, its energy will be deposited in different calorimeter systems. The Electromagnetic Calorimeter (ECAL) is located about 1.3m away from the collision point, surrounding the silicon tracker, and is therefore the next subsystem that the particles encounter on their way out of CMS. A detailed overview of the ECAL can be found in [13].

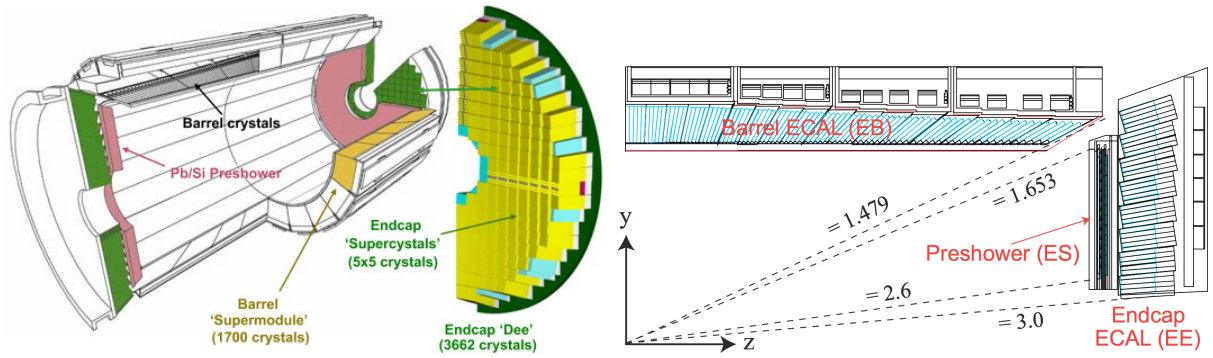


Figure 2.5: (left) Schematic illustration of the ECAL showing its subsystem, namely, the central barrel subsystem in grey and yellow, the pre-shower detector (ES) in pink and the ECAL Endcaps (EE) in green. (right) Partial ECAL cross-section showing the positive pseudo-rapidity values (dashed lines) accessible by the top components of the inner barrel (EB), the pre-shower detector (ES) and the Endcap (EE) [14].

Although its main purpose is to measure the energy of electrons and photons, (with precision $\pm 1\%$), the ECAL, together with the Hadron Calorimeter (HCAL), also plays a role in determining the energy of hadronic jets. One of the challenges in measuring particle energies is the small time interval between consecutive bunch crossings. The material used for the ECAL must therefore have a short scintillation-decay time. It must also satisfy the space constrains inside the solenoid magnet, withstand its strong magnetic field and be suitable for a high radiation environment. For these reasons, the chosen material was lead-tungstate crystal. This crystal structure, which is denser than steel but highly transparent, emits light proportionally to the energy deposited by the particles that pass through it. To avoid particles escaping the ECAL undetected, the crystals are aligned homogeneously and hermetically. The crystals scintillate at approximately 4.5 γ/MeV at 18°C, however, this rate is highly dependent on temperature. Thus, the temperature is kept stable at ± 0.1°C by a purpose built system. In the 25 nanoseconds between bunch crossings, these crystals emit about 85% of light, allowing for lower cross-contamination. This light is then seen by photo-detectors attached to the back of each crystal, and converted into an electrical signal. To cover a pseudo-rapidity range of $|\eta| < 3.0$, the ECAL, similarly to the tracker system, consists of a central barrel component and two endcaps, one at each end. An illustration of the ECAL subsystems can be found in Fig. 2.5.

- **Preshower detector (ES)** – The endcaps are supplemented by a preshower detector, composed of lead and silicon trips detectors. The lead initiates the electromagnetic shower, and the strips provide precision measurements. Due to its highly granular structure, the ES allows for individual resolution of shower clusters. This is particularly important to distinguish between a single high energy photon from a lower energy photon pair resulting from pion decays or other diphoton resonances.

2.2.3 Hadron Calorimeter

Located outside the solenoid magnet, the HCAL [13] is in place to measure the energy of composite particles, e.g. pions or kaons. It also allows for the estimation of the missing transverse momentum p_T^{miss}.

The particles are measured through layers of alternating brass absorbers and plastic scintillator tiles. When the high energy particles encounter the heavy brass nuclei, they produce hadronic showers. The bi-products of the showers are then detected by the scintillators, through electron excitations caused by the passage of charged particles. Photons are then emitted during electron de-excitation, which are guided through wavelength-shifting fibers to photo-diodes.

The HCAL comprises four different sections, Barrel (HB), Endcaps (HE), Outer detector (HO) and the Forward detector (HF). The HB covers a pseudo-rapidity region up to $|\eta| < 1.3$ while the HE can reach a pseudo-rapidity up to $|\eta| < 3$. Lastly, the HF can detect particles in a region of $|\eta| < 5$, through the emission of Cherenkov radiation.

2.2.4 Superconducting Solenoid

One way to identify the charge and momentum of particles is precisely by observing their behaviour within a magnetic field. When a magnetic field is present, electrically neutral particles will not interact with it and will keep following straight trajectories. On the other hand, the trajectories of electrically charged particles will be bent, as a result of the Lorentz force. Particles with opposite charges will bend in opposite directions and particles with lower momentum will have trajectories with higher curvature.

In CMS, a magnetic field is created by a superconducting solenoid coil [9]. The magnet system measures 12.9 m in length and has an inner and outer diameters of 5.9 m and 6.4 m, respectively, making it the largest superconducting solenoid ever built. Rutherford cables conduct a current of 19.14 kA over 2168 turns, creating an uniform magnetic field of 3.8 Tesla, with field lines parallel to the beam direction. To sustain the superconducting regime, the coil is embedded in a liquid-helium cooled aluminium tank, kept at an approximate working temperature of 4.5 Kelvin. The design of the aluminium structure securing the magnet in place, besides counter-acting the mechanical forces induced by the magnetic field itself, also allows for the relatively thin overall design. This is a crucial feature, as the available space inside CMS is limited and, thus, should be prioritized for detector modules. The whole magnet system is then enclosed by an iron yoke, which returns the magnetic flux. In this region, the residual magnetic field is 2 Tesla. With this set-up, the silicon tracker is able to achieve a charge misidentification of less than 5% for muons with $p_T < 200$ GeV.

2.2.5 Muon Chambers

Since muons have a much higher mass compared to electrons, they are less affected by the magnetic field produced by the superconducting solenoid magnet. This allows muons to pass by through all inner sub-systems and eventually escape the solenoid magnet. Thus, the muon chambers are located in the outermost layer of the CMS detector [15]. They measure the muon tracks, allowing for the calculation of the momentum.

The components of the muon chambers are: the *drift tubes* (DTs) located in the barrel section, the *cathode strip chambers* (CSCs) located at the endcaps and the *resistive plate chambers* (RPCs). The barrel section can detect muons in the pseudo-rapidity region $|\eta| < 1.2$ while the endcaps have a reach of $0.9 < |\eta| < 2.4$. Note that there is an overlap between $0.9 < |\eta| < 1.2$.

- **Drift Tubes**
  The drift tubes measure 2 meters in length and have a cross sectional area of 13×42 square millimeters. The tubes themselves are filled with a gas mixture. Moreover, an anode wire can be found at the center of each tube. This allows for a high electrical potential to be created between the wire and the tube walls. When electrically charged muons interact with the gas mixture, ionizing some of the atoms, the free electrons will drift along in the direction of the electric field, created by the CMS magnet, towards the anode wire. The accumulated charge can then be measured as current spikes. The achievable spacial resolution is around $100\ \mu m$. Note that, with this set up, it is not possible to determine the location of the muons along the wire direction.

- **Cathode Strip Chambers**
  The cathode strips are mounted in the endcap region, in trapedoizal chambers. CSCs consist of gaseous detectors containing several perpendicularly aligned wires and cathode strips, which are read out separately. Unlike the drift tubes, the CSCs do not rely on drift effects, making them suitable to operate in non-uniform magnetic fields. They are also capable to withstand a high rate of muons, which is an important feature to operate in the endcap region.

- **Resistive Plate Chambers**
  The resistive plate chambers serve as a complement to the DTs and the CSCs and are also based on gaseous detection. Because the RPCs are operated in avalanche mode, the spatial resolution is lower. Although, the time resolution is significantly higher, since the avalanche of charged particles is extremely fast. This last property is particularly useful when trying to assign tracks to a bunch crossing.

  Together with the tracking system, the transverse momentum resolution for muons in the barrel region, with $20 < p_T < 100$ GeV, is higher than $\sim 2\%$. For muons reaching the endcaps, with $p_T < 1$ TeV, the resolution is still below 10% [16].

2.2.6 Trigger & Data Acquisition System

Because of the extremely high number of collision at the LHC, resulting in around 40 million events every second, storing all the information of such events would reacquire 20 Terabytes of storage per second. This is clearly unfeasible and the majority of event information must be disregarded, which is done by the *trigger system* [17]. The selection of events to be stored for offline analysis is based on the corresponding physics content, the available statistics and bandwidth restrictions.

The first step in the selection process is performed by the Level-1 Trigger (L1). The L1 trigger automatically rejects low-energy events based on input information directly from the calorimeters and the muon systems. Events which survive this selection are read out and temporarily stored to be more carefully evaluated by the High Level Trigger [18]. The High Level Trigger has access to the full detector information and is programmed to make the final selection on whether an event will be stored for further offline analyses, or if it will be permanently deleted.

To analyse all events which have passed the High Level Trigger selection, the Worldwide LHC Computing GRID (WLCG) project was created. The WLCG is a group of computing facilities located across the world and operated by CERN member states. The several facilities are categorized as:

- *Tier 0* computing centers – One of these is located at the CERN data center, in Geneva, while the other is located at the Wigner Research Centre for Physics, in Budapest. Tier 0 facilities store and reconstruct the raw data.
- *Tier 1* data centers – There are currently 13 Tier 1 facilities, located in several countries, which perform further data processing. These facilities also serve as data backups, and as distributors to the Tier 2 facilities.
- *Tier 2* computing centers – There are 160 Tier 2 facilities, mostly operated by universities or scientific institutions. The goal of Tier 2 facilities is to provide enough computing power for event generation and offline analyses. Note that the RWTH Aachen University hosts its own Tier 2 facility, which is operated by the Department of Physics.

Chapter 3

Model Unspecific Searches in CMS

"Theoretical physicists used to explain what was observed.
Now they try to explain why they can't explain what was not observed."

– Sabine Hossenfelder

With the increasing luminosity delivered by the LHC, high energy physics has become synonymous with big data handling. Thus, CMS employs different search strategies to find physics beyond the SM. One possible way to find Beyond Standard Model (BSM) physics is through precision measurements, where SM processes and input parameters are measured ever more precisely, trying to find any discrepancy with current theory predictions. A second analysis strategies are the so-called BSM Searches, which consider a certain theoretical model to specify a signal search region[1]. While dedicated analysis benefit from a higher sensitivity in the probed signal region, they are limited to the final states predicted by their proposed BSM model. With numerous BSM theories being put forward, and considering the finite amount of computational and human resources, an in-depth exploration of all possible signal regions becomes unfeasible. It is important to note that, as a natural consequence of such search strategy, signal regions incompatible with any current BSM theory do not received enough attention. Lastly, there are model-independent searches which do not propose a given BSM model, but instead, are sensitive to several possible signals.

Model Unspecific Search in CMS (MUSiC) [19] was developed as an important complementary approach to dedicated searches, and is the only model-independent analysis within the CMS experiment. MUSiC is a algorithm performing BSM searches considering a wide range of final states. It searches for discrepancies between measured data and simulated SM-like background processes, treating all regions as possible signal regions. MUSiC analyses final states containing at least one electron, muon, photon, jet or b-tagged jet. Such final states may, additionally, contain missing transverse momentum. The SM background estimation is exclusively based on Monte Carlo (MC) methods, and no data-driven approach is used. MUSiC analyses hundreds of final states, and is currently the only model-independent search strategy within CMS. Even though MUSiC losses sensitivity to specific signals, by considering all phase-space regions equally, it can detect BSM signals that have not yet been proposed theoretically, or that have passed unexplored in the large amounts of CMS data. The current chapter first gives an overview of previous MUSiC results in Section 3.1, followed by a description of the measured, and simulated, data sets used in the this work, in Sections 3.2 and 3.3, respectively. Later, the

[1]For instance, one can look at specific final states to search for super-symmetric particles, dark matter particles or the graviton.

different steps of a typical MUSiC analysis are introduced in Section 3.4, namely, the *Skimming* procedure, the *Classification* of final states, and the discrepancy analysis through the *Region of Interest (ROI) Scanning* algorithm. Lastly, a file conversion process is introduced in Section 3.5 such that MUSiC pre-processed data becomes compatible with a machine learning approach.

3.1 Previous MUSiC Results

MUSiC analysis have been actively performed in CMS since 2009 [20–29]. The latest published result regards the RUN2 2016 full data set, at centre-of-mass energy $\sqrt{s} = 13$ TeV and integrated luminosity 35.9fb^{-1} [19]. No significant deviations from SM prediction were found. The final states with the highest level of discrepancy between data and SM simulation are presented in Fig. 3.1. The event class names are specified at the bottom of the plot, the y axis represents the total number of events per class and, at the top of the plot, the corresponding event class p-value is shown. Within each event class bin, the event count contribution from each SM process is shown in different colours, together with the experimentally measured total number of events. The combined systematic and statistical uncertainty of the MC simulation for each class is plotted as a grey crossed pattern within each bin.

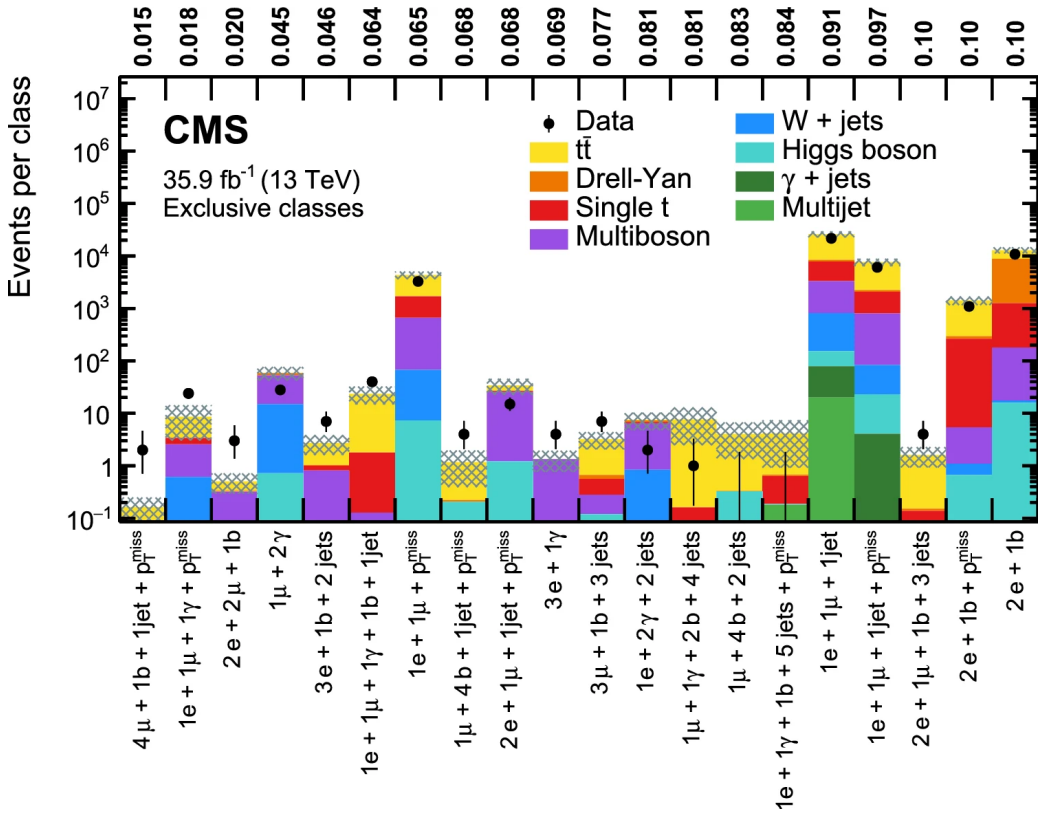


Figure 3.1: The most discrepant exclusive event classes found by the MUSiC analysis in the RUN2 2016 data set are shown at the bottom of the plot. The y axis shows the total number of events per class, and the colours within each bin represent the contributions of different SM processes to the total event count. For each event class, the respective p-value calculated by the MUSiC algorithm is also displayed at the top of the plot. The total MC uncertainty (systematic and statistical) is shown as a grey crossed pattern overlapping each bin, and the measured CMS data is shown as black dots with the corresponding experimental uncertainty error bar [19].

3.2 Collision Data

This thesis uses measured data from proton-proton collisions recorded by CMS, during the 2017 data-taking period. At the time, the LHC was delivering collisions at a centre-of-mass energy of $\sqrt{13}$ TeV, with a proton bunch spacing of 25 nanoseconds. The full 2017 data set is considered, consisting of a total integrated luminosity of $41.48\text{fb}^{-1} \pm 2.3\%$ [30]. The CMS Physics Data and the Monte Carlo Validation group have certified all data sets used throughout this thesis [31].

3.3 Standard Model Simulation

Since protons are composite particles, their collisions within CMS are, in fact, complex interactions between their constituents, i.e. the valence/sea quarks and the gluons. In MC simulations, pseudo-random numbers generators are used to model a hard scattering process, a subsequent parton shower and, lastly, the shower hadronization. These different interaction steps are schematically shown on the left hand side of Fig. 3.2, where a proton-proton collision is illustrated. On the right hand side, example PDF functions are plotted, which contain information regarding the momentum fraction carried by each quark or gluon. The PDFs $xf(x, Q^2)$ are parametrized in x-space, corresponding to the bottom axis, and were calculated for an energy scale of $Q^2 = 10$ GeV [32].

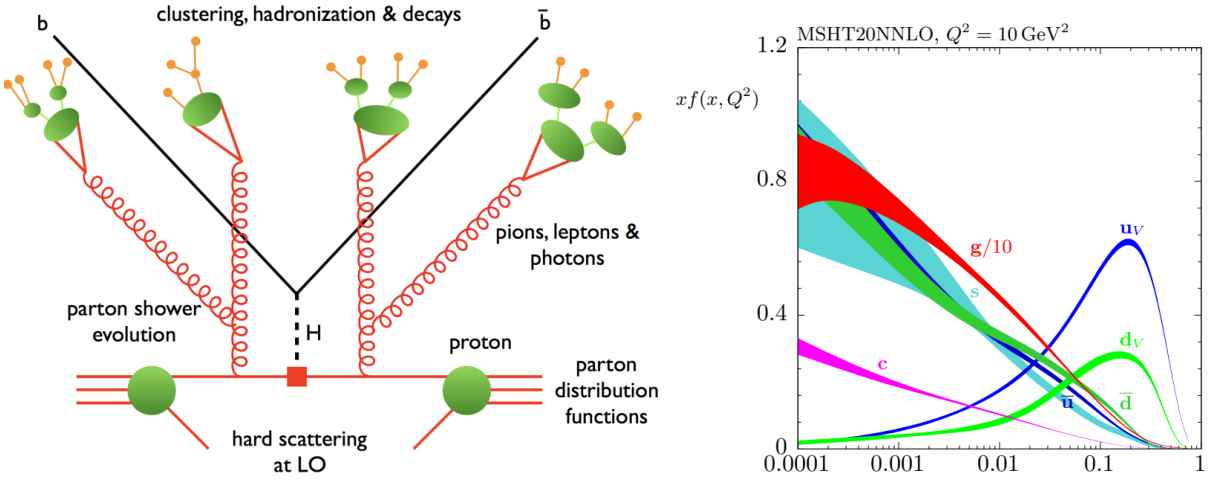


Figure 3.2: (left) Schematic of a proton-proton collision showing the leading order (LO) hard scattering of two quarks, creating a Higgs boson (H), which then decays into the $b\bar{b}$ quark pair. The image shows the additional appearance of parton showers and hadronization. (right) PDFs of up/down valence quarks (u_V/d_V), up/down anti-quarks ($\overline{u}$ /$\overline{d}$), charm quarks (c), sea quarks (s) and gluons (g/10), for an energy scale of $Q^2 = 10$ GeV [32].

For this thesis, the MC generators used to simulate the full range of SM-like processes were MadGraph5_aMC@NLO version 2.2.2 [33], Pythia 8.212 [34], powheg v2 [35] and sherpa 2.1.1 [36]. Additionally, the detector response is modelled with the Geant4 package [37]. The MCgenerators mostly used the NNPDF3.0 *Parton Distribution Functions* (or PDFs) [38] to

calculate cross-sections either to Leading Order (LO) or to Next-to-Leading Order (NLO). If the theoretical prediction of a certain process in known to a higher precision, i.e. to Next-to-Next-to-Leading Order (NNLO) or to Next-to-Next-to-Next-to-Leading Order (N3LO), then the MC predicted cross-section is corrected by a k-factor defined as

$$k = \frac{\sigma_{(N)NLO}}{\sigma_{LO}}. \tag{3.1}$$

3.3.1 Expected Background

All SM processes analysed in this thesis are presented in Table 1. The first column specifies the different background processes considered, while the second column provides further details on the respective process, e.g. minimum thresholds for the considered invariant mass or transverse momentum. Where no details are given, all available data is used. The third and fourth columns show which generator was used, and at which order the process was generated, respectively. Lastly, the fifth column specifies the final cross-section order generated, as corrected by the k-factor, when necessary.

Table 1 : Summary of analysed SM simulated samples. The SM processes are specified in the first column, with details given on the second column. The third and fourth columns refer to which matrix element generator software was used and to which order, respectively. The last column shows the order of the cross section of the respective process, which might have been corrected by a k-factor.

Process	Details	Generator	Generator order	Cross section order
Z($\to l^+l^-$)+jets	$M_{l^+l^-} > 10$ GeV	MADGRAPH	NLO	NNLO
	p_T(Z)> 50 GeV	MADGRAPH	NLO	NNLO
	$M_{l^+l^-} > 120$ GeV	POWHEG	NLO	NNLO
Z($\to \nu\nu$)+jets		MADGRAPH	LO	NLO
W($\to l\nu$)+jets	Inclusive	MADGRAPH	NLO	NNLO
	p_T(W)> 100 GeV	MADGRAPH	NLO	NNLO
	$M_{l\nu} > 100$ GeV	PYTHIA 8	LO	NNLO
γ + jets		MADGRAPH	LO	LO

[Continue on next page]

Process	Details	Generator	Generator order	Cross section order
$t\bar{t}$	Inclusive	POWHEG	NLO	NNLO
	$M_{t\bar{t}} > 700$ GeV	POWHEG	NLO	NNLO
	$t\bar{t}\gamma$	MADGRAPH	NLO	NLO
	$t\bar{t}$W	MADGRAPH	NLO	NLO
	$t\bar{t}$Z	MADGRAPH	NLO	NLO
	$t\bar{t}\gamma\gamma$	MADGRAPH	NLO	NLO
$t\bar{t}t\bar{t}$		MADGRAPH	LO	LO
Top	t(tW-channel)	POWHEG	NLO	NNLO
	t(t-channel)	POWHEG	NLO	NNLO
	t(s-channel)	MADGRAPH	NLO	NLO
	$t\gamma$	MADGRAPH	NLO	NLO
	tZq	MADGRAPH	NLO	NLO
Z($\rightarrow 2l$)γ		MADGRAPH	NLO	NLO
W($\rightarrow 2l$)γ	$p_T(\gamma) > 40$ GeV	MADGRAPH	LO	LO
	$p_T(\gamma) > 130$ GeV	MADGRAPH	NLO	NLO
ZZ	ZZ$\rightarrow 4l$	MADGRAPH	NLO	NLO
	ZZ$\rightarrow 2l2q$	MADGRAPH	NLO	NLO
	ZZ$\rightarrow 2l2\nu$	POWHEG	NLO	NLO
WW	WW$\rightarrow l\nu qq$	POWHEG	NLO	NNLO
	WW$\rightarrow 4q$	MADGRAPH	NLO	NLO
	WW$\rightarrow 2l2\nu$	POWHEG	NLO	NNLO
WZ	WZ$\rightarrow l\nu 2q$	POWHEG	NLO	NLO
	WZ$\rightarrow 3l\nu$	MADGRAPH	NLO	NLO
	WZ$\rightarrow 2l2q$	MADGRAPH	NLO	NLO
	WZ$\rightarrow 1l3\nu$	MADGRAPH	NLO	NLO
	WZ$\rightarrow 1l1\nu 2q$	MADGRAPH	NLO	NLO
$\gamma\gamma$	40 GeV $< M_{\gamma\gamma} <$ 80 GeV	SHERPA	LO	LO
	$M_{\gamma\gamma} > 200$ GeV, $p_T^{\gamma} > 15$ GeV	SHERPA	LO	LO
QCD multi-jet		MADGRAPH	LO	LO

[Continue on next page]

Process	Details	Generator	Generator order	Cross section order
Triboson	ZZZ	MADGRAPH	NLO	NLO
	WZZ	MADGRAPH	NLO	NLO
	WZγ	MADGRAPH	NLO	NLO
	WWW	MADGRAPH	NLO	NLO
	WWZ	MADGRAPH	NLO	NLO
	WWγ	MADGRAPH	NLO	NLO
Higgs	ggH$\rightarrow \tau\bar{\tau}$, WW($2l2\nu$), ZZ($4l$), ZZ($2l2\nu$), Z$\gamma$	POWHEG	NLO	N3LO
	VBF(H$\rightarrow \tau\bar{\tau}$, WW($2l2\nu$), Z$\gamma$)	POWHEG	NLO	NNLO
	VBF(H$\rightarrow \gamma\gamma$)	MADGRAPH	NLO	NNLO
	$t\bar{t}$H(not H $\rightarrow b\bar{b}$)	POWHEG	NLO	NLO
	$t\bar{t}$H(H $\rightarrow b\bar{b}$)	POWHEG	NLO	NLO

3.3.2 Monte Carlo Weights

Prior to the analysis, the number of MC generated events needs to be scaled to the measured luminosity of the CMS data sets used. Thus, each MC event is scaled by its corresponding process-specific luminosity weight factor, $w_{\mathcal{L}}$, given by

$$w_{\mathcal{L}}(\text{process}) = \frac{k \cdot \sigma \cdot \mathcal{L}}{N_{MC}^{events}}, \tag{3.2}$$

where k is the k-factor defined in Eq. 3.1, σ is the MC calculated cross-section, $\mathcal{L}$ is the desired luminosity and N_{MC}^{events} is the number of MC generated events. Moreover, events at the LHC do not happen in isolation, since the proton beams cross in bunches and each bunch contains several inelastic interactions. Due to these extra interactions, there will be an higher number of primary event vertices, leading to a so-called *pile-up* effect. This is corrected by applying an event-specific weight correction factor, w_{PU}, defined as

$$w_{PU}(\text{event}) = \frac{N_{data}^{vertices}}{N_{MC}^{vertices}}. \tag{3.3}$$

The numerator stands for the total number of primary vertices in an event in measured data and the denominator for the primary vertices in the corresponding MC event generated. Usually, more MC events than data events are desired, allowing a more accurate modelling of low-data regions. For instance, in the high-energy regime of classical MUSiC histograms, there may be no data events in certain bins, and only one event in others. Since MC simulation cannot

predict which high-energy intervals will have no measured data, it should provide a smoothly decreasing event probability density, until the expected number of events is less than one in this regime. This possible re-scaling during the event generation is included in the event weight by the factor w_{gen}, which is both process- and event-specific. The total weight applied to each MC generated event is then given by

$$w_{total}(\text{event, process}) = w_{\mathcal{L}}(\text{process}) \cdot w_{UP}(\text{event}) \cdot w_{gen}(\text{event, process}). \tag{3.4}$$

3.4 Analysis Workflow

This thesis introduces a significant change in the classic MUSiC analysis workflow. A schematic of a typical MUSiC analysis workflow is shown within the yellow dashed lines in Fig. 3.3. The necessary steps for a MUSiC analysis incorporating a machine learning approach can be found within the blue dashed lines. The common steps for both analysis are shown in grey, namely, the *Skimming* and the *Classification*. While the *Skimming* remains unaltered for the work developed in this thesis, the second step, the *Classification*, is slightly modified to fit a machine learning approach. It is only after the *Classification* stage that a typical MUSiC analysis and the work in this thesis diverge significantly. The last step in the MUSiC workflow is the *Scanning* process, highlighted in yellow. This step is not performed at all for this thesis. Instead, the *Scanning* algorithm is replaced by the New Physics Learning from a Machine (NPLM), highlighted in blue, which is introduced in Section 4.3. The motivation for such a replacement is also presented in the next chapter, namely in Section 4.2. However, in order to make MUSiC analysis compatible with the NPLM methodology, one must first implement a file conversation algorithm, also highlighted in blue. The file formats used in the analysis are described throughout the present chapter.

3.4.1 Skimming

The *Skimming* converts `ROOT` files, based on the `ROOT` data analysis framework [39], into `pxlio` files, based on the **P**hysics e**X**tension **L**ibrary (PXL) [40]. Within `ROOT` files, the data may be stored in different formats. CMS has developed the Analysis Object Data (`AOD`), the `miniAOD` and the `nanoAOD` formats. These formats disregard a significant portion of low-level detector information, reducing the storage space and run-time at each iteration. This thesis uses `miniAOD` data sets. These data sets are stored in several Tier-2 sites of the WLCG. During *Skimming*, the `ROOT` files are further stripped of any irrelevant information for the MUSiC analysis, *e.g.* information on low energy objects, and only necessary data sets are kept. These trimmed down data sets are saved in `pxlio` files and stored locally, in the Tier-2 Aachen site.

Object & Event Selection

Even the trimmed down `pxlio` files contain more information than necessary for MUSiC analysis, since no events have yet been discarded during the *Skimming* process. Thus, not all events within the `pxlio` file will be compatible with a MUSiC analysis. It is first necessary to determine, without any ambiguity, the physics object content of each event. For this, one must identify the reconstructed individual objects within an event, removing any possible overlap

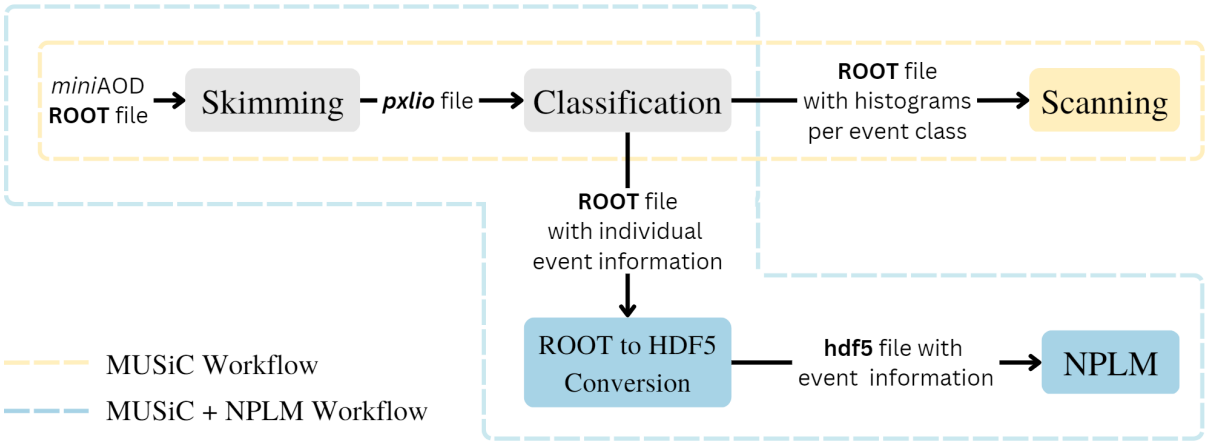


Figure 3.3: Schematic of a typical MUSiC analysis workflow, within the yellow dashed lines, and of the combined MUSiC and NPLM analysis workflow, within the blue dashed lines. The steps illustrated with grey backgrounds, the *Skimming* and the *Classification*, are common to both analysis strategies, while the *Scanning* step in a yellow background is only performed for a typical MUSiC analysis. The steps in a blue background, the file conversation and the NPLM implementation, were the ones introduced in this thesis for a combined MUSiC and NPLM analysis.

between them. Since MUSiC relies exclusively on MC background estimation, the misidentification of reconstructed objects should be minimized as much as possible, without compromising the selection efficiency. Therefore, tight object selection criteria are applied to each event. These criteria are summarized in Table 2.

Table 2: Object selection criteria for different p_T ranges.

Object	p_T (GeV)	Pseudorapidity
Muon (μ)	> 25	$\|\eta\| < 2.4$
Electron (e)	> 25	$0 < \|\eta\| < 1.4442$ or $1.566 < \|\eta\| < 2.5$
Photon (γ)	> 25	$\|\eta\| < 1.4442$
Jet	> 50	$\|\eta\| < 2.4$
b-tagged jet	> 50	$\|\eta\| < 2.4$
Missing transverse momentum (p_T^{miss})	> 100	–

Each selected event in the precious step must have triggered at least one *Single/Double Electron Trigger*, one *Single/Double Muon Trigger*, or a *Single Photon Trigger*. For each trigger, a corresponding minimum transverse momentum is established as a criteria for the event to be accepted. The offline MUSiC analysis places further requirements on the minimum transverse momentum of each trigger. Both selection criteria are presented in Table 3. All events satisfying the criteria above will go through the *Classification* process, described next. Both in the classical MUSiC *Scanning* step, and in the NPLM approach in this thesis, only event classes containing at least one electron or muon are targets for discrepancy searches between data and MC expected background.

Table 3: Summary of online and offline trigger criteria.

Trigger	Trigger level requirement	Analysis requirement
Single muon	$1\ \mu$ with $p_T > 50$ GeV	$\geq 1\ \mu$ with $p_T > 53$ GeV
Single electron	$1\ e$ with $p_T > 115$ GeV	$\geq 1\ e$ with $p_T > 120$ GeV
Double muon	$1\ \mu$ with $p_T > 17$ GeV and 2nd μ with $p_T > 8$ GeV	$\geq 2\ \mu$ with $p_T > 20$ GeV each
Double electron	$2\ e$ with $p_T > 33$ GeV each	$\geq 2\ e$ with $p_T > 40$ GeV each
Single photon	$1\ \gamma$ with $p_T > 175$ GeV	$\geq 1\ \gamma$ with $p_T > 200$ GeV

3.4.2 Classification

Both the *Classification* and the *Scanning* steps occur within the *Three A Physics Analysis* (`TAPAS`) framework [41], developed at RWTH Aachen. `TAPAS` is built on CMS software, `CMSSW` and runs on the local Aachen cluster. In previous MUSiC analysis, `TAPAS` would take the resulting `pxlio` file from the *Skimming*, classify the events of a given process into event classes, and produce `ROOT` files containing histograms of the kinematic variables to be analysed. These histograms were plotted per event class found in the considered process, and would serve as an input to the *Scanning* process. The three possible event classes are detailed below.

Event Classes

- **Exclusive** – An exclusive class is filled solely with events containing all selected physics objects, in the respective multiplicity, specified in the name of the class. Thus, each event can only contribute to one exclusive class. For instance, a final state with physics objects $1e, 1\mu, p_T^{miss}$ only contributes to the $1e + 1\mu + p_T^{miss}$ exclusive class, while a final state with event content $2e, 2\mu, 1jet$ only contributes to the $2e + 2\mu + 1jet$ exclusive class.

- **Inclusive** – An inclusive class is filled with events containing both a minimum set of selected objects, plus any additional non-*jet* physics objects (X). The example event given above, consisting of 1 electron, 1 muon and missing transverse momentum, will, for instance, contribute simultaneously to the classes $1e + X$, $1\mu + X$ and n $1e + 1\mu + X$.

- **Jet-inclusive** – A jet-inclusive class is an inclusive class where the additional objects X are exclusively *jets*. Moreover, since the MC simulation does not model higher jet multiplicities in a sufficiently accurate manner, the *exclusive* classes containing more than 5 jets are assigned to the jet-inclusive class $X + 5jets + \mathrm{N}jets$.

For a better understanding of this classification, Fig. 3.4 illustrates an example construction of event classes from two distinct final states, with one event each, and with event content $1e, 1\mu, p_T^{miss}$ and $2e, 2\mu, 1jet$, respectively. The exclusive event classes are displayed in olive green, the inclusive event classes in light blue and the jet-inclusive event classes in orange. The light grey backgrounds represent which event classes are associated to which final state. Note that event classes within the overlap region of the grey backgrounds contain two events each, while all other classes only contain one event. Moreover, while each final state may contribute to several inclusive or jet-inclusive event classes, it only contributes to one exclusive event class.

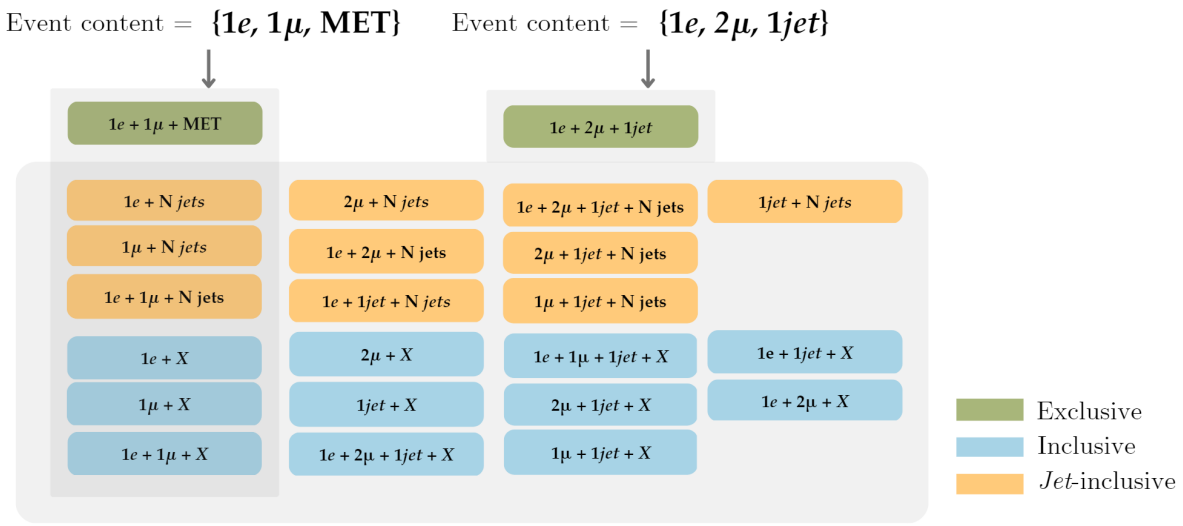


Figure 3.4: MUSiC classification of two final states, $1e + 1\mu + p_T^{miss}$ and $2e + 2\mu + 1jet$. The first state contributes to 1 exclusive, 3 inclusive and 3 *jet*j-inclusive event classes. The second state contributes to 1 exclusive, 11 inclusive and 10 *jet*-inclusive event classes. Note that the classes inside the intersection of the two grey regions contain 2 events each, while the classes outside the intersection contain only 1 event.

For this thesis, however, the *Classification* step was slightly modified. Although the events are still assigned to the same event classes described above, the outputted `ROOT` files now contain kinematic information per individual event, instead of histograms per event class. The process-specific information and event classes found are stored as `TTrees`, while the kinematic information of each event, as well as its MC weight (see 3.3.2), are stored as `TBranches`.

Kinematic Distributions

The search for signs of BSM physics may be carried out in a number of kinematic variables. To be particularly sensitive to high transverse momenta signals, predicted by a large number of BSM models, MUSiC chooses to focus on the three variables described below.

- **Sum of transverse momentum** $(\sum |\vec{p}_T|)$ – It consists on the sum of the transverse momentum of the physics objects in a given event class. It is calculated on a event-by-event basis, for all events satisfying the analysis requirements. For classes with missing transverse momentum, this value is included as well. Thus, $\sum |\vec{p}_T|$ quantifies the total energy in the event. For BSM models predicting new higher-mass particles, the signals would mainly be detected in the tail of this distribution.

- **Invariant mass** (M_{inv}) – It consists on the sum of the calculated invariant mass of all physics objects in a given event, with two or more objects. If the event includes missing transverse momentum, and since the longitudinal momentum component is unknown, the transverse mass of the objects is used instead. For BSM predicting new massive particles appearing as a resonant peak, the M_{inv} distribution of the corresponding decay products is an important search region for SM deviations.

- **Missing transverse momentum** (p_T^{miss} or MET) – It is the total missing transverse momentum of an event, resulting from particles which were not detected. Since SM processes

with neutrinos, and detector resolution, contribute the most to the low p_T^{miss} region, only events containing $p_T^{miss} > 100$ GeV are considered. BSM models containing new particles with large transverse momentum which do not interact with the detector will result in high p_T^{miss} values[2].

3.4.3 Scanning

In a classic MUSiC approach, each histogram, corresponding to a certain event class and stored inside a `ROOT` file, is fed into a search algorithm, the so-called *Scanning* step. The *Scanning* is a statistical analysis aiming at identifying discrepancies between MC simulated SM background and CMS measured data. When a certain region of an histogram is found to contain a significant enough deviation from the SM expectation value, the final state of its corresponding event class may be recommended as a target of dedicated analysis.

The first step in the analysis is to compare the total MC event yield to the measured one. Secondly, the algorithm will define several *Scanning Regions* within the histogram to look for deviations. A *Scanning Region* is any possible connected bin region. Examples of possible regions include single bin regions; a single bin plus its immediately adjacent(s) bin(s), to its left or(and) right; a single bin plus its two immediately adjacent(s) bin(s); etc. However, single or double bin regions are only allowed when the kinematic variable is the invariant mass. If the kinematic variable is either $\sum |\vec{p}_T|$ or p_T^{miss}, the regions must contain a minimum of three bins to minimize the impact of statistical fluctuations. If the considered kinematic distribution contains a total number of bins N_{bins}, the scanning algorithm will define a total number of *Scanning Regions* equal to $N_{bins}\left(N_{bins}-1\right)/2$. Moreover, a check is in place to assure that the scanning regions have a sufficient total number of simulated events, otherwise they are removed from the analysis and not taken further into account. However, individual bins with very low numbers of simulated events can still be part of larger scanning regions. An important remark is that different *Scanning Regions* may share some common bins, meaning that they are not disjoint.

p-value Calculation

For each region which has been identified, the *Scanning* algorithm also calculates their associated p-value, which quantifies the significance of the discrepancy found within that region. A brief introduction to the p-value hypothesis testing is given in the interlude below.

Statistics Interlude – *Hypothesis Testing*

Consider a statistical experiment where observed data may be represented by a random variable. Any conjecture regarding the underlying probability distribution of the data, of which the true distribution is unknown, is called a *statistical hypothesis*. For an observed data set, one may propose several hypothesis H_μ, namely a *Null/Reference hypothesis* $R \equiv H_0$ and one, or more, *Alternative hypothesis* $H_{\mu\neq 0}$. Each hypothesis proposes a different underlying property of the data, and one can quantify the agreement of the observed data with such properties by the calculation of a so-called p-value. If the data is represented by a test statistic t_μ, then the p-value is the one, or two, sided integral of the probability density

[2]For instance, if the LHC produces a Dark Matter particle, which only interacts through gravity, it would be undetectable by any CMS subsystem. The particle would only be noticed by observant physicists analysing the corresponding p_T^{miss} distribution!

function of the test statistic t_μ, $f(t\mu|\mu)$, from the observed $t_{\mu,\text{obs}}$ up to infinity,

$$p_\mu = \int_{t_{\mu,\text{obs}}}^{\infty} f(t\mu|\mu)dt_\mu. \tag{3.5}$$

Thus, the p-value represents the probability of finding data of *equal or greater incompatibility* with predictions from the considered hypothesis. If the p-value is lower than a specified threshold, usually 0.05, the proposed hypothesis is rejected. In particle physics, it is also common to translate the p-value to a Z sigma significance given by

$$Z_\mu = \Phi^{-1}(1 - p_\mu), \tag{3.6}$$

where Φ is the cumulative distribution of the zero mean and unit variance Gaussian function [42]. The relation between the p-value, test statistic t_μ, and Z significance is better understood in plots shown in Fig. 3.5. The left plot shows the p-value as the blue shaded region of the $f(t\mu|\mu)$ function, where the x-axis represents the test statistic t_μ. On the right plot, the Z significance is shown in the x-axis as the distance from the Gaussian mean to the beginning of the p-value integrated blue region.

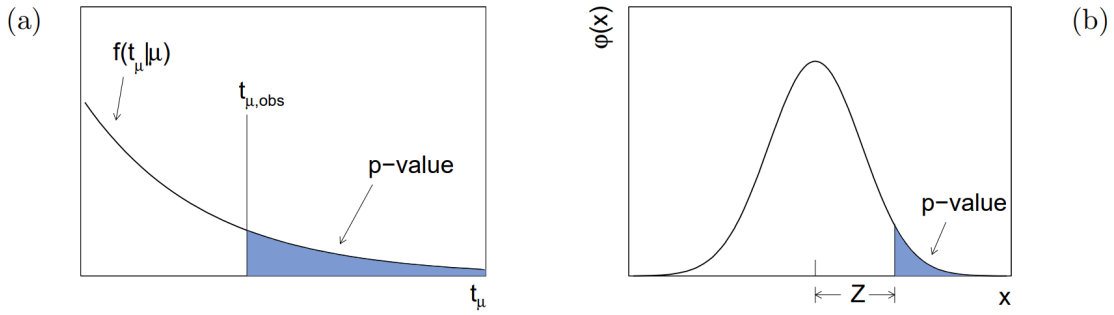


Figure 3.5: (a) Illustration of the relation between the p-value, the observed test statistic t_μ, and the probability density function $f(t_\mu|\mu)$. (b) Gaussian distribution with zero mean and unity variance $\psi(x) = \left(1/\sqrt{2\pi}\right)\exp(x^2/2)$, showing the relation between the significance Z and the observed p-value [42].

The p-value defined for the MUSiC analysis is a composite p-value, defined as

$$p_\text{data} = \begin{cases} \sum_{i=N_\text{data}}^{\infty} C \cdot \int_0^\infty d\lambda \exp\left(-\frac{(\lambda - N_{SM})^2}{2\sigma_{SM}^2}\right) \cdot \frac{e^{-\lambda}\lambda^i}{i!} & \text{, if } N_\text{data} \geq N_{SM} \\ \sum_{i=0}^{N_\text{data}} C \cdot \int_0^\infty d\lambda \underbrace{\exp\left(-\frac{(\lambda - N_{SM})^2}{2\sigma_{SM}^2}\right)}_{\text{systematics}} \cdot \underbrace{\frac{e^{-\lambda}\lambda^i}{i!}}_{\text{statistics}} & \text{, if } N_\text{data} < N_{SM}, \end{cases} \tag{3.7}$$

where N_data is the measured number of events; C is a normalization to unity factor; and $N_{SM} \pm \sigma_{SM}$ is the SM simulated number of events with the corresponding standard deviation. Note that σ_{SM} accounts for both the statistical and systematic uncertainties. The statistical fluctuations are assumed to be Poisson distributed, while the systematic uncertainties, treated as nuisance parameters, are modelled to follow a Gaussian distribution. The p-value is symmetric to the excess and deficit of measured events compared to MC simulated events. If there are more measured than simulated events, then the p-value is given by the first line of Eq. 3.7, where

the sum is performed from the number of measured events up to positive infinity. On the other hand, if there are less measured events, the p-value is given by the second line in the equation, where the sum runs from zero up to the observed number of data events.

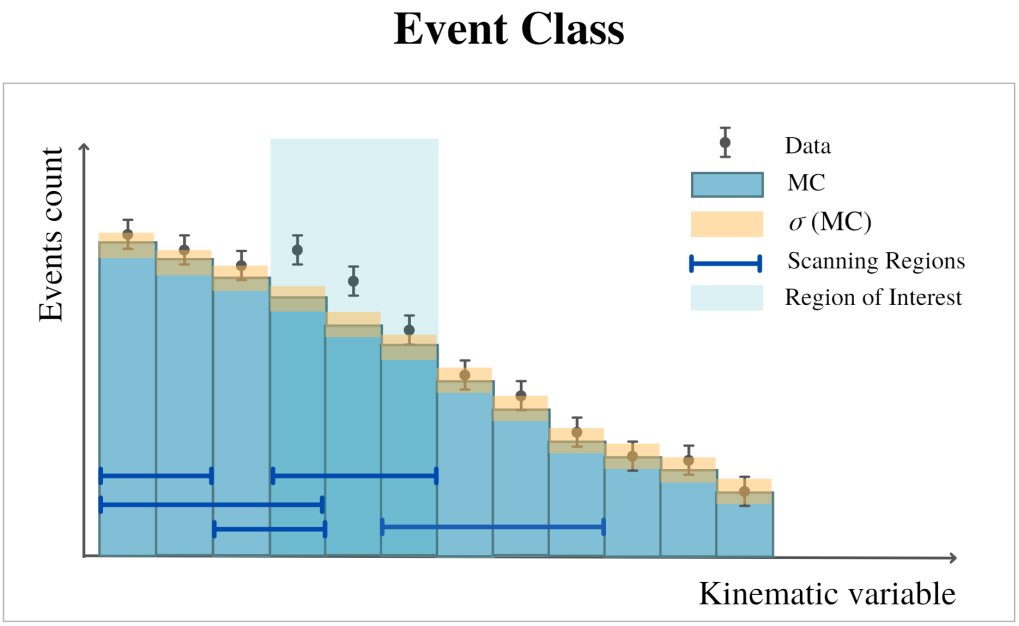


Figure 3.6: Schematic illustration of the MUSiC *Scanning* procedure showing example *Scanning Regions*, in dark blue lines, and the identified *Region of Interest*, highlighted with a light blue background, where the data points (black dots) are more discrepant from the background estimation (bins). The total MC uncertainty (systematic and statistical) is illustrated as the yellow regions.

At the end of the analysis, the region with the lowest p-value, $p_{\text{min}}^{\text{data}}$ (and, therefore, with the largest MC/data discrepancy) is defined as the *Region of Interest*. This region can then be signaled as a promising BSM search region for dedicated analysis. The *Scanning* procedure is illustrated in Fig. 3.6, where the schematic of an histogram is shown with some possible scanning regions signalled by the dark blue lines and the *Region of Interest* highlighted by the light blue background. The yellow regions in the plot labelled $\sigma(MC)$ represent the total MC simulation uncertainty.

Look-Elsewhere Effect

The p-value calculation introduced above only allows to quantify the probability of finding a discrepancy within a specific scanning region. Since the *Scanning* algorithm analyses many connected bin regions, calculating several associated p-values, the probability of finding a significantly low p-value in an arbitrary region is higher than if the algorithm only analysed the entire distribution at once. Thus, the significance of the p-values found is overestimated and must be corrected. This correction is implemented by transitioning from a locally defined p-value to a globally defined $\tilde{p}$-value, known as the *Look-Elsewhere Effect* [43]. The calculation of the $\tilde{p}$-value relies on pseudo-experiments, where the MC simulation is compared to other MC simulation, which acts as a pseudo-data set with no BSM signals. These pseudo-experiments go through the exact same *Scanning* procedure described above, where *Scanning Regions* are

defined and the corresponding p-values are calculated. Several thousand pseudo-experiments are ran where, for each one, the pseudo-data set is randomly varied according to the associated uncertainties. This results in a different *Region of Interest* from the one found with measured data. When all local p-values from the pseudo-experiments are known, the number of experiments with a smaller local p-value, p_{min}, than the one observed with measured data, $p_{\text{min}}^{\text{data}}$, is recorded as $N_{\text{pseudo}}(p_{\text{min}} < p_{\text{min}}^{\text{data}})$. The $\tilde{p}$-value is then calculated as

$$\tilde{p} = \frac{N_{\text{pseudo}}(p_{\text{min}} < p_{\text{min}}^{\text{data}})}{N_{\text{pseudo}}^{\text{total}}} , \tag{3.8}$$

where $N_{\text{pseudo}}^{\text{total}}$ is the total number of pseudo-experiments ran. The $\tilde{p}$-value therefore quantifies the probability of finding a discrepancy of equal or lesser compatibility in *any* possible region of the distribution. After the $\tilde{p}$-values are calculated for all considered distributions, of the same kinematic variable, they can be compared to each other as a whole. This is precisely the end goal of the *Scanning* algorithm.

3.5 Data Conversion

Most machine learning applications, including NPLM, work with files in the Hierarchical Data Format (HDF5) [44], which allows for easier management of large, and possibly heterogeneous, data sets. Thus, it is necessary to convert `ROOT` files, produced during the modified *Classification* step, into HDF5 files. This conversation is necessary for both `ROOT` files corresponding to measured CMS data and to files corresponding to MC simulated data.

Each `ROOT` file contains events from a single SM process[3] and contain, at `TTree` level, all MUSiC event classes found. Within each class, there are `TBranch` levels corresponding to the kinematic variables $\sum |\vec{p}_T|$ (stored as `"SumPt"`), M_{inv} (stored as `"MET"`) and p_T^{miss} (stored as `"InvMass"`). The number of entries in each `TBranch` corresponds to the total number of events in the class. Within the conversation algorithm, the `TBranches` are read as Python arrays, where each entry stores the corresponding kinematic information of a single event. The three final arrays, one for each kinematic variable, are then stored as data sets in a newly created HDF5 file, corresponding to the respective event class. Additionally, each event class `TTree` in the `ROOT` file contains a `TBranch` storing a partial MC event weight, equal to $w_{\text{PU, event}} \cdot k \cdot \sigma_{\text{process}}$. Thus, during the conversation algorithm, the calculation of the full MC weight is implemented, as described in Eq. 3.4. The process-specific information necessary for this calculation is stored at `TTree` level: the process name is stored as `"ProcessName"`, the total number of (accepted or not) events as `"EvCounts"`, the total number of weighted events as `"TotalEvents"` and the sum of the generator weights as `"TotalEventsUnweighted"`. After the calculation is performed and each event weight has been updated, a new data set containing all MC total event weights is appended to the corresponding HDF5 file. For `ROOT` files corresponding to CMS measured data, the conversation step simply creates a data set filled with unity weights in the HDF5 file. The conversation algorithm can be found in Appendix B.

Finally, the conversation step is tested for the four most populous event classes out of the twenty most discrepant classes found in 2016 (Fig. 3.1), namely, the class $1e + 1\mu + p_T^{miss}$ (Fig.3.7), the class $1e + 1\mu + 1jet$ (Fig. 3.8), the class $1e + 1\mu + 1jet + p_T^{miss}$ (Fig. 3.9)

[3]Note, however, that different `ROOT` files may correspond to the same process, but contain distinct events.

and the class $2e + 1bJet$ (Fig. 3.10). Comparing histograms produced from `ROOT` files from a typical MUSiC analysis[4] (upper plots) with histograms produced from HDF5 files (bottom plots), it is clear that the conversation algorithm is accurate. This can be verified through the total event counts for Monte Carlo and Data samples. Note that the classic MUSiC plots constituted preliminary earlier works and so, there are small differences in event counts for specific SM process groups. The bottom plots, produced from HDF5 files, use updated Monte Carlo samples, which explain the small disparities in some event counts.

[4]The plots were provided by Lorenzo Vigilante. Note that the bin width varies within each histogram, according to a bin width optimization algorithm implemented within MUSiC.

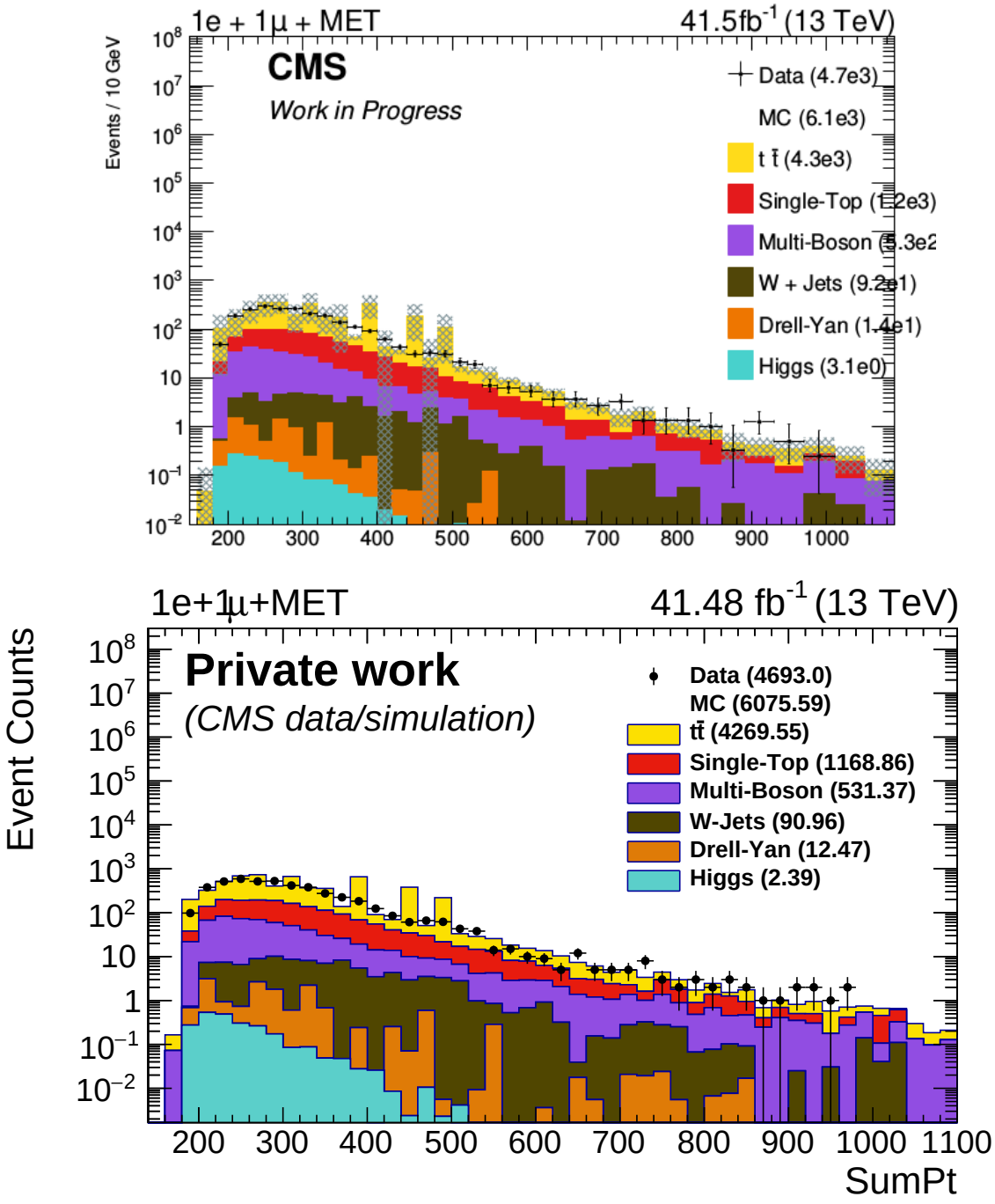


Figure 3.7: (top) Total transverse momentum histogram for the event class $1e + 1\mu + p_T^{miss}$, taken directly from the outputted `ROOT` file from the unmodified *Classification* step; (bottom) Total transverse momentum histogram for the same class, where each event information was read from the corresponding HDF5 file.

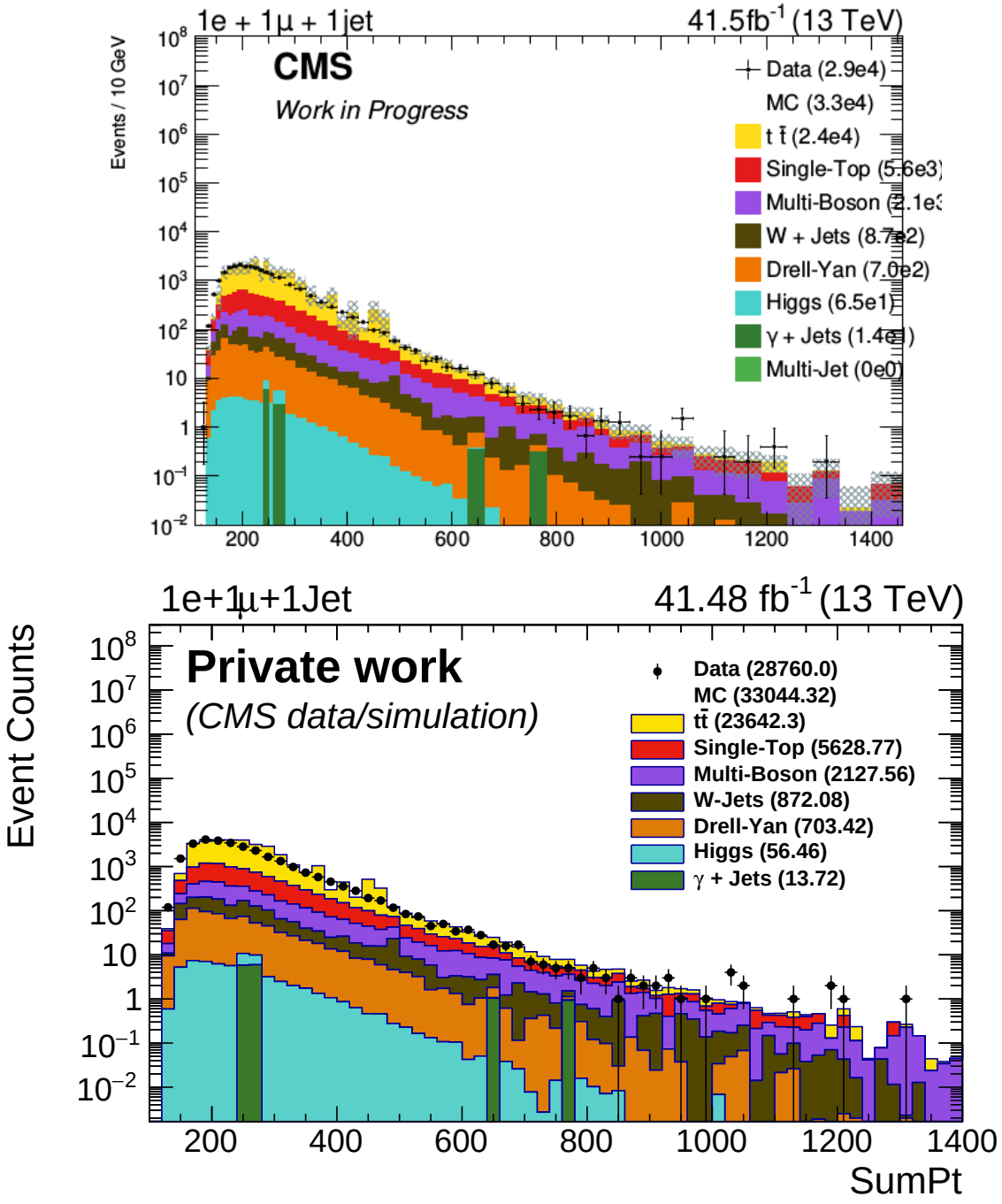


Figure 3.8: (top) Total transverse momentum histogram for the event class $1e + 1\mu + 1jet$, taken directly from the outputted `ROOT` file from the unmodified *Classification* step; (bottom) Total transverse momentum histogram for the same class, where each event information was read from the corresponding HDF5 file.

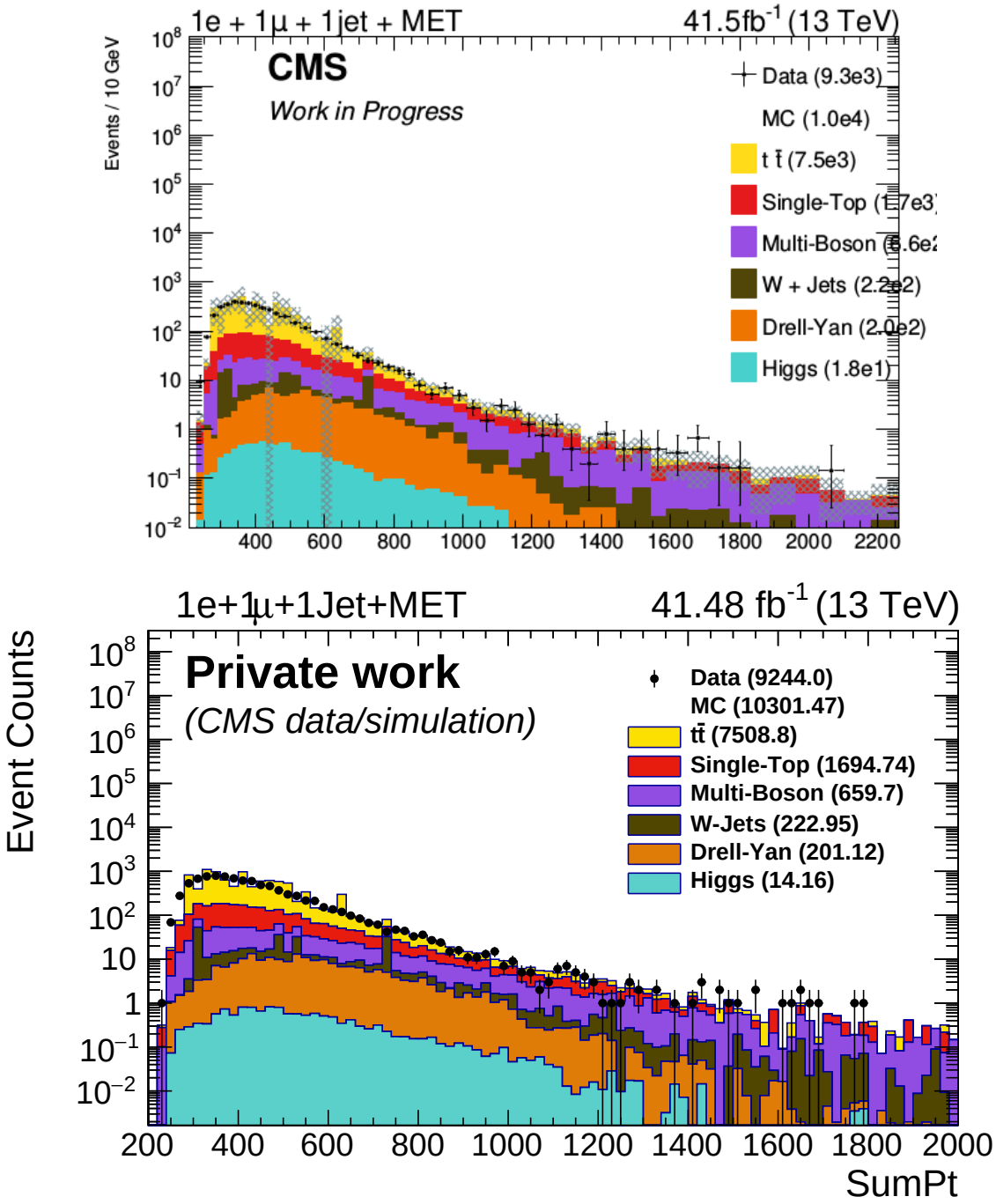


Figure 3.9: (top) Total transverse momentum histogram for the event class $1e + 1\mu + 1jet + p_T^{miss}$, taken directly from the outputted `ROOT` file from the unmodified *Classification* step; (bottom) Total transverse momentum histogram for the same class, where each event information was read from the corresponding HDF5 file.

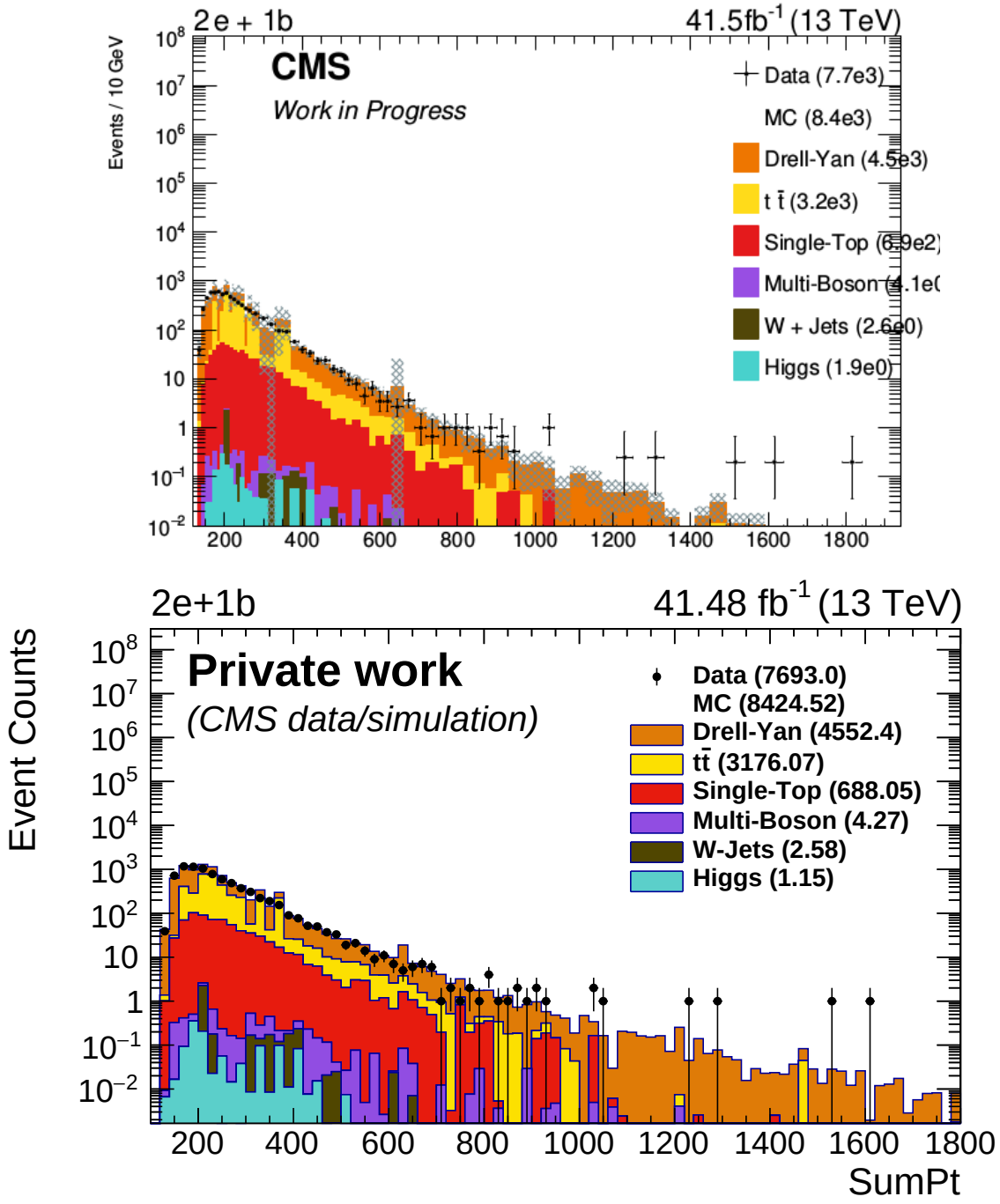


Figure 3.10: (top) Total transverse momentum histogram for the event class $2e + 1b - jet$, taken directly from the outputted `ROOT` file from the unmodified *Classification* step; (bottom) Total transverse momentum histogram for the same class, where each event information was read from the corresponding HDF5 file.

Chapter 4

Machine Learning Methodology

"We are not interested in the fact that the brain has the consistency of cold porridge."
– Alan Turing

Machine learning, which is a sub-field of artificial intelligence, has been around for a couple of decades now, as W. Pitts invented Artificial Neural Networks (ANNs) in 1943 [45]. The goal of a machine learning algorithm is to *learn* certain properties about the data it receives as an input. More concretely, one can use the definition of *"learning"* from Mitchell [46],

> *"A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T**, as measured by **P**, improves with experience **E**."*

There are several kinds of tasks T where a machine learning algorithm is better suited than a fixed computer programme. Some selected examples of complex tasks tackled by machine learning algorithms are [47]

- ***Classification –*** the algorithm is designed to find the probability that a given input data belongs to a certain predefined category, labelled by an integer k. Thus, the algorithm will determine a function $f : \mathrm{R}^n \rightarrow \{1, ..., k\}$, taking the n-dimensional vector $\vec{x}$ as input and giving the numerical value $y = f(\vec{x})$, corresponding to a certain category, as output. Alternatively, f can also output a probability density function over the categories, as opposed to a single value k.

- ***Regression –*** the algorithm is designed to predict a numerical value given some input data. Although, unlike the classification task described above, the value need not be an integer and does not represent a category. Instead, the algorithm will produce a function $f : \mathrm{R}^n \rightarrow \mathrm{R}$.

- ***Anomaly Detection –*** the algorithm takes several data samples as input and it is designed to recognize and predict patterns in the data set. It can then identify if new data samples are anomalous, when compared to the previously analysed data.

The above list is by no means an exhaustive one, as many other tasks can be tackled by machine learning algorithms.

The simplest machine learning algorithm is the Feed-forward Neural Network (FNN), which is introduced in Section 4.1. The mathematical construction of FNN is presented, followed by an explanation of the required training procedure. Section 4.2 then discusses the advantages of neural networks, over binned histograms, as functions approximants. Unless otherwise specified, the following sections are based on [47, 48].

4.1 Theory of Feed-forward Neural Networks

One starts by defining a *neural network* simply as a set of functions. Consider now a real function $f_{\vec{a}}(x, \mathbf{w})$, where x is a d-dimensional random variable, $\mathbf{w}$ are free parameters and the vector $\vec{a}$, of integers, specifies the type of neural network. The family of real functions for all possible parameters $\mathcal{F}_{\vec{a}} = \{f_{\vec{a}}(x, \mathbf{w}), \forall\ \mathbf{w}\}$ then corresponds to a specific neural network *architecture*. To be more specific, consider an integer value L to be the number of building blocks in the network, referred to as *layers*, which are more concretely defined later in this section. Moreover, let a^l, with $l \in \{1, 2, ..., L\}$, represent the total number of elements, or *neurons*, within the l-th layer. Each neuron in the l-th layer can then be labelled as a^l_n, where $n \in \{1, ..., a^l\}$. The *architecture* of a neural network can then be specified as $\vec{a} = (a^1, a^2, ..., a^L)$. Note that the first layer, $l = 0$, is often called *input layer*, and the last l-th layer is called *output layer*. Any layers in between are referred to as *hidden layers*. For instance, a $(1, 5, 1)$ network contains three layers, where the first and last layers contain a single neuron, and the second layer contains five neurons. Additionally, following the notation from MATHEMATICA [49], layers can be categorized either as

- **Element-wise layers** s **–** apply a scalar function s, the so-called *activation function*, to each element of the input vector. Thus, the output vector has the same dimensionality as the input vector. The goal of this scalar function is to establish a decision boundary for when a certain neuron is activated. The most commonly used activation functions are later discussed in subsection 4.1.1.

- **Linear layers** λ **–** apply a linear transformation such that the dimension of the output vector, m, might be different from that of the input vector, n. Such a transformation is defined as

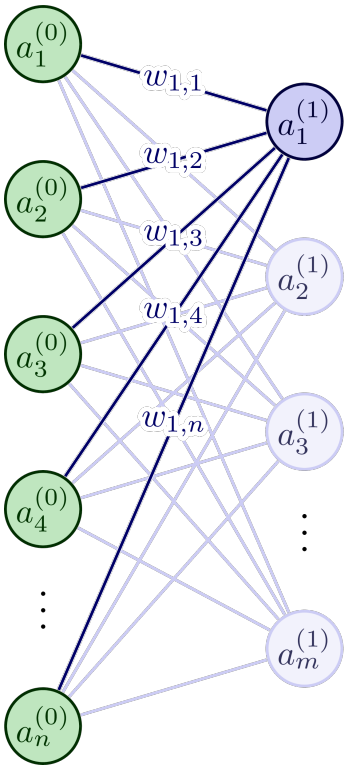
$$[\lambda(a^{(l)}_n)]_m = \sum_{n=1}^{n} w_{m,n} a^{(l)}_n + b_m \tag{4.1}$$

  where some free parameters are contained in $m \times n$ dimensional *weight matrix*, where each entry $w_{m,n}$ is a multiplicative factor, called a *weight*; the remaining free parameters are commonly called *bias* values, and are represented by b_m. The weights and bias are applied by the n-th neuron to the output of the m-th neuron. Thus, the neural network has a total number of $m(n + 1)$ free parameters.

A *layer*, as mentioned above, is then defined as a composition of linear and an element-wise layer. When a specific set of values is chosen for the free parameters in the network, the function $f_{\vec{a}}(\cdot, \mathbf{w})$ is defined as a composition of functions

$$f_{\vec{a}}(\cdot, \mathbf{w}) = \lambda_{\mathbf{1}} \circ \mathbf{s_1} \circ ... \circ \lambda_{\mathbf{L}} \circ \mathbf{s_L}. \tag{4.2}$$

Fig. 4.1 illustrates the application of weights and bias by each neuron of the network. The input layer, $l = 0$, is shown in green and contains $a^0 = n$ neurons, while the output layer, $l = 1$, is shown in blue and contains $a^m = m$ neurons. Each neuron is represented as a circle labelled by $a_i^{(l)}$, where l refers to the l-th layer, containing a^l neurons, and $i \in \{1, ..., a^l\}$ specifies the neuron. Moreover, the value of first neuron in the output layer, $a_1^{(1)}$, is shown explicitly as the result of a scalar function $s = \sigma$ being applied to the output of the linear transformation of the vector $\vec{a}^{(0)} = (a_1^{(0)}, a_2^{(0)}, ..., a_n^{(0)})$. In matrix notation, one can see how the vector $\vec{a}^{(0)}$ is linearly transformed by the weight $w_{m,n}$ and bias b_m values, and subsequently fed into the σ function, to become the $\vec{a}^{(1)} = (a_1^{(1)}, a_2^{(1)}, ..., a_m^{(1)})$ vector.



$$= \sigma\left(w_{1,0}a_0^{(0)} + w_{1,1}a_1^{(0)} + \ldots + w_{1,n}a_n^{(0)} + b_1^{(0)}\right)$$

$$= \sigma\left(\sum_{i=1}^{n} w_{1,i}a_i^{(0)} + b_1^{(0)}\right)$$

$$\begin{pmatrix} a_1^{(1)} \\ a_2^{(1)} \\ \vdots \\ a_m^{(1)} \end{pmatrix} = \sigma\left[\begin{pmatrix} w_{1,0} & w_{1,1} & \cdots & w_{1,n} \\ w_{2,0} & w_{2,1} & \cdots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m,0} & w_{m,1} & \cdots & w_{m,n} \end{pmatrix}\begin{pmatrix} a_1^{(0)} \\ a_2^{(0)} \\ \vdots \\ a_n^{(0)} \end{pmatrix} + \begin{pmatrix} b_1^{(0)} \\ b_2^{(0)} \\ \vdots \\ b_m^{(0)} \end{pmatrix}\right]$$

$$a^{(1)} = \sigma\left(\mathbf{W}^{(0)}a^{(0)} + \mathbf{b}^{(0)}\right)$$

Figure 4.1: Illustration of a neural network with architecture $(1, 1)$. Each neuron is illustrated as a circle labelled by $a_i^{(l)}$, where l refers to the l-th layer, containing a_l neurons, and $i \in \{1, ..., a^l\}$ specifies the neuron. The input layer is shown in green and contains n neurons, while the output layer is shown in blue and contains m neurons. The blue lines connecting the neurons are labelled by the corresponding weight matrix elements $w_{m,n}$. Moreover, in matrix notation, one can see how the vector $\vec{a}^{(0)} = (a_1^{(0)}, a_2^{(0)}, ..., a_n^{(0)})$ is linearly transformed by the weight $w_{m,n}$ and bias b_m values, and subsequently fed into the σ function, to become the $\vec{a}^{(1)} = (a_1^{(1)}, a_2^{(1)}, ..., a_m^{(1)})$ vector.[50].

Moreover, if, for a certain network parametrized by $\vec{a}$, the outputs from layer $l - 1$ are fed as an input to every neuron of the following layer l, for every layer in the network, the network is said to be *fully connected*. In this case, its number of free parameters $N_{par}(\vec{a})$ is given by

$$N_{par}(\vec{a}) = \sum_{l=0}^{L-1} a_{l+1}(a_l + 1). \tag{4.3}$$

This type of simple neural network is referred to as a FNN due to the forward behaviour of the data inside the network, i.e. the data is propagated from the input to the output layer.

An example of a fully-connected FNN with architecture $(1, 3, 1)$ is illustrated in Fig. 4.2. The input layer is shown in green and contains four neurons, the three hidden layers, shown in blue, contain five neurons each and, finally, the last layer, shown in dark red, contains three neurons. Note that each neuron is illustrated as a circle labelled by $a_i^{(l)}$ where, as mentioned previously, l refers to the l-th layer containing a^l neurons and $i \in \{1, ..., a^l\}$ specifies the neuron. Lastly, the arrows inside the network represent the application of weights and bias as explained previously.

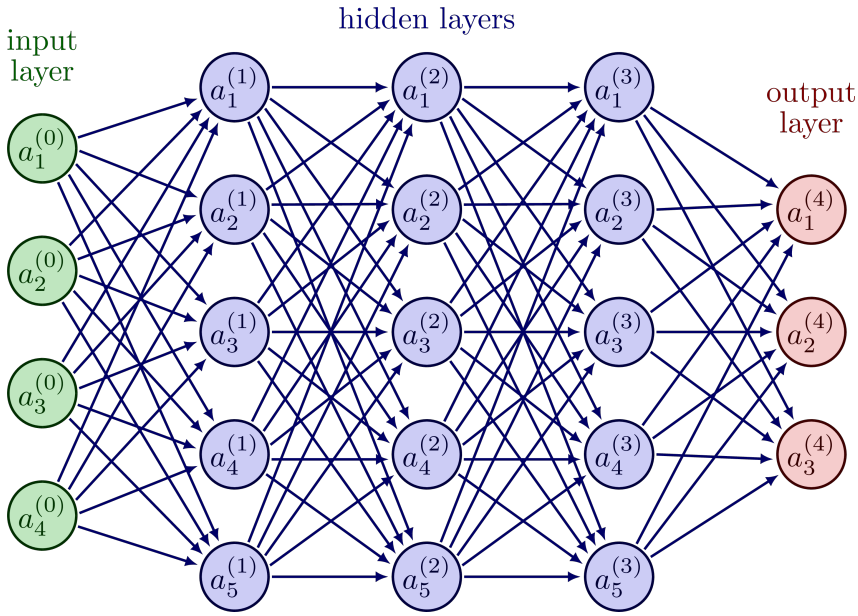


Figure 4.2: Illustration of a fully-connected FNN with architecture $(1, 3, 1)$. Each neuron is illustrated as a circle labelled by $a_i^{(l)}$, where l refers to the l-th layer, containing a^l neurons, and $i \in \{1, ..., a^l\}$ specifies the neuron. The input layer ($l = 0$) is shown in green and contains $a^0 = 4$ neurons; the hidden layers ($l = 1, 2, 3$) are shown in blue and contain five neurons each $a^1 = a^2 = a^3 = 5$; and the output layer ($l = 4$) is shown in dark red, containing $a^4 = 3$ neurons. The arrows represent the application of weights and bias as previously explained [50].

4.1.1 Activation Functions

The choice of scalar function s to be applied by an element-wise layer is dependent on the problem at hand. The simplest example is the Heaviside function, defined by [51]

$$H(x) := \begin{cases} 0, & \text{, if } x < 0 \\ 1, & \text{, if } x \geq 0, \end{cases} \tag{4.4}$$

where a neuron is only activated if its input variable x is a non-negative value. Other examples include the *logistic sigmoid* function σ and the *Rectified Linear Unit* function $ReLU$, defined below [51, 52]

$$\sigma(x) = (1 + e^{-x})^{-1} \tag{4.5}$$

$$ReLU(x) = x \cdot H(x) = \max(0, x). \tag{4.6}$$

The σ function acquired its name from its *sigmoidal* behaviour, as it approaches zero for increasingly negative values of x, $\lim_{x \to -\infty} \sigma(x) = 0$, and one for increasingly higher values

of x, $\lim_{x\to\infty}\sigma(x) = 1$. This behaviour becomes relevant in the next section **??**, when neural networks will be discussed as arbitrary function approximants. It is also relevant to note that σ is a real analytic function, being everywhere infinitely differentiable. Moreover, because $\sigma(x) \in [0, 1]$, the function is bounded both from below and above, and is commonly used to describe probabilities. As a final remark, the logistic sigmoid function may also be seen as a continuous counter-part of the Heaviside function.

All three activation functions are shown in Fig. 4.3 [53], in the interval $x \in [-1, 1]$. The Heaviside function is plotted in orange, the logistic sigmoid function in blue and the $ReLU$ function in dark green. Note that the σ function was modified by a shape parameter $k = 5$, defined as $\sigma_k(x) := \sigma(kx)$, for visualization purposes. Other example activation functions are also shown in the plot, namely the $Mish$ and $LReLU$ functions. More details on these, and other activation functions can be found in [51, 54].

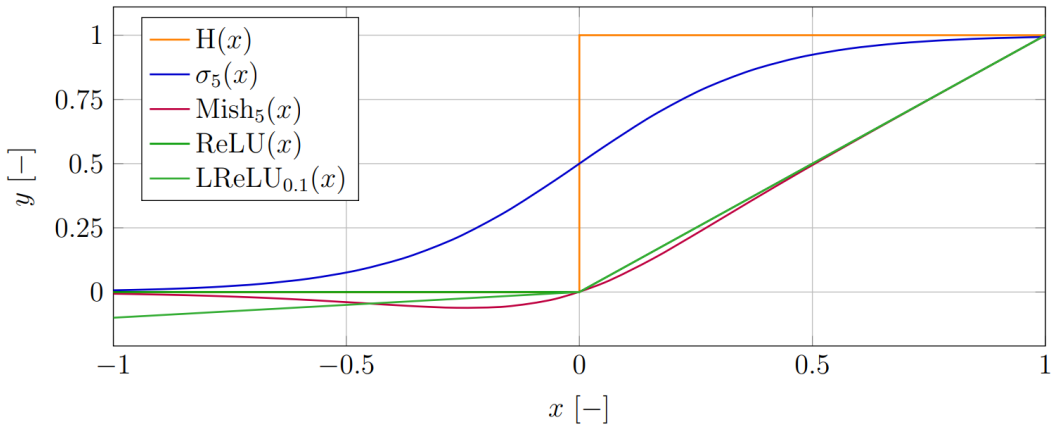


Figure 4.3: Plotting of different activation functions. The Heaviside function $H(x)$ is plotted in orange; the logistic sigmoid function $\sigma_5(x)$ is plotted in blue with shape parameter $k = 5$ and the $ReLu$ function is plotted in dark green. The $Mish$ and $LReLU$ functions are also plotted in red and light green, respectively [53].

4.1.2 Loss Functions

Once the neural network parameters are specified via $\vec{a}$, one must evaluate the performance of the network $f_{\vec{a}}$. What Mitchell [46] previously referred to as a *"performance measure P"*, is now known, in the machine learning literature, as a *loss function*. The present work will only concern supervised machine learning methods, i.e. all the data are labelled data, so this subsection introduces loss functions in this context. For a deeper understanding on the construction of loss functions for semi-supervised or unsupervised learning (unlabelled data) please refer to [51, 55, 56].

The performance of the network can be evaluated by giving it, as an input, a test data set $\mathcal{D}^{test} = \{x_j^{test}, y_j^{test}\}$, where $j \in \{1, 2, ..., D\}$ and D is the number of data tuples. Given this information, it is possible to define the loss function L^{test} for the network $f_{\vec{a}}$, representing a quantization of its performance, as the *Mean Squared Error* function ([48], pp.107),

$$L[f_{\vec{a}}]^{test} = \frac{1}{D}\sum_{j=1}^{D}\left[f_{\vec{a}}(x_j^{test}, \mathbf{w})^{test} - y_j^{test}\right]^2 . \tag{4.7}$$

It is clear that the loss function $L[f_{\vec{a}}]^{test}$ approaches zero as the network prediction $f_{\vec{a}}(x_j^{test}, \mathbf{w})$ approaches the value of the true test label y_j^{test}. Thus, for the network to make better predictions, the algorithm must find the optimal weight and bias values, contained in $\mathbf{w}$, such that $L[f_{\vec{a}}]^{test}$ is minimized. To design a machine learning algorithm capable of such optimization procedure, once can first give the network an input in the form of a training data set $\mathcal{D}^{train} = \{x_j^{train}, y_j^{train}\}$. To minimize its corresponding loss function $L[f_{\vec{a}}]^{train}$, one can simply solve for its gradient to equal 0. A common approach to achieve this is *Stochastic Gradient Descent (SGD)* [57, 58], based on a gradient descent method described by

$$\vec{a} \leftarrow \vec{a} - \eta \nabla L[f_{\vec{a}}], \tag{4.8}$$

where the step size parameter η is referred to as the *learning rate* and

$$\nabla L[f_{\vec{a}}] := \frac{1}{D} \sum_{j=1}^{D} \nabla \left[f_{\vec{a}}(x_j, \mathbf{w}) - y_j \right]^2 . \tag{4.9}$$

Since $L[f_{\vec{a}}]$ is written as a sum over the data set $\mathcal{D}$, its total gradient can also be expressed as a sum of the individual gradients corresponding to each data tuple (x_j, y_j) in the data set $\mathcal{D}$, which is what Eq. 4.9 precisely expresses. This procedure allows the neural network to gain *"experience E"*, commonly known as the *training* of the network.

Since problems tackled by machine learning usually reacquired large training data sets, employing all its data points to update the network parameters becomes computationally expensive. So, the idea introduced by stochastic gradient descent is to modify Eq. 4.8 such that only certain data points j, chosen uniformly at random, are considered to update the network parameters, instead of the full data set. Thus, the new update equation becomes

$$\vec{a} \leftarrow \vec{a} - \eta \nabla \left[f_{\vec{a}}(x_j, \mathbf{w}) - y_j \right]^2 . \tag{4.10}$$

Note how the learning rate η has a significant impact in the training procedure, as it will impact the computational time reacquired to minimize the gradient.

Once a set of weights and bias values is found that better fits the data, one must replace the previous parameters $\vec{a}$ by the new ones. Then, one must update the loss function and calculate it once again. For this, it is necessary to know the derivative of $L[f_{\vec{a}}]$, which can be calculated by a process known as *backpropagation*, introduced by Rumelhart et al. [59].

Besides stochastic gradient descent, there are other optimization algorithms. Two examples are the ADAM algorithm [60] or the *Root Mean Squared Propagation* (RMSprop) algorithm [61]. RMSprop is employed in Chapter 4.3 and is also based on stochastic gradient descent.

4.2 Neural Networks as Function Approximants

Following the construction presented above, it is possible to show that a single hidden layer neural network can approximate, to arbitrary precision, and making use of the logistic sigmoid function, any continuous function[1] taking as an input a d-dimensional variable [62–64]. Without presenting the full mathematical proof, consider the following heuristic demonstration of the

[1]The function must be defined in a compact domain of R^N.

power of neural networks as function approximants. Let $f_{\vec{a}}$, with $\vec{a} = (1, 2, 1)$, be a 3 layer neural network taking a one dimensional input x in the first layer. Let the two neurons in the hidden layer, labelled by 1 and 2, apply a logistic sigmoid function to their inputs, and send their outputs to the same single neuron of the last layer. The corresponding function describing $f_{\vec{a}}$ reads [65]

$$f_{(1,2,1)}(x) = w_1{}'\sigma(w_1 x + b_1) + w_2{}'\sigma(w_2 x + b_2) + b', \tag{4.11}$$

where w_i and b_i, with $i \in \{1, 2\}$, correspond to the weights and bias applied by the hidden neurons, and $w_i{}'$ and b' correspond to the weights and bias applied by the neuron in the last layer. For the specific case when $w_1{}' = -w_2{}' = w'$ and $b' = 0$, Eq. 4.11 reduces to

$$f_{(1,2,1)}(x) = w'[\sigma(w_1 x + b_1) - \sigma(w_2 x + b_2)]. \tag{4.12}$$

Setting $w' = 1$, Fig. 4.4 shows $f_{(1,2,1)}(x)$ plotted for different values of w_i and b_i. By varying the values of w_1 and w_2, the transition between $f_{(1,2,1)}(x) = 0$ and $f_{(1,2,1)}(x) = 1$ can be made arbitrarily sharp. This effect can be seen by comparing the function plotted in blue, where $w_1 = w_2 = 1$ and the transition is smoother, to the function plotted in red, where $w_1 = w_2 = 10$ and the transition is sharper. On the other hand, varying the values of b_1 and b_2 can make the domain in which $f_{(1-2-1)}(x)$ is non zero arbitrarily narrow. This effect is demonstrated by the function plotted in purple, which has the same w_i parameters as the function plotted in red, but is much narrow as its b_i parameter are varied to $b_1 = 90$ and $b_2 = 120$.

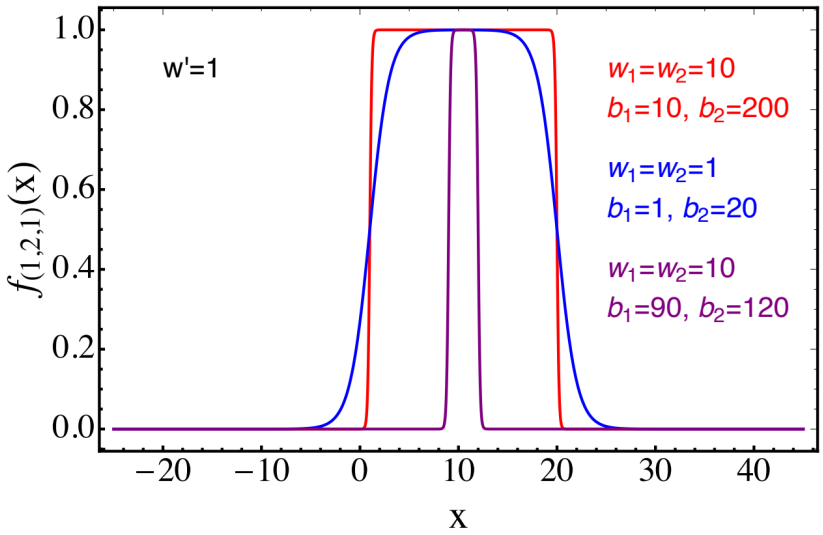


Figure 4.4: Illustration of how a FNN with architecture $(1, 2, 1)$ can reproduce either a rectangular function, shown in red, a function with a smooth peak, shown in purple, or an intermediate case, shown in blue. The x axis acts as the one dimensional input variable, while the y axis represents the output of the network, $f_{(1,2,1)}(x)$ [65].

Thus, with only three neurons, two in the hidden layer and one output neuron, the network $f_{(1,2,1)}$ is able to reproduce a smooth peak (purple), a broad plateau (blue) or a rectangular function (red). By combining several functions of this kind, i.e. building a juxtaposition of rectangular functions, it is possible to approximate any real function to arbitrary precision [62–64].

If the family of real functions defined in the beginning of this chapter, $\mathcal{F}$, is chosen to be comprised of piece-wise functions instead of neural networks $f_{\vec{a}}(x, \mathbf{w})$, then the output will

be a binned histogram, where each bin α has an independent free parameter w_α associated to it. A straightforward disadvantage to binned histograms includes the arbitrariness of the binning. Piece-wise functions are discontinuous in nature and, thus, can oscillate rapidly. Due to statistical fluctuations, the best fit parameters to the data can result in histograms with large gradients between adjacent bins, which randomly assume positive and negative values depending on the choice of binning. If these gradients are present, binned histograms do not faithfully reproduce the true underlying distribution. This issue is often addressed through optimization algorithms which automatically set variable bin widths, where the bins are set to be as narrow as possible while still being compatible with experimental resolution. Neural networks, on the other hand, are continuous smooth functions and do not suffer from this issue.

Moreover, it was shown that a neural network with a single hidden layer, with architecture $(1, 2, 1)$ and, therefore, with 7 free parameters as per Eq. 4.3, can reproduce any real function containing arbitrarily narrow peaks. Using the binned histogram method, approximating similar features would reacquire a large number of bins. The number of free parameters would then be given by the number of bins minus one. Additionally, as the dimensionality of the input variable x increases, the number of necessary bins to approximate the given function increases exponentially. On the other hand, FNNs can approximate the same distribution with a much lower number of free parameters, only depending on the number of neurons in the hidden layer [62]. For an higher dimensional input variable x, only two more hidden neurons are necessary per additional dimension [63]. Similarly to the case presented above, the outputs from all hidden neurons are sent to a final neuron in the last layer, which then generates the multi-dimensional equivalent of a rectangular function.

4.3 New Physics Learning Machine

As explained in the previous chapter, MUSiC analysis save kinematic information in binned histograms, employing a Region of Interest Scanning algorithm to find deviations between measured data and SM simulation. This section introduces NPLM [65, 66], a newly proposed machine learning algorithm, as an alternative method to find such deviations. For the advantages of neural networks over binned histograms as function approximants, please refer to Section 4.2.

The construction and statistical foundations of the NPLM algorithm are first explained in the subsection 4.3.1. The construction of a loss function from a maximum likelihood hypothesis test is then explained in 4.3.2. Finally, the dependence of the NPLM performance on the non-trainable hyper-parameters is briefly discussed in 4.3.3.

4.3.1 Statistical Foundations of NPLM

Consider an experiment consisting of repeated measurements of a d-dimensional random variable $x \in X$, where X refers to a specified phase-space region. Take $\mathcal{D} = \{x_1, ..., x_{\mathcal{N}_\mathcal{D}}\}$ to be the set of observed values of x, with $\mathcal{N}_\mathcal{D}$ representing the total number of independently performed measurements. Each value of x follows an unknown underlying probability density function $n(x|T)$, where T stands for the *True Hypothesis*. Like any a statistical hypothesis test, NPLM aims to find an appropriate hypothesis that fits the data sufficiently well, i.e. an hypothesis that, given a set of measurements, cannot be disregarded. If one has prior knowledge

about the data distribution through a *Reference Hypothesis* R, then R can be compared to a given *Alternative Hypothesis* H_i, $i \in \mathbb{N}$, to check which of the two is most favoured by the data. In the case that $\mathcal{R}$ is a weighted data set, the total number of *weighted* events $\mathcal{N}_\mathcal{R}$ is given by

$$\mathcal{N}_\mathcal{R} = \sum_{i=1}^{N_\mathcal{R}} w_i \tag{4.13}$$

where w_i is the weight of the i-*th* event. Additionally, let $n(x|R)$ denote the normalized probability density function given by

$$n(x|R) = N(R)P(x|R) \quad \text{with} \quad N(R) = \int dx\, n(x|R) \quad \text{and} \int dx\, P(x|R) = 1, \tag{4.14}$$

where $N(R)$ corresponds to the expected observed events, as predicted by R, and $P(x|R)$ is the unnormalized probability density function. The true number of observations $\mathcal{N}_\mathcal{D}$ from the data set $\mathcal{D}$ is then interpreted as being thrown from a Poisson distribution with mean $N(R)$.

In the case of new physics searches at the LHC, R refers to SM prediction, while $\{H_i\}$ is the set of possible BSM theories. The number of expected events, in a phase-space region X, is then given by the total cross section in X, σ_X, multiplied by the integrated luminosity $\mathcal{L}$,

$$N(R)_X = \sigma_X \cdot \mathcal{L}. \tag{4.15}$$

In dedicated analysis, an alternative hypothesis $H \in \{H_i\}$ is proposed *a-priori* according to the theory model being studied. The data is carefully analysed in the restricted phase-space region where R and H are predicted to be most discrepant. However, in model-independent analysis, such as MUSiC, H is left undefined and the phase-space region is minimally restricted. For simplicity, let the subscript X be dropped in the following derivations.

Since most LHC events are known to be in agreement with the SM, the discrepancy between R and any of H can present itself in two ways: either as a large discrepancy in a low probability phase-space region[2]; or as small correlated discrepancies over a large region. Dedicated searched are designed to be sensitive the first scenario, where a signal would be easily detectable. That is, of course, if the proposed BSM model is the correct one. Model-independent searches are designed to be sensitive to both scenarios, where the sensitivity to specific signals is reduced in favour of increased sensitivity to different signals at once.

The foundational postulate of NPLM is the construction of the event distribution $n(x|\mathbf{w})$, corresponding to the alternative hypothesis $H_\mathbf{w}$, as a local re-scaling of $n(x|R)$. The re-scaling is defined by an exponential function parametrized by the single output of a neural network $f_{\vec{a}}(\cdot;\mathbf{w})$, as defined in Section 4.1. Mathematically, the postulate is expressed as

$$n(x|\mathbf{w}) = e^{f_{\vec{a}}(x;\mathbf{w})} n(x|R), \tag{4.16}$$

where $\mathbf{w}$ are the network trainable hyper-parameters and $\vec{a}$ defines the architecture of the network. Moreover, given the observed data set $\mathcal{D}$, and assuming the data points $x \in \mathcal{D}$ were thrown from the probability distribution $P(x|R)$, the likelihood of the reference hypothesis R is given by

$$\mathcal{L}(R|\mathcal{D}) = \frac{N(R)^{\mathcal{N}_\mathcal{D}}}{\mathcal{N}_\mathcal{D}!} e^{-N(R)} \prod_{x\in\mathcal{D}} P(x|R) = \frac{e^{-N(R)}}{\mathcal{N}_\mathcal{D}!} \prod_{x\in\mathcal{D}} n(x|R). \tag{4.17}$$

[2]Otherwise, if the signal would be in a high-probability region, one would have already found signs of new physics.

Similarly, if one assumes that the data set $\mathcal{D}$ is thrown from the alternative hypothesis $H_{\mathbf{w}}$, the corresponding likelihood of $H_{\mathbf{w}}$ is given by

$$\mathcal{L}(H_{\mathbf{w}}|\mathcal{D}) = \frac{e^{-N(\mathbf{w})}}{\mathcal{N}_{\mathcal{D}}!} \prod_{x\in\mathcal{D}} n(x|\mathbf{w}), \tag{4.18}$$

where $N(\mathbf{w})$ is the number of expected data events as predicted by the alternative hypothesis $H_{\mathbf{w}}$. More explicitly, $N(\mathbf{w})$ is given by

$$N(\mathbf{w}) = \int dx\; n(x|\mathbf{w}) = \int dx\; e^{f_{\vec{a}}(x;\mathbf{w})} n(x|R). \tag{4.19}$$

Given the parametrization in Eq. 4.16, the most suitable statistical test to determine which hypothesis is most favoured by the data set $\mathcal{D}$ is defined by the Neyman–Pearson construction [67], based on a Maximum Likelihood log ratio test. This construction leads to the test statistic $t(\mathcal{D})$ defined by

$$t(\mathcal{D}) = 2\log \frac{\max_{\mathbf{w}}[\mathcal{L}(H_{\mathbf{w}}|\mathcal{D})]}{\mathcal{L}(R|\mathcal{D})} = 2\log\left[\frac{e^{-N(\hat{\mathbf{w}})}}{e^{-N(R)}} \prod_{x\in\mathcal{D}} \frac{n(x|\hat{\mathbf{w}})}{n(x|R)}\right], \tag{4.20}$$

where the maximized likelihood of the alternative hypothesis is denoted with hatted hyper-parameters, i.e. $\max_{\mathbf{w}}[\mathcal{L}(H_{\mathbf{w}}|\mathcal{D})] = \mathcal{L}(H_{\hat{\mathbf{w}}}|\mathcal{D})$. The final expression for $t(\mathcal{D})$ is derived by substituting the likelihoods by the expressions given in Eq. 4.17 and Eq. 4.18. Note that $n(x|\hat{\mathbf{w}})$ and $N(\hat{\mathbf{w}})$ are defined analogously to Eq. 4.16 and Eq. 4.19, respectively. The concepts and notation introduced so far are summarized in Fig. 4.5.

Lastly, it is important to mention that, according to Wilks Theorem [68], the test statistic $t(\mathcal{D})$ will asymptotically follow a χ^2 distribution when the data is distributed according to the reference hypothesis[3], i.e. when $H_{\mathbf{w}} = R$. Note that the corresponding χ^2 function has the same number of degrees of freedom as the maximum log-likelihood ratio. In this context, these are the number of trainable parameters $\mathbf{w}$.

	Distributions
$n(x\|\mathrm{R})$	Distribution of the variable x in the reference model R
$n(x\|\mathrm{T})$	True distribution of x
$n(x\|\widehat{\mathbf{w}})$	Distribution of x estimated by the Neural Network (NN)
	Events
$N(\mathrm{R})$	Number of expected events in the reference model R
$N(\widehat{\mathbf{w}})$	Number of events in the data estimated by the NN
	Normalization
$\int n(x)dx = N$	$n(x)$: Events distribution
$\int P(x)dx = 1$	$P(x)$: Probability distribution

Figure 4.5: Summary of the introduced concepts relating to the *Reference Hypothesis* R, the *True Hypothesis* T and the optimized *Alternative Hypothesis* $H_{\hat{\mathbf{w}}}$, accompanied by the corresponding notation. Adapted from [65].

[3]This is true for the case of infinite statistics.

4.3.2 Constructing the Loss Function

To employ NPLM, the maximization of the likelihood log ratio in Eq. 4.20 must first be translated into a minimization of a loss function. Note that it is possible to rewrite Eq. 4.20 such that it reads

$$t(\mathcal{D}) = -2\text{Min}_{\{\mathbf{w}\}}[N(\mathbf{w}) - N(R) - \sum_{x\in\mathcal{D}} f_{\vec{a}}(x;\mathbf{w})]. \tag{4.21}$$

However, to compute $t(\mathcal{D})$, the following steps are necessary:

(i) **Estimating $N(\mathbf{w})$**
Note that it is not possible to determine $N(\mathbf{w})$ by computing the integral in Eq. 4.19, since the analytical form of the probability density function $n(x|R)$ is unknown. To solve this, one estimates $N(\mathbf{w})$ using Monte Carlo methods by setting

$$N(\mathbf{w}) = \frac{N(R)}{\mathcal{N}_{\mathcal{R}}} \sum_{x\in\mathcal{R}} e^{f_{\vec{a}}(x;\mathbf{w})}. \tag{4.22}$$

(ii) **Expressing $t(\mathcal{D})$ as a Loss Function**
Inputting the above definition of $N(\mathbf{w})$ into Eq. 4.20, the test statistic expression now becomes

$$t(\mathcal{D}) = -2\text{Min}_{\{\mathbf{w}\}} \left[\frac{N(R)}{\mathcal{N}_{\mathcal{R}}} \sum_{x\in\mathcal{R}} (e^{f_{\vec{a}}(x;\mathbf{w})} - 1) - \sum_{x\in\mathcal{R}} f_{\vec{a}}(x;\mathbf{w}) \right]. \tag{4.23}$$

It is now possible to define the loss function as

$$L[f] \equiv \frac{N(R)}{\mathcal{N}_{\mathcal{R}}} \sum_{x\in\mathcal{R}} (e^{f_{\vec{a}}(x;\mathbf{w})} - 1) - \sum_{x\in\mathcal{R}} f_{\vec{a}}(x;\mathbf{w}). \tag{4.24}$$

Introducing the target variable y with $y = 0$ and $y = 1$ for the reference and the toy data sets, respectively, the loss function can be rewritten as

$$L[f_{\vec{a}}(x,\mathbf{w})] = \sum_{(x,y)} \left[(1-y)\frac{N(R)}{\mathcal{N}_{\mathcal{R}}} (e^{f_{\vec{a}}(x,\mathbf{w})} - 1) - y f_{\vec{a}}(x,\mathbf{w}) \right]. \tag{4.25}$$

Minimizing L with respect to the neural network parameters $\mathbf{w}$ is therefore a standard supervised learning process.

(iii) **Divergence of the Loss Function**
A final point that must be addressed is the fact that L is unbounded from below, i.e. Eq. 4.25 will tend to $-\infty$ if L diverges in any localized region of $x \in \mathcal{D}$. This issue is solved by introducing a **Weight Clipping parameter**, which sets a maximum value to the weights and bias of the network. The value of the weight clipping parameters must be carefully chosen depending on the data sets used.

Note that once the network has been properly trained, and $\hat{\mathbf{w}}$ have been found through the minimization of L, $f(x;\hat{\mathbf{w}})$ is the best approximant to the true data distribution $n(x|T)$, i.e. $f(x;\hat{\mathbf{w}}) \simeq \log\left[\frac{n(x|T)}{n(x|R)}\right]$.

4.3.3 Hyper-parameter Dependence

A detailed study on the performance of NPLM based on different hyper-parameters is found in Chapter 4.4 of the original paper [65]. Here, a summary is presented of the conclusions from the authors. The hyper-parameters that must be set before training include:

- **Learning rate** (defined in Eq. 4.10)
  The authors concluded that a learning rate higher than 10^{-3} does not allow the loss function to converge, as L keeps oscillating as training proceeds. Moreover, a lower value does not improve the performance of the algorithm. Thus, throughout this thesis the learning rate is fixed at 10^{-3}.

- **Training rounds**
  The authors verified that at least 1.5×10^5 training rounds are needed for a good NPLM performance. Moreover, it was also observed that increasing the training rounds to 1.5 million, effectively increasing the necessary computer power, does not significantly improve the performance of the algorithm.

- **Architecture** (as constructed in Section 4.1)
  The architecture of the network should be chosen as to improve the agreement between the distribution of the test statistic and that of the corresponding χ^2 function, given that the toy data set has been generated according to the reference hypothesis R. This agreement is quantified by the authors through the Kolmogorov-Smirnov test [69]. However, there is no concrete method to define an appropriate architecture, as this agreement is highly dependent on the problem. It is noteworthy to mention the increased computational resources required for larger architectures, as more training rounds are necessary to achieve a reasonable agreement to the χ^2 function.

- **Weight clipping**
  In principle, a large enough weight clipping can be set for the problem at hand, such that it does not restrict the flexibility of the neural network to vary its weights and bias values. The problem with this approach is that it may lead to unstable training. In that case, the test statistic distribution keeps changing as the training proceeds. On the other hand, if the weight clipping value is too low, the network does not have enough flexibility, resulting in low t values and missing the χ^2 behaviour. Instead, a weight clipping sweep must be performed, where one trains the same network over a range of weight clipping values and evaluates which ones provides the best χ^2 agreement. A full discussion of the weight clipping selection can be found in Section 3.1 of [66].

Once the hyper-parameters have been set, the same network can be used to test the sensitivity to alternative hypothesis and, if found to be sensitive enough, it can be further employed to compare the reference model to real data.

A schematic illustration of the NPLM workflow is found in Fig. 4.6. The first step is to follow the workflow shown in blue. One must first create the reference sample $\mathcal{R}$ and generate a toy sample $\mathcal{D}_\mathcal{R}$ following the reference hypothesis. This is done through the rejection sampling algorithm [70–72], also known as the *Hit or Miss* method, shown in the yellow box. A brief explanation of the method is found in the interlude below.

Interlude – *Hit or Miss Method*

1. Sample an event $x \in \mathcal{R}$ with weight w_x, and a random number $u \in [0, 1]$.
2. Define the variable w_f such that the probability of accepting an event with w_x is $r(w_x) = w_x/w_f$. Notice that w_x must not be negative, as this will create negative probabilities. Additionally, w_f must be lower than the maximum weight, labelled, w_{max}, such that the probability of accepting the corresponding event is smaller than 1. Therefore, events with negative weights must be previously removed from $\mathcal{R}$.
3. Compare u with $r(w_x)$ and accept the event as a toy event if $r \geq u$, and reject the event otherwise.
4. Repeat the procedure until enough toy events have been selected.

Both samples, $\mathcal{R}$ and $\mathcal{D}_{\mathcal{R}}$, then serve as inputs to NPLM, which will output a test statistic $t(\mathcal{D}_{\mathcal{R}})$, calculated through the minimum loss value found by the network. To construct the test statistic distribution $P(t|R)$ more toy samples must be generated and NPLM must be trained multiple times. In the illustration, this is shown by the light grey box, suggesting that this workflow must be performed $1\ 000$ times for sufficient statistics. After this procedure is complete, it is possible to compare the distribution of the test statistic with the corresponding χ^2 function. If these are not in sufficient agreement, the network hyper-parameters must be adjusted (e.g. try a different architecture or weight clipping parameter.). Only then, it is possible to move to the next workflow, shown in green.

The green workflow is used to perform sensitivity studies on the chosen neural network, i.e. the hyper-parameters are those found to be optimal once the workflow in blue has been completed. This time, the same reference sample $\mathcal{R}$ is used, but the toy samples, labelled by $\mathcal{D}_{H_w}$, will include additional signal events. As before, $1\ 000$ toy signal samples should be used to construct the test statistic distribution $t(\mathcal{D}_{H_w})$. Since the toy sample does not follow the reference hypothesis anymore, it is not expected that the test statistic distribution follows a χ^2 function. Indeed, the further the distribution is from the corresponding χ^2, the easier it is for the network to distinguish between toy samples generated according to the reference or to an alternative hypothesis.

Lastly, the workflow in orange compares the reference data set $\mathcal{R}$ with the single data set of measured data $\mathcal{D}_{\wr\lfloor\int}$. The hyper-parameters of the network should be the same as for the green workflow. Note that the output of the network is now a single test statistic t_{obs}, since only one measured data set is used. The test statistic can then be used to calculate the corresponding observed p-value through

$$p_{obs} = \int_{t_{obs}}^{\infty} dt\ P(t|R). \tag{4.26}$$

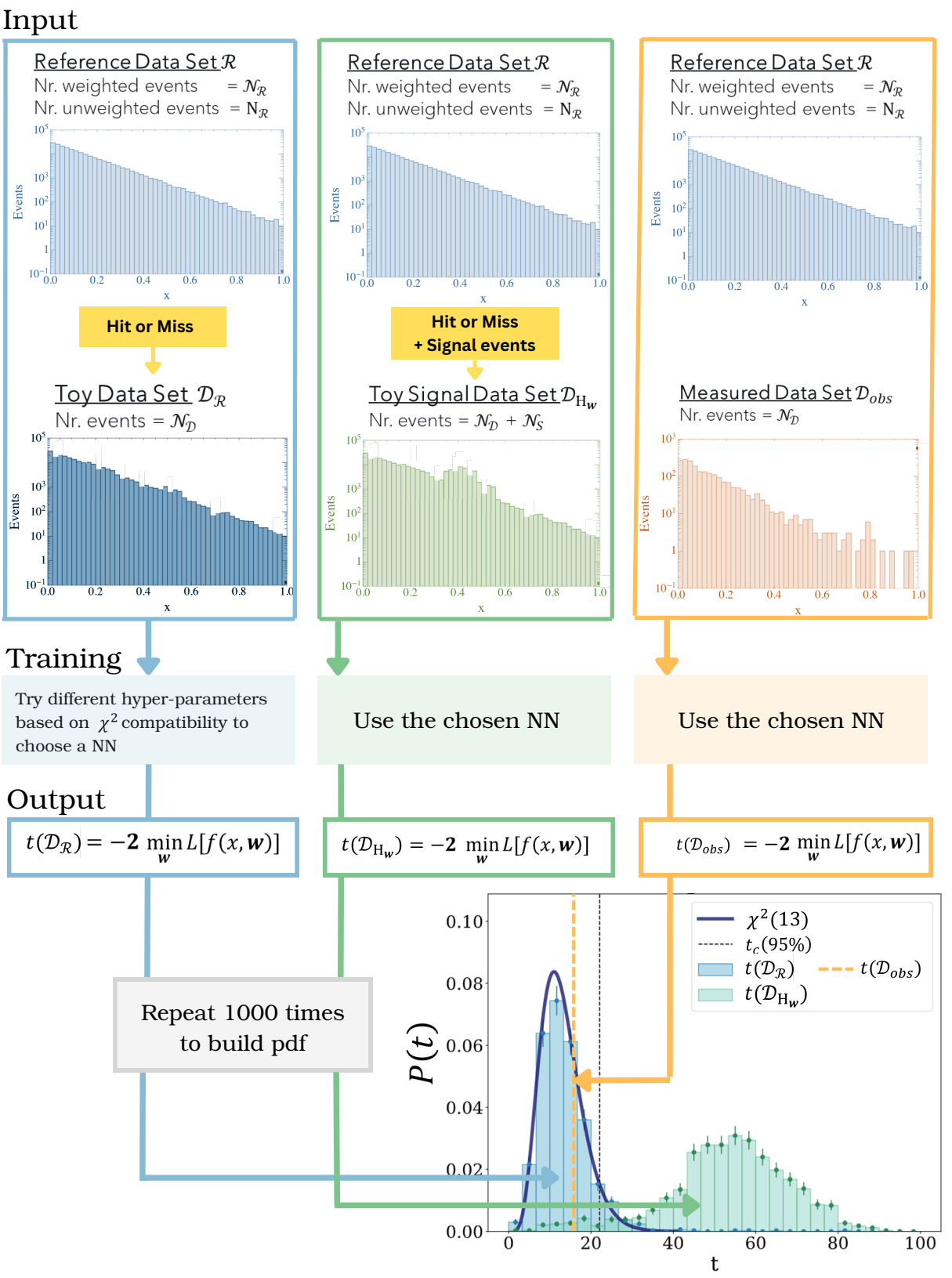


Figure 4.6: Schematic of the NPLM workflow. The blue workflow regards the comparison between the reference data set $\mathcal{R}$ and a toy data set $\mathcal{D}_{\mathcal{R}}$, thrown from the reference hypothesis R using the *Hit or Miss* procedure. Similarly, the green workflow compares $\mathcal{R}$ to a toy data set with added signal events $\mathcal{D}_{\mathcal{S}}$. Both these workflows are repeated 1000 times to build the probability distribution functions of the test statistic. Lastly, the orange workflow compares $\mathcal{R}$ with the observed data set $\mathcal{D}_{obs}$. The outputted test statistic $t(\mathcal{D}_{obs})$ can then be used for the p-value calculation.

Chapter 5

NPLM Results

"When I first got into it, nobody knew what it was that we were doing.
It was like the Wild West."
– Margaret Hamilton

The present chapter validates NPLM on a simple case study, where weights are applied to the input data points. In order to approximate MUSiC data sets, different weight distributions and varying percentages of negative weights are considered. Moreover, the case study introduces the statistical power test as a measure of hyper-parameter suitability to the problem at hand. This extends the work presented in [65], where the hyper-parameter suitability was only evaluated through the Kolmogorov–Smirnov test. Finally, Section 5.2 presents NPLM results on the MUSiC event class $1e + 1\mu + p_T^{miss}$, with 2017 MC data. To test NPLM sensitivity to both resonant and non-resonant deviations from the SM, a cross section variation is introduced, independently, to Higgs processes, Multi-Boson processes and top quark pair processes, with $M_{t\bar{t}>700}$ GeV. These processes were chosen for their relevance to the particular event class considered. As a final remark, note that, for both the case study and the MUSiC implementation, NPLM is employed with the restrictive conditions of single-dimensional inputs and absence of systematic uncertainties in the data sets.

5.1 Results on a Simple Case Study

Before running the NPLM algorithm on MUSiC data, it is first necessary to establish its reliability on weighted data sets. In the present section, NPLM is tested on a uni-variate case study, aimed at replicating the features of a total transverse momentum distribution. Nuisance parameters are not considered and the null hypothesis R is assumed to be a perfect description of the observed data. This section then explores the sensitivity of NPLM regarding the:

1. value of the weight clipping parameter;
2. probability distribution function of the weights;
3. relative amount of negatively weighted events.

5.1.1 Reference Samples

The reference model considered for this study is a falling exponential distribution

$$n(x|R) = e^{-x}, \tag{5.1}$$

where $x \in [0, 25\,000]$ represents the transverse momentum variable. Four reference samples are drawn from the distribution in Eq. 5.1, $\mathcal{R}_i = \{x_j\}$, with $i \in \{1, 2, 3, 4\}$ and $j \in \{1, ..., N_{\mathcal{R}}\}$, with $N_{\mathcal{R}} = 200\,000$. For each sample, the event weights w_j follow a different weight distribution $n(w)$, but all weighed events sum to $\mathcal{N}_{\mathcal{R}} = 2\,000$. This condition ensures the reference samples are sufficiently large to avoid double counting when generating unweighted toy data sets via the *Hit or Miss* method. To evaluate the impact of the weight distribution function in the reference data set, the first two samples, $\mathcal{R}_1$ and $\mathcal{R}_2$, are weighted by a Gaussian function $f_N(\mu_1, \sigma_1)$ (Eq. 5.2) and the last two samples, $\mathcal{R}_3$ and $\mathcal{R}_4$, are weighted by a Gaussian mixture function.

$$f_N = \frac{1}{\sqrt{2\pi{\sigma_1}^2}}\, e^{\frac{(w_1-\mu_1)^2}{2{\sigma_1}^2}} \tag{5.2}$$

In the latter case, the first $20\,000$ weights were drawn from $f_N(\mu_2, \sigma_2)$, the following $150\,000$ weights from $f_N(\mu_3, \sigma_3)$, and the remaining $30\,000$ weights from $f_N(\mu_4, \sigma_4)$. Furthermore, to evaluate the impact of the relative amount of negative weights, the medians and standard deviations were chosen such that the negatively weighted events in $\mathcal{R}_1$ and $\mathcal{R}_3$ account for about 8% of the total weighted events, and in $\mathcal{R}_2$ and $\mathcal{R}_4$ account for about 16%. The selected means and standard deviations are defined in Table 1, together with the final ratio of weighted events for each reference sample. Additionally, the event weight distributions for each sample are plotted in Fig. 5.1.

Finally, it is important to note that large weight values ($w \gg 0$) can be present in MUSiC data sets, which reduces the efficiency of the *Hit or Miss* method. Thus, a maximum weight cut off, w_{max}, is necessary. For the present case study, a value of $w_{max} = 0.06$ was implemented for all reference samples. The relative amount of weights larger than w_{max} is 1%, which does not significantly reduce the reference samples when the corresponding events are discarded from the data set. Note that, after removing negative weights and weights larger than w_{max} from the reference samples, a weight re-scaling is performed, to ensure the remaining weighted events still sum to $2\,000$.

Table 1: Summary table of the four different reference hypothesis used, specifying their corresponding weight distribution functions as well as the percentage of negative weighted events in the data set generated.

Reference Sample	$n(w)$	Parameters	Ratio of weighted events with w<0
$\mathcal{R}_1$	$f_N(\mu_1, \sigma_1)$	$\mu_1 = \sigma_1 = 0.01$	~ 0.083
$\mathcal{R}_2$		$(\mu_1, \sigma_1) = (0.078, 0.01)$	~ 0.160
$\mathcal{R}_3$	$f_N(\mu_2, \sigma_2)$ + $f_N(\mu_3, \sigma_3)$ + $f_N(\mu_4, \sigma_4)$	$(\mu_2, \sigma_2) = (-0.009, 0.01)$ $(\mu_3, \sigma_3) = (0.01, 0.06)$ $(\mu_4, \sigma_4) = (0.05, 0.01)$	~ 0.081
$\mathcal{R}_4$		$(\mu_2, \sigma_2) = (-0.02, 0.01)$ $(\mu_3, \sigma_3) = (0.01, 0.06)$ $(\mu_4, \sigma_4) = (0.05, 0.01)$	~ 0.161

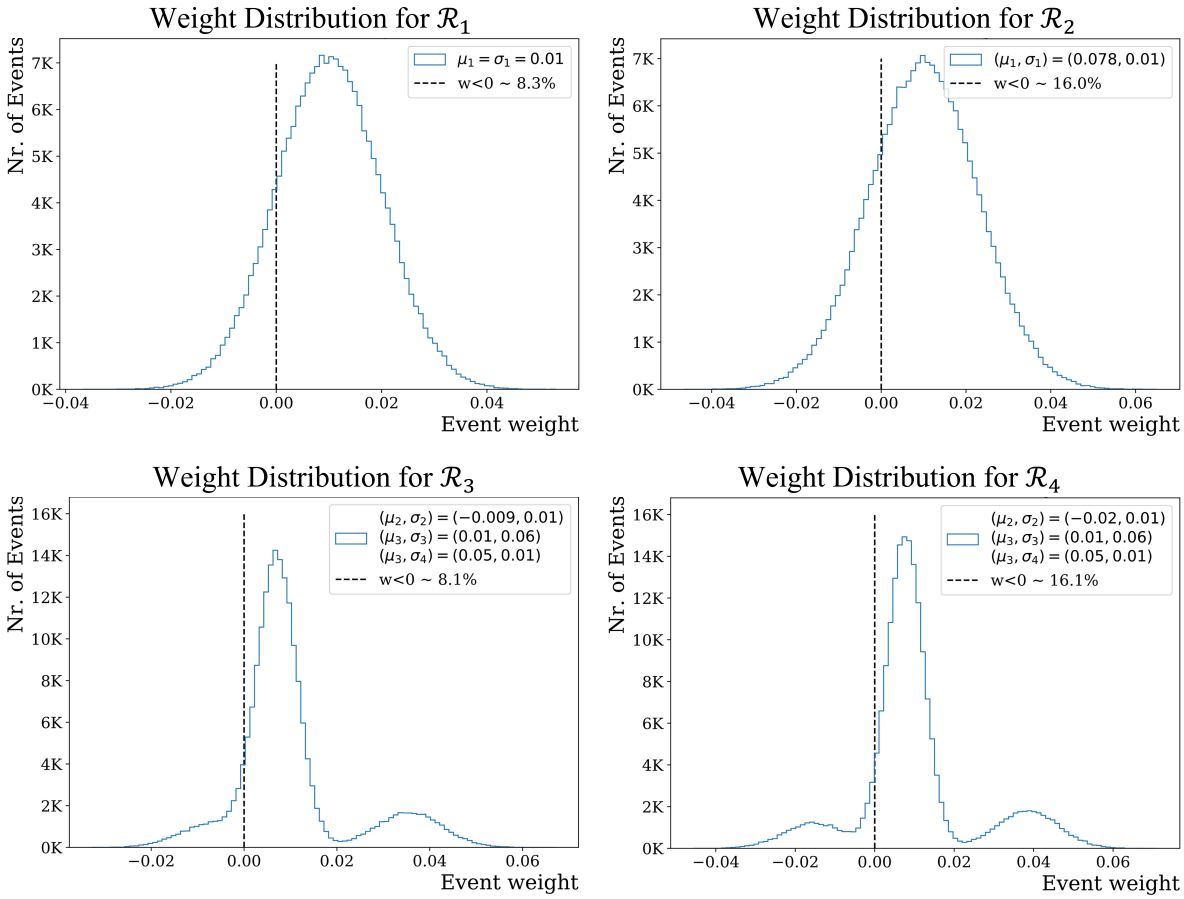


Figure 5.1: Weight distributions for the reference samples $\mathcal{R}_1$ (top left), $\mathcal{R}_2$ (top right), $\mathcal{R}_3$ (bottom left) and $\mathcal{R}_4$ (bottom left). The ratio of negative weighted events to the total weighted events is shown in the legend.

5.1.2 Toy Samples

Let $\mathcal{D}_\mathcal{R}$ label toy data sets generated according to the reference hypothesis R, and let $\mathcal{D}_\mathcal{S}$ label toy data sets generated according to the *alternative hypothesis* $\mathcal{S}$[1]. The alternative hypothesis consists of an added Gaussian signal to the reference hypothesis. This choice results in a probability distribution $P(x|S)$ defined by $P(x|S) = P(x|R) + P(x|S')$ with

$$P(x|S') = e^{-\frac{(x-\bar{x})^2}{2\sigma_s^2}} \quad \text{where} \quad \bar{x} = 5\,000 \text{ and } \sigma_s = 100. \tag{5.3}$$

For $\mathcal{D}_\mathcal{R}$ toy samples, the total number of events $\mathcal{N}_\mathcal{D}$ is drawn from a Poisson distribution with a mean of $\mathcal{N}_\mathcal{R} = 2\,000$. For $\mathcal{D}_\mathcal{S}$ toy samples, a fixed number of 50 signal events were added, such that the total number of events becomes $\mathcal{N}_\mathcal{D} + 50$. Similarly to real data, all toy events are unweighted. Fig. 5.2 shows the histogram representation of the reference sample $\mathcal{R}$ and the toy samples $\mathcal{D}_\mathcal{R}$ and $\mathcal{D}_\mathcal{S}$. The signal events are highlighted by the dashed green box. Recall that NPLM uses individual event information as an input, so these histograms are for illustration purposes only.

[1]The notation for the alternative hypothesis was changed from H to S to clarify that S stands for an hypothesis containing signal events.

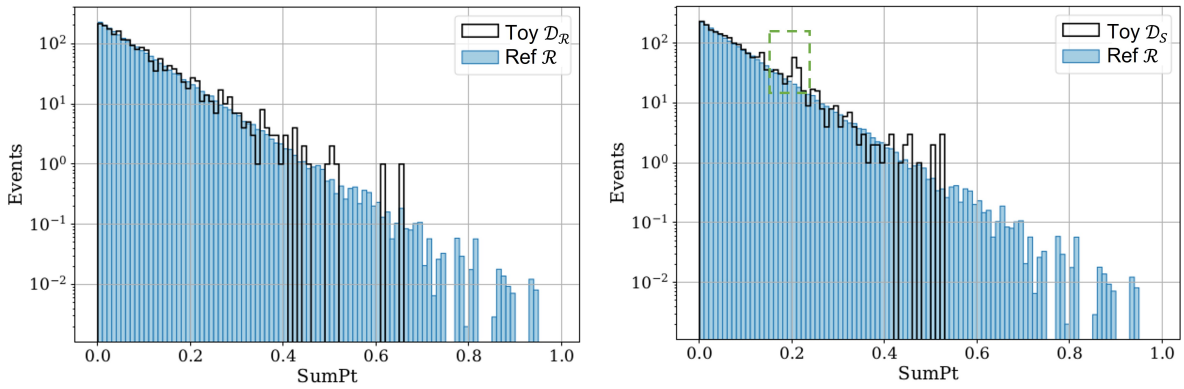


Figure 5.2: Histogram representation of the reference sample $\mathcal{R}$ and the toy sample $\mathcal{D}_\mathcal{R}$ (left) and $\mathcal{D}_\mathcal{S}$ (right), where the signal events are highlighted in the dashed green box. The x axis, representing the sum of the transverse momentum, was normalized such that $x = 1$ corresponds to $x = 25\ 000$.

Note that other alternative hypothesis were not studied as the nature of the chosen alternative hypothesis does not impact the performance of NPLM. This was shown in Section 4.1 of the original paper [65].

5.1.3 NPLM Validation

Following the approach from the authors in [65], the architecture for this simple case study was set to include a single hidden layer with four neurons, i.e. $\vec{a} = (1, 4, 1)$, corresponding to 13 trainable parameters.

The toy samples $\mathcal{D}_\mathcal{R}$ were used as inputs in the network to validate the expected $\chi^2(13)$ behaviour for the test statistic $t(\mathcal{D}_\mathcal{R})$, as explained in Section 4.3.1. Several values of the weight clipping parameter wc were tested, namely $wc \in 1, 25, 50, 75, 100$, to compare the $t(\mathcal{D}_\mathcal{R})$ distribution with the $\chi^2(13)$ function. To reduce statistical uncertainty, $1\ 000$ toy data sets were generated for each experiment. The number of training rounds was always set to $300\ 000$ and the learning rate to 10^{-3}. A weight clipping of 100 was found to be the most appropriate, as it has been already tested in [65]. The training was observed to remain stable for all weight clipping values, expect for $wc = 75$, where some fluctuations were observed even after $150\ 000$ training rounds. Upon finishing training, the weight clipping parameter of 100 resulted in the best percentile agreement between the test statistic values and the percentiles expected from a χ^2 distribution. These results are plotted in Fig. 5.3 and Fig. 5.4

The toy samples $\mathcal{D}_\mathcal{S}$ should lead to a distribution of the test statistic $t(\mathcal{D}_\mathcal{S})$ that deviates from the $\chi^2(13)$ function, since the samples were generated according to an alternative hypothesis. The results are shown in Fig. 5.5-5.8. The test statistic distributions in blue, $t(\mathcal{D}_\mathcal{R})$, result from the comparison between a specific reference sample $\mathcal{R}$ and its corresponding toy sample $\mathcal{D}_\mathcal{R}$. The test statistic distributions in green, $t(\mathcal{S})$, result from the comparison between a the reference sample $\mathcal{R}$ with the toy sample $\mathcal{D}_\mathcal{S}$, generated according to the alternative hypothesis S. Moreover, the ability of the network to distinguish between a toy sample being generated according to either R or S is quantified according to a statistical power test [73]. The critical value t_c was set such that t_c is larger than 95% of test statistics when the toy samples $\mathcal{D}_\mathcal{R}$ are considered. The result of the power test is shown at the top of each plot (*power*), together with

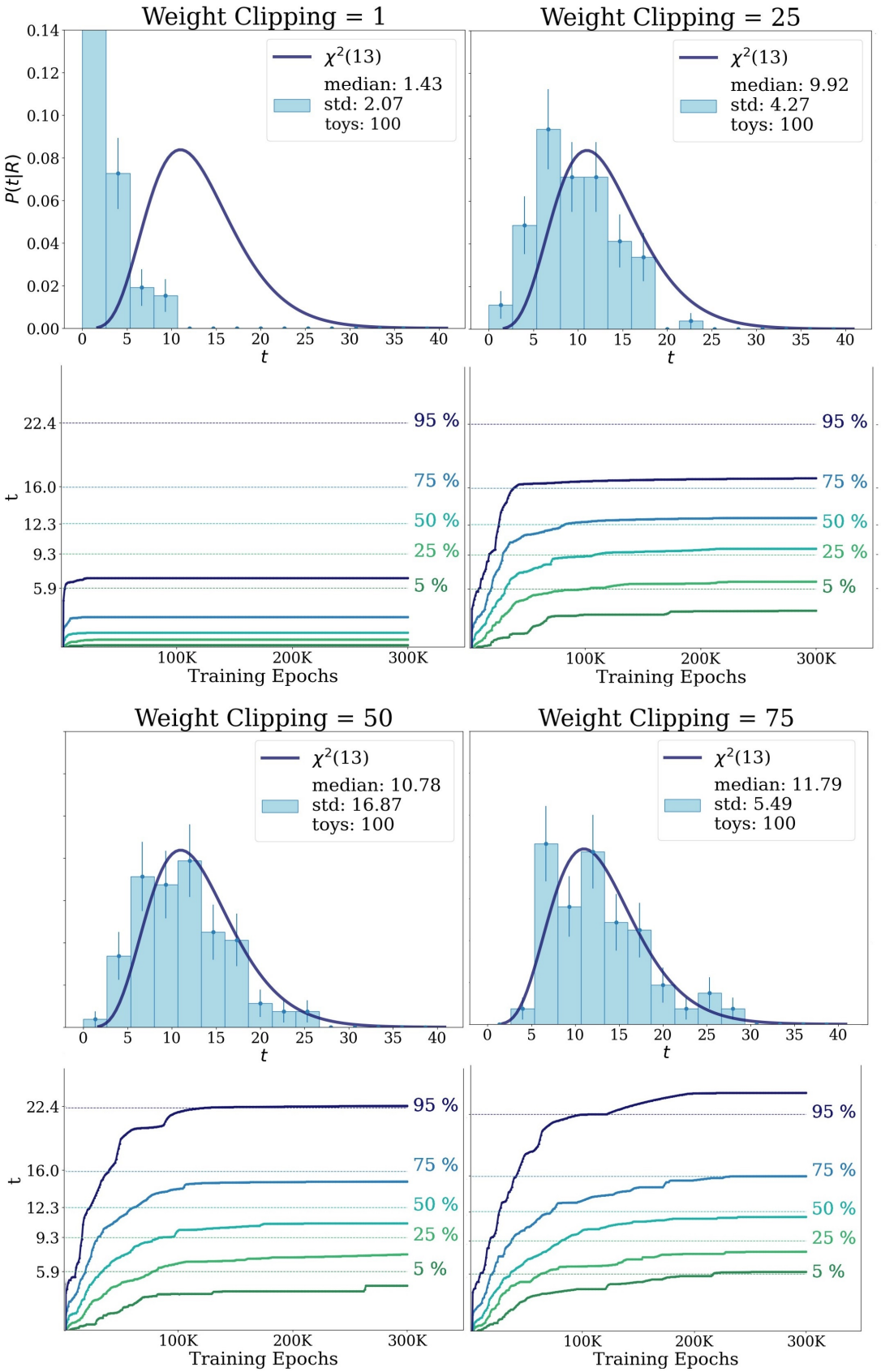


Figure 5.3: Weight clipping study for $wc = 1$, $wc = 25$, $wc = 50$ and $wc = 75$. The training was performed on a network with architecture $\vec{a} = (1, 4, 1)$ using 1 000 toy samples. The learning rate was set to 10^{-3} and 300 000 training rounds were performed.

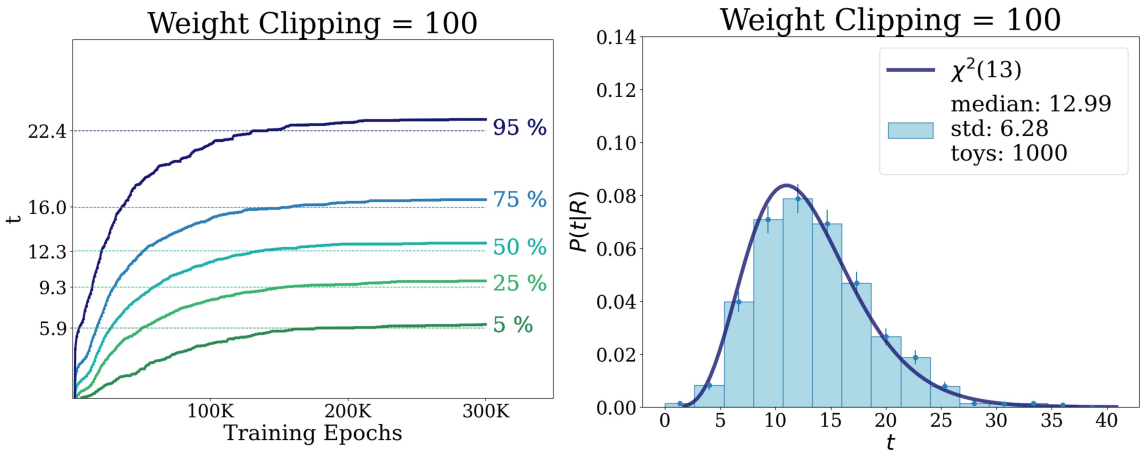


Figure 5.4: Weight clipping study for $wc = 100$. The training was performed on a network with architecture $\vec{a} = (1, 4, 1)$ using $1\,000$ toy samples. The learning rate was set to 10^{-3} and $300\,000$ training rounds were performed.

the corresponding weight clipping parameter applied (*wc*). Additionally, the power test results are summarized in Table 2.

Reference Sample	**Power for wc=5**	**Power for wc=100**
$\mathcal{R}_1$	0.56	0.94
$\mathcal{R}_2$	0.48	0.88
$\mathcal{R}_3$	0.52	0.97
$\mathcal{R}_4$	0.46	0.94

Table 2: Summary of the power test results for each of the four reference samples uses, according to the selected weight clipping parameter.

Conclusions

1. **Value of the weight clipping parameter**
   It was observed that the NPLM algorithm was always more efficient in identifying toy samples which were generated according to the alternative hypothesis S when the weight clipping parameter was set to 100. This was the expected result, as the network has a greater flexibility to vary its weights and bias, resulting in a better recognition of the reference hypothesis R. The level of recognition of R is understood through the agreement between the test statistic distribution $t(\mathcal{D}_{\mathcal{R}})$ with the $\chi^2(13)$ function, quantified by the p-value from the Kolmogorov–Smirnov test. Note that even when the p-value is zero for both weight clippings, the power is significantly higher for a $wc = 100$. Meaning that, even when the distribution $t(\mathcal{D}_{\mathcal{R}})$ is not in great agreement with the expected χ^2 function, it may be in *sufficient* agreement, depending on what is the desired power for the problem at hand. The use of the statistical power for network tuning is therefore a useful addition to the original procedure proposed by the authors in [65].

2. **Probability distribution function of the weights**
   Similar results were observed between data sets with Gaussian distributed weights ($\mathcal{R}_1$ and $\mathcal{R}_2$) and data sets with a more complex weight distribution ($\mathcal{R}_3$ and $\mathcal{R}_4$). For a $wc = 100$, three reference samples resulted in a statistical power > 0.90, with only $\mathcal{R}_2$ showing a slightly lower power at 0.88. This is a preliminary check that the NPLM algorithm should be appropriate for MUSiC data sets, which contain complex weight distributions.

3. **Relative amount of negatively weighted events**
   The power test for $\mathcal{R}_1$ and $\mathcal{R}_2$, with $wc = 100$, shows that increasing the ration of negatively weighted events from 8% to 16% decreases the power of the network from 0.94 to 0.88. However, in the case of a more complex weight distribution, when the ratio of negatively weighted events is increased from 8% in $\mathcal{R}_3$ to 16% in $\mathcal{R}_4$, the decrease in power is only from 0.97 to 0.94. This result can be interpreted as a preliminary check that the number of negatively weighted events, which are common in MUSiC data sets, should not have a significant impact on the NPLM performance, as long as the removal of such events does not cause double counting when applying the *Hit or Miss* method.

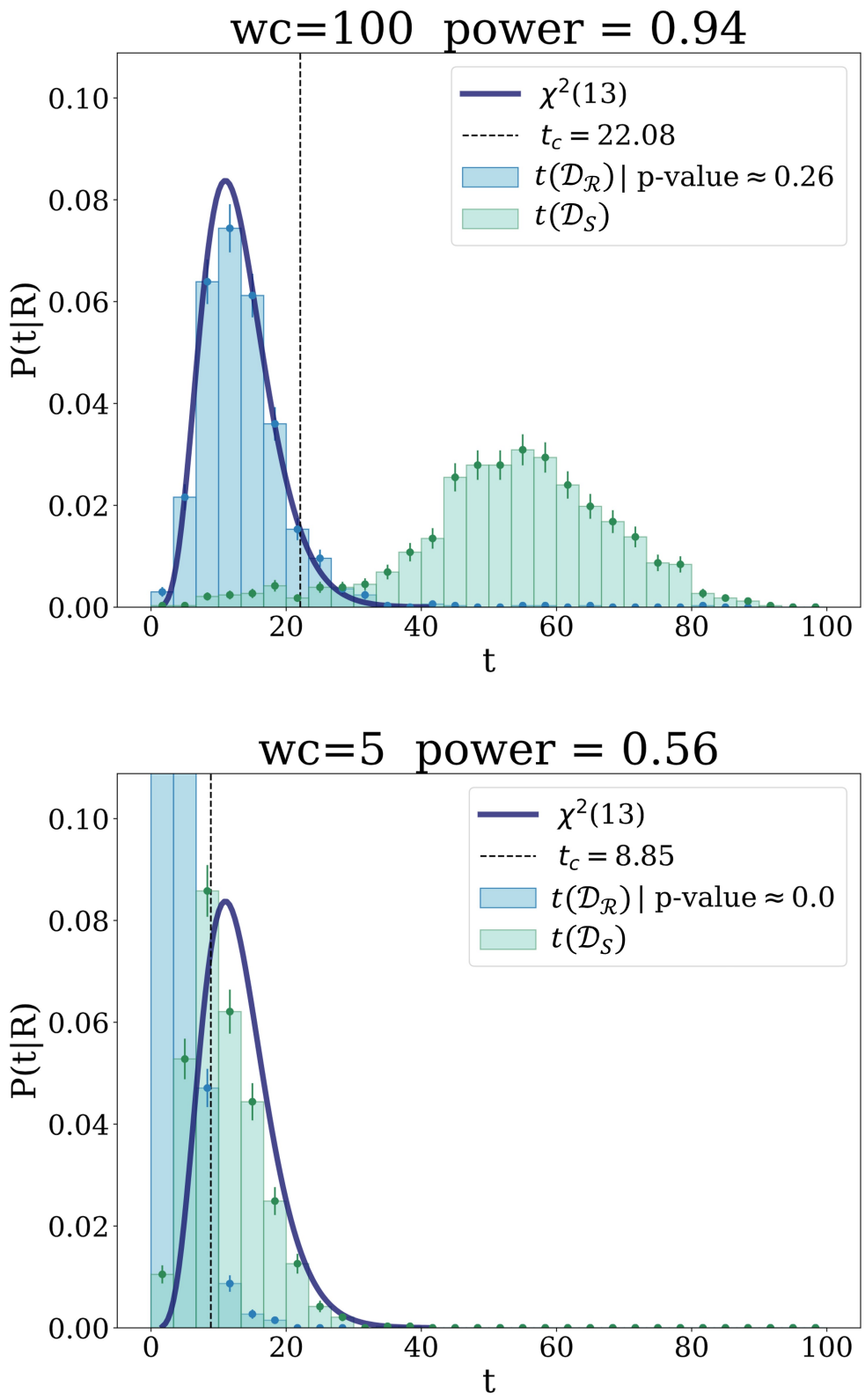


Figure 5.5: Power test results for a weight clipping of 100 (top) or 5 (bottom), when the $\mathcal{R}_1$ reference hypothesis is considered. $\mathcal{D}_\mathcal{R}$ and $\mathcal{D}_S$ stand for toy samples generated according to the R or S hypothesis, respectively. The p-value of the $t(\mathcal{D}_\mathcal{R})$ distribution, when compared to the $\chi^2(13)$ function, plotted as the dark blue line, is also shown. Moreover, t_c stands for the critical test statistic.

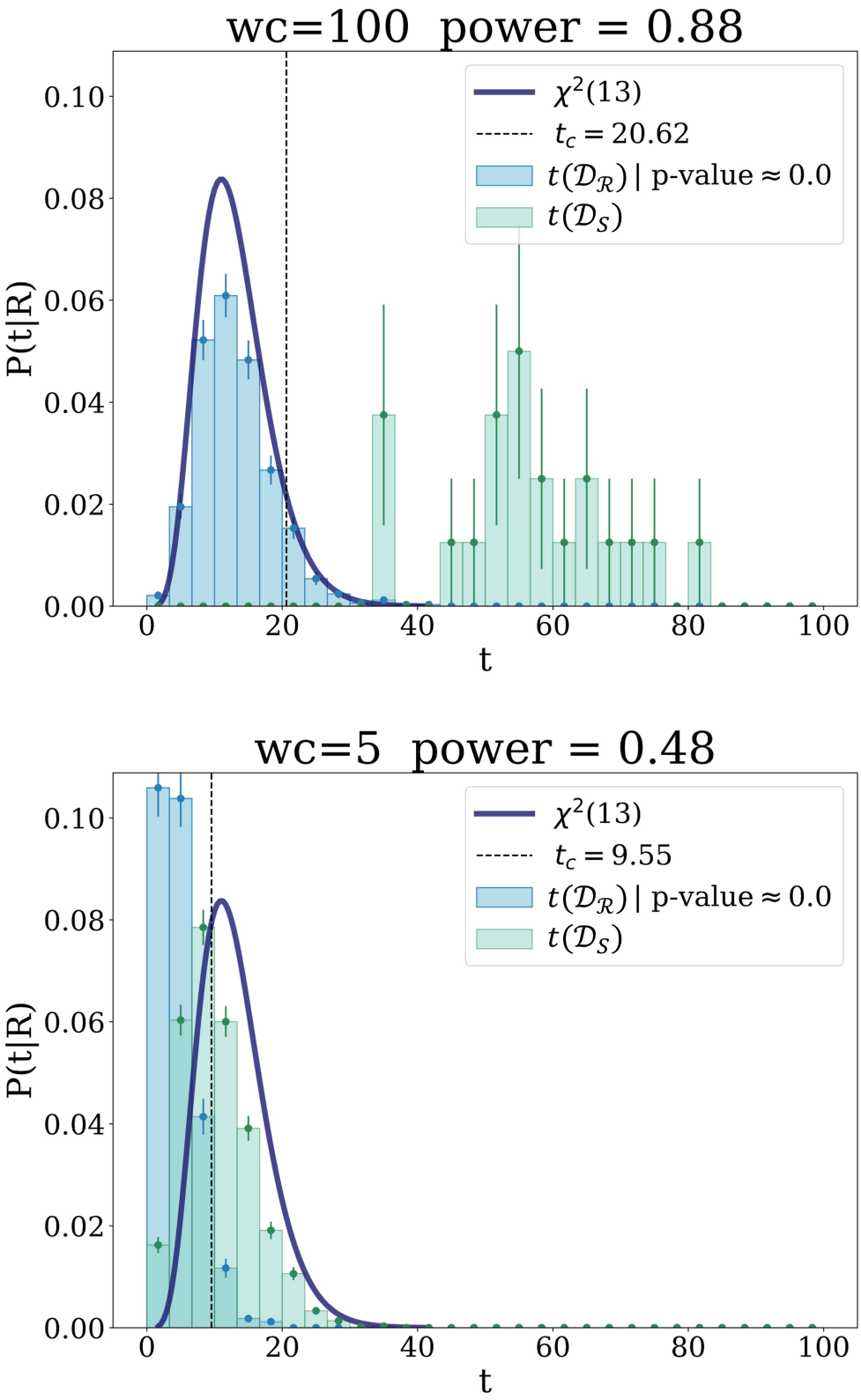


Figure 5.6: Power test results for a weight clipping of 100 (top) or 5 (bottom), when the $\mathcal{R}_2$ reference hypothesis is considered. $\mathcal{D}_\mathcal{R}$ and $\mathcal{D}_\mathcal{S}$ stand for toy samples generated according to the R or S hypothesis, respectively. The p-value of the $t(\mathcal{D}_\mathcal{R})$ distribution, when compared to the $\chi^2(13)$ function, plotted as the dark blue line, is also shown. Moreover, t_c stands for the critical test statistic.

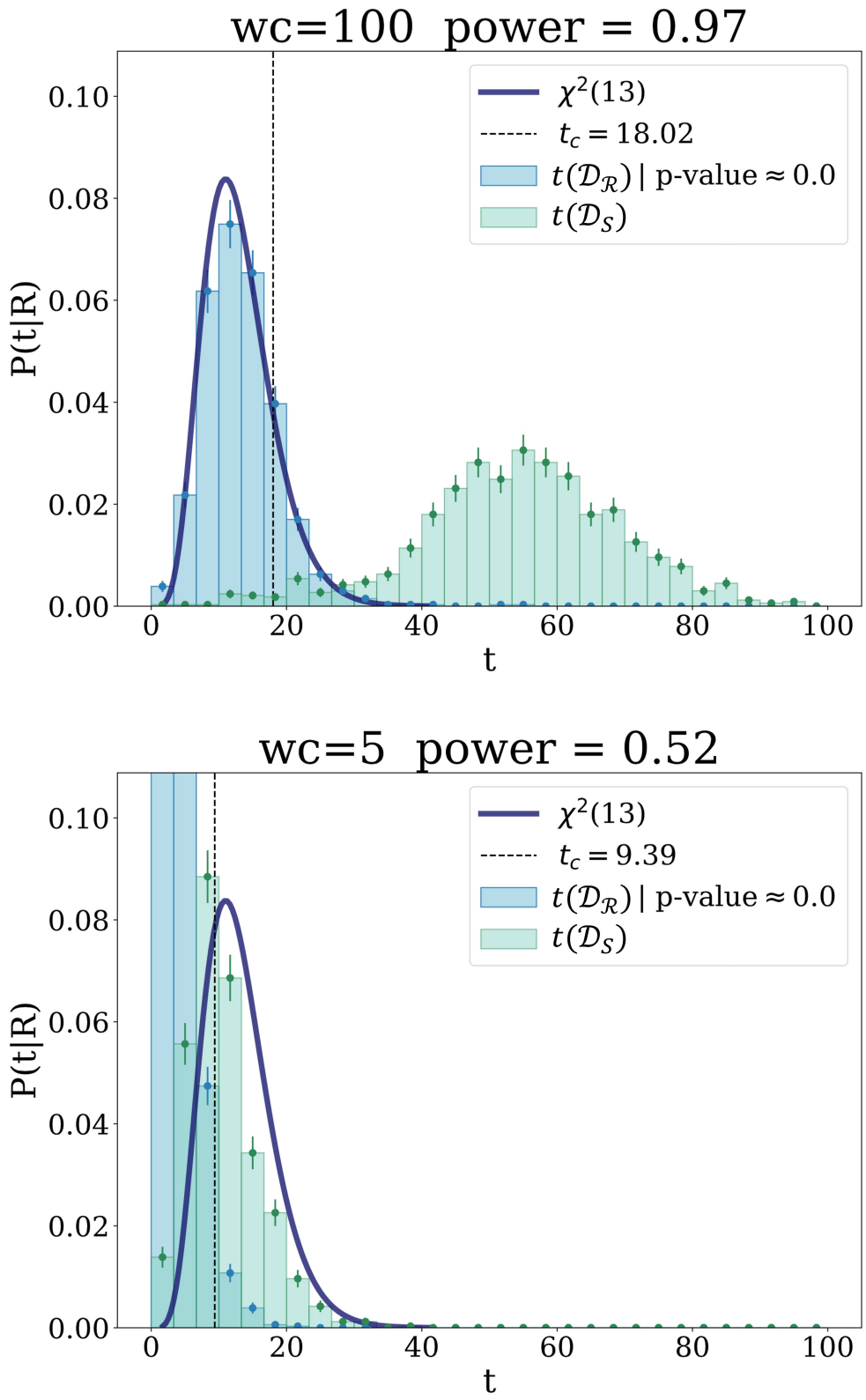


Figure 5.7: Power test results for a weight clipping of 100 (top) or 5 (bottom), when the $\mathcal{R}_3$ reference hypothesis is considered. $\mathcal{D}_\mathcal{R}$ and $\mathcal{D}_S$ stand for toy samples generated according to the R or S hypothesis, respectively. The p-value of the $t(\mathcal{D}_\mathcal{R})$ distribution, when compared to the $\chi^2(13)$ function, plotted as the dark blue line, is also shown. Moreover, t_c stands for the critical test statistic.

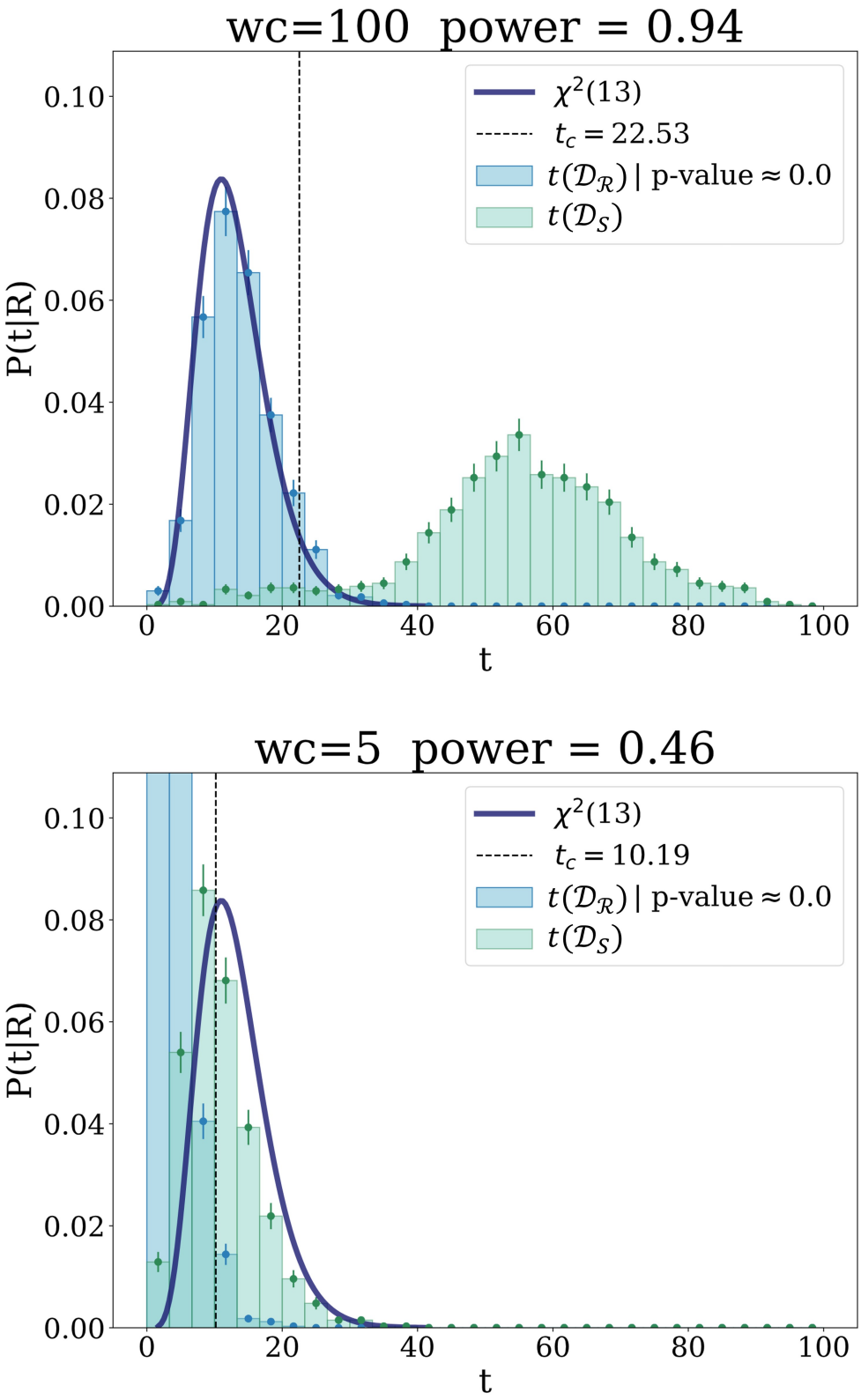


Figure 5.8: Power test results for a weight clipping of 100 (top) or 5 (bottom), when the $\mathcal{R}_4$ reference hypothesis is considered. $\mathcal{D}_\mathcal{R}$ and $\mathcal{D}_S$ stand for toy samples generated according to the R or S hypothesis, respectively. The p-value of the $t(\mathcal{D}_\mathcal{R})$ distribution, when compared to the $\chi^2(13)$ function, plotted as the dark blue line, is also shown. Moreover, t_c stands for the critical test statistic.

5.2 Results on MUSiC Data Sets

5.2.1 Reference Sample

In this section, the NPLM algorithm is applied to the event class $1e + 1\mu + p_T^{miss}$ for 2017 Monte Carlo data. Let $\mathcal{R}$, standing for reference sample, label the set of events which contribute to the class. The corresponding SM process groups of such events are shown in Fig. 5.9, plotted from information contained in the corresponding event class HDF5 file, created during the *Conversation* step (Section 3.5). The labels in the legend refer to the Monte Carlo generated processes specified in Table 1. The total number of weighted events per process group in shown inside the brackets. Note that Fig. 5.9 serves for illustration purposes only, as NPLM receives the full data set $\mathcal{R}$ as an input, containing individual event information, and not the histogram displayed here.

This choice of event class was informed both by the results of the most discrepant event classes found for 2016 data (Fig. 3.1) [19], as well as the number of events per class. As it was previously discussed, it is important to avoid double counting when generating toy samples out of the reference sample. Thus, in order to create toy data out of the available Monte Carlo data sets for 2017, the chosen event class must have a reasonable amount of statistics. In this case, the class $1e + 1\mu + p_T^{miss}$ has a total number of Monte Carlo simulated events of $N_{\mathcal{R}} = 74\,389$, summing to $\mathcal{N}_{\mathcal{R}} \sim 6075.6$ weighted events. This results in a factor of $N_{\mathcal{R}}/\mathcal{N}_{\mathcal{R}} \sim 12$, which is compatible with the requirements that the NPLM authors deemed necessary [65, 66].

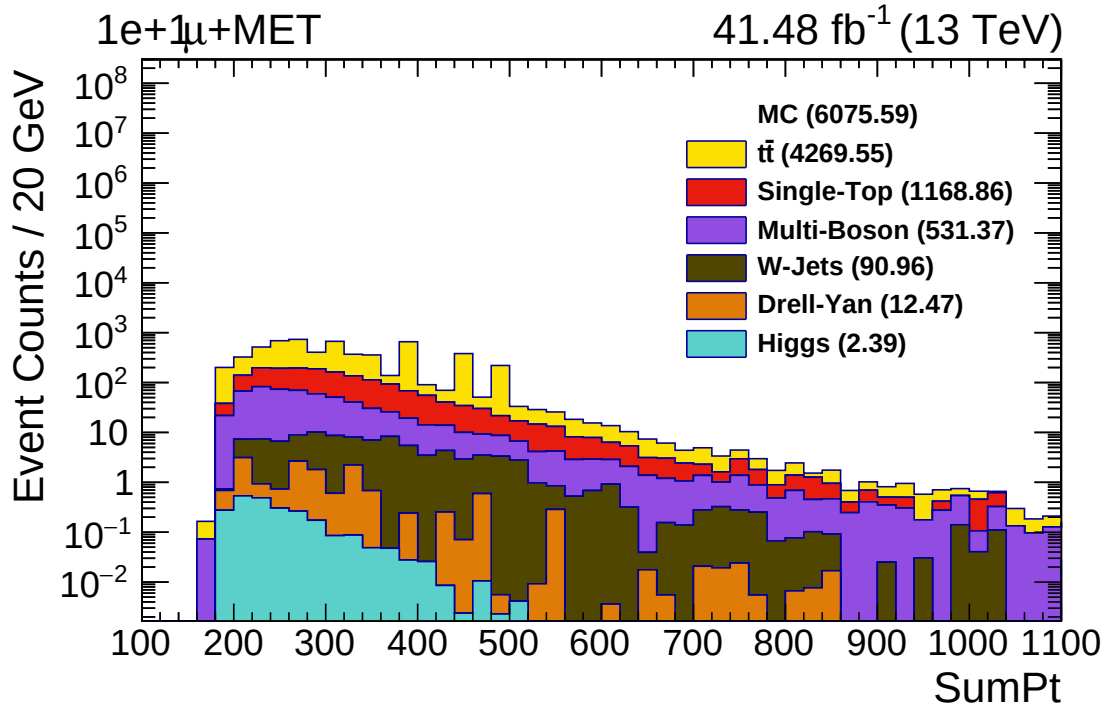


Figure 5.9: SM processes contributing to the event class $1e + 1\mu + p_T^{miss}$, with a total number of .weighted events of 6075.59.

Cleaning the data set for *Hit or Miss*

To make sure the *Hit or Miss* procedure can be employed to generate toy samples, one must first remove negative weights. The number of negatively weighted events in this class is $11\,042$, corresponding to $\sim 14.8\%$ of the total generated events. Moreover, to improve the efficiency of the *Hit or Miss*, a maximum weight was set at $w_{max} = 0.7$, which removed a further 40 events of the data set. Lastly, since no events with high values of the summed transverse momentum were observed in the 2016 analysis for this event class [19], a maximum value was also set for this kinematic variable, corresponding to $x = 1350$ GeV. This resulted $1\,797$ events being further removed from the Monte Carlo simulated reference sample. All in all, the combined number of removed events was $12\,879$ which accounts for $\sim 17.3\%$ of total generated events, and 58.3% of the total weighted events. This information is summarized in Table 3. Note that, after the removal of these events, a weight re-scaling is applied, such that the total number of weighted events remains $\mathcal{N}_{\mathcal{R}} = 6075.6$.

	Nr. of total events	**Nr. of events with w<0**	**Nr. of events with w>0.7**	**Nr. of events with x>1350**	**Nr. of events removed from $\mathcal{R}$**
unweighted	74 389.0	11 042.0	40.0	1 797.0	12 879.0
weighted	6 075.6	−112.4	3 654.2	0.9	3 542.8

Table 3: Summary of the data set clean up parameters, together with the unweighted and weighted number of removed events.

5.2.2 Toy Samples

Similarly to the procedure for the simple case study in Section 5.1, $1\,000$ toy samples, labeled by $\mathcal{D}_{\mathcal{R}}$, are first generated following the reference hypothesis, i.e. SM expectation. These toy samples are used to tune the hyper-parameters of the network, which is discussed in the next subsection. These toy samples were generated via the *Hit or Miss* procedure applied to the cleaned reference data set, discussed above. A factor of $w_f = 0.9$ was chosen such that an event with weight equal to $w_{max} = 0.7$ has a $w_{max}/w_f \sim 0.78$ probability of being selected as a toy event. Although this set up showed a good *Hit or Miss* efficiency, it was still necessary to loop twice over the reference data set to build a toy sample with a number of events $\mathcal{N}_{\mathcal{D}}$, where $\mathcal{N}_{\mathcal{D}}$ is thrown from a Poisson distribution with mean $\mathcal{N}_{\mathcal{R}} \sim 6076$. This does not constitute a problem, as the probability to select the same event twice is not significant.

Similarly, $1\,000$ toy samples, labeled now by $\mathcal{D}_{\mathcal{S}}$, were generated according to one of three alternative hypothesis. For each hypothesis, the construction of the toy samples was also done through the *Hit or Miss* method, with the added step of including the desired signal events. The three considered hypothesis are described below:

- The first hypothesis was to vary the cross section of $t\bar{t}$ events, with $M_{t\bar{t}} > 700$ GeV (see Table 1), by a scale factor of 0.5, 0.8, 1.2 or 1.5. This choice was made in order to test whether or not NPLM can detect changes applied to a specific regime within a SM process group. Two examples of toy samples are shown in Fig 5.10, namely for a cross section variation by a factor of 1.2 (top) and 1.5 (bottom), which resulted in a number of weighted events of 6199.79 and 6386.08, respectively.

- The second hypothesis was to vary the cross section of Higgs processes by a scale factor of 2, 5 or 10. This values were chosen due to the low count of Higgs related events in this class. Moreover, Higgs processes were also removed completely for an additional sensitivity test. One of the toy samples containing no Higgs processes is shown at the top of Fig 5.11, resulting in a number of weighted events equal to 6073.2. In the same figure, at the bottom, is shown a toy sample where the Higgs processes have their cross section scaled by a factor of 10, resulting in a total number of weighted events of 6097.13.
- Lastly, the third hypothesis was to vary the cross section of the sub-leading Multi-Boson processes by a scale factor of 0.8, 0.9, 0.95, 0.98, 1.02, 1.05, 1.1 or 1.2. Fig 5.12 shows two toy samples with the cross section of Multi-Boson processes scaled by a factor of 1.05 (top) and 1.2 (bottom), resulting in a total number of weighted events of 6097.73 and 6164.16, respectively.

The scale parameters chosen above were motivated by the number of events in the specific process group. For instance, for Higgs events, which account for 2.39 weighted events in the reference sample, it is expected that varying the cross section by values lower than 1 will not create a significant enough signal to be searched by NPLM.

Moreover, note that, in most plots mentioned above, it is visually hard to see the difference between the background prediction and the toy data points. The difference is most easily noticeable by looking directly at the event counts in brackets. This provides a good sensitivity test for the NPLM algorithm, by testing the level of small changes in the number of events that NPLM could be able to detect.

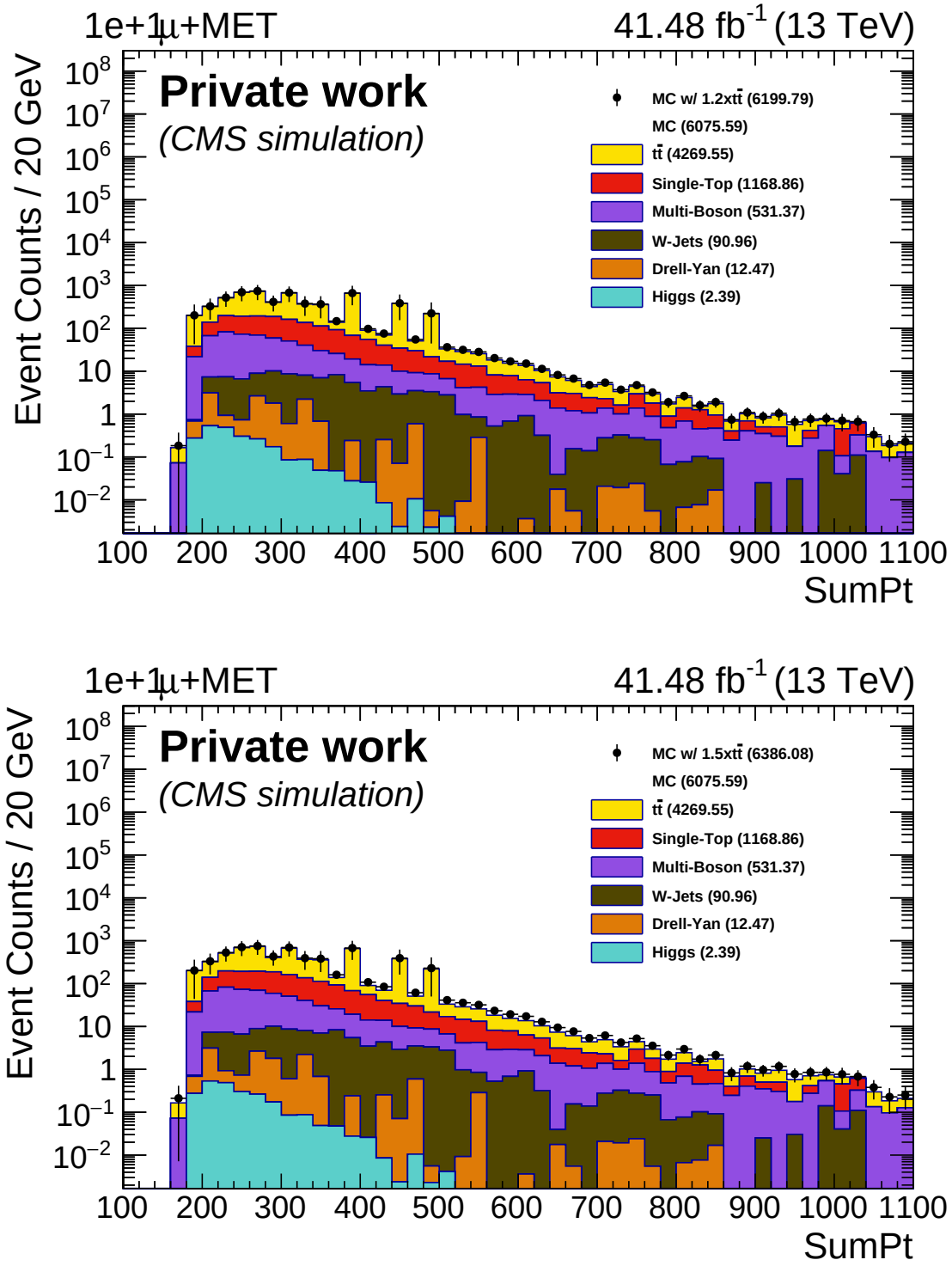


Figure 5.10: The cross section of selected $t\bar{t}$ processes is varied for the class $1e + 1\mu + p_T^{miss}$. (top) The pseudo-data is generated according to the MC prediction, with the cross-section of the $t\bar{t}$ processes increased by 20%. This results in a number of weighted events of 6199.79 as opposed to the 6075.59 predicted MC events for this event class. (bottom) The pseudo-data is generated according to the MC prediction, with the cross-section of the $t\bar{t}$ processes increased by 50%. This results in a number of weighted events of 6386.08.

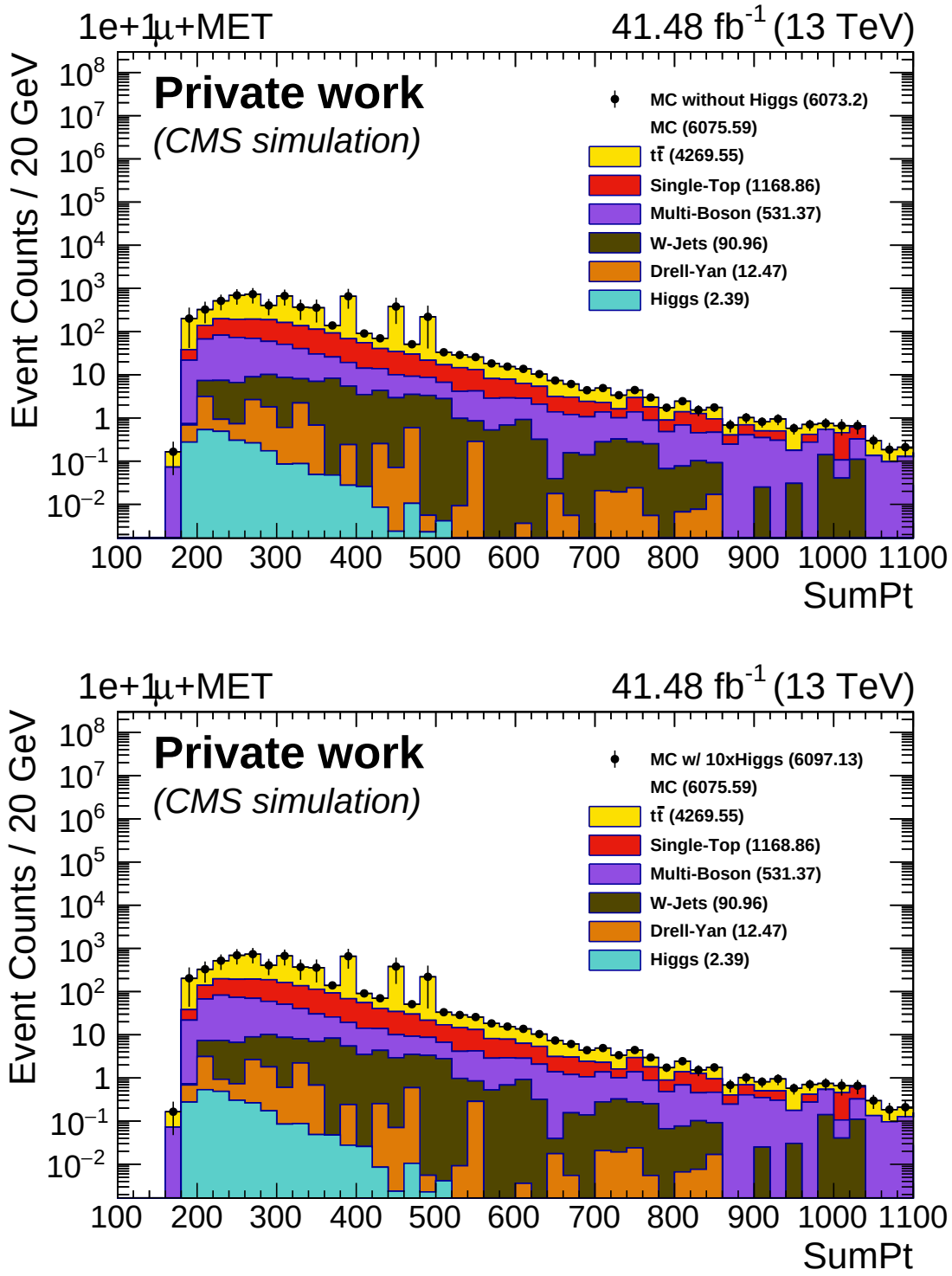


Figure 5.11: The cross section of Higgs processes is varied for the class $1e + 1\mu + p_T^{miss}$. (top) The pseudo-data is generated according to the MC prediction but it does not include Higgs processes. This results in a number of weighted events of 6073.2 as opposed to the 6075.59 predicted MC events for this event class. (bottom) The pseudo-data is generated according to the MC prediction, with the cross-section of Higgs processes increased 10 times. This results in a number of weighted events of 6097.13.

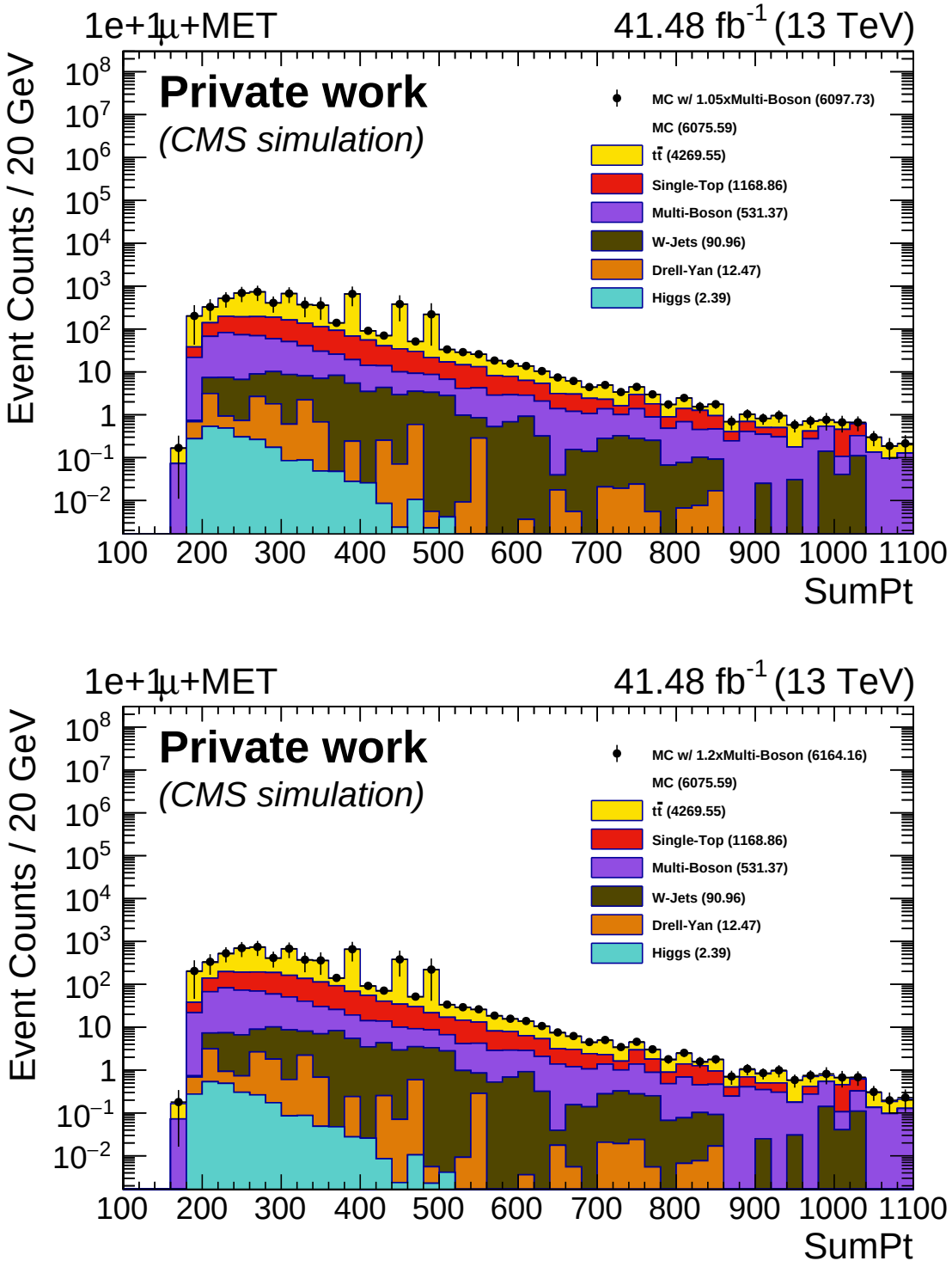


Figure 5.12: The cross section of selected Multi-Boson processes is varied for the class $1e + 1\mu + p_T^{miss}$. (top) The pseudo-data is generated according to the MC prediction, with the cross-section of the Multi-Boson processes increased by 5%. This results in a number of weighted events of 6097.73 as opposed to the 6075.59 predicted MC events for this event class. (bottom) The pseudo-data is generated according to the MC prediction, with the cross-section of the Multi-Boson processes increased by 20%. This results in a number of weighted events of 6164.16.

5.2.3 Sensitivity Studies

The hyper-parameter optimization strategy is first presented. This step corresponds to the blue workflow in Fig. 4.6, where the toy samples $\mathcal{D}_{\mathcal{R}}$ are used as an input to the network, together with the reference sample $\mathcal{R}$.

- **Architecture, Learning Rate and Training Rounds**
  For the MUSiC implementation, the architecture $\vec{a} = (1, 4, 1)$ was also chosen as in the case study, considering that the data sets have somewhat similar complexities. The learning rate was set to 10^{-3} and the number of training rounds was set to $300\ 000$. Other architectures were not studied in detail, as the authors did not see significant improvements in their original paper [65].

- **Weight Clipping Parameter**
  The weight clippings $wc \in \{1, 5, 6, 7, 8, 9, 10, 15, 20, 30, 50, 100\}$ were tested for an architecture $\vec{a} = (1, 4, 1)$. Due to limited computational and time resources, a smaller number of toy samples were initially used, namely 50 and 100, to test all weight clipping values.

  Fig. 5.13 shows the empirical distribution of the test statistic t, in light blue, for the weight clipping parameters $wc \in \{1, 10, 50, 100\}$, as well as the percentile values for t during training. From the empirical distributions and the percentile plots, it can be observed that the optimum weight clipping value will reside around $wc = 10$, which results in a better approximation of $\chi^2(13)$ by the test statistic distribution. Thus, to fine tune the weight clipping parameter even more, 500 toy samples were used to test the three weight clipping parameters $wc \in \{8, 9, 10\}$. The agreement between the resulting test statistic distribution t and the $\chi^2(13)$ expected behaviour was quantified by the Kolmogorov–Smirnov test, the results of which can be found in Table 4.

  A final weight clipping of $wc = 9$ was selected, which resulted in a p-value of $\sim 10^{-4}$ and 0.0002795, at 500 and $1\ 000$ toy samples, respectively. Thus, a $wc = 9$ was used throughout the rest of this section. Fig. 5.14 shows the empirical distribution of the test statistic t compared to the χ^2_{13} asymptotic limit, when $1\ 000$ toy samples were used. The evolution of the t percentiles during training is also shown, together with the expected χ^2_{13} percentiles. Note that, as discussed in the case study analysis, even though the p-value for this weight clipping is not particularly large, it might still result in enough statistical power of the network to distinguish between an alternative and a reference hypothesis.

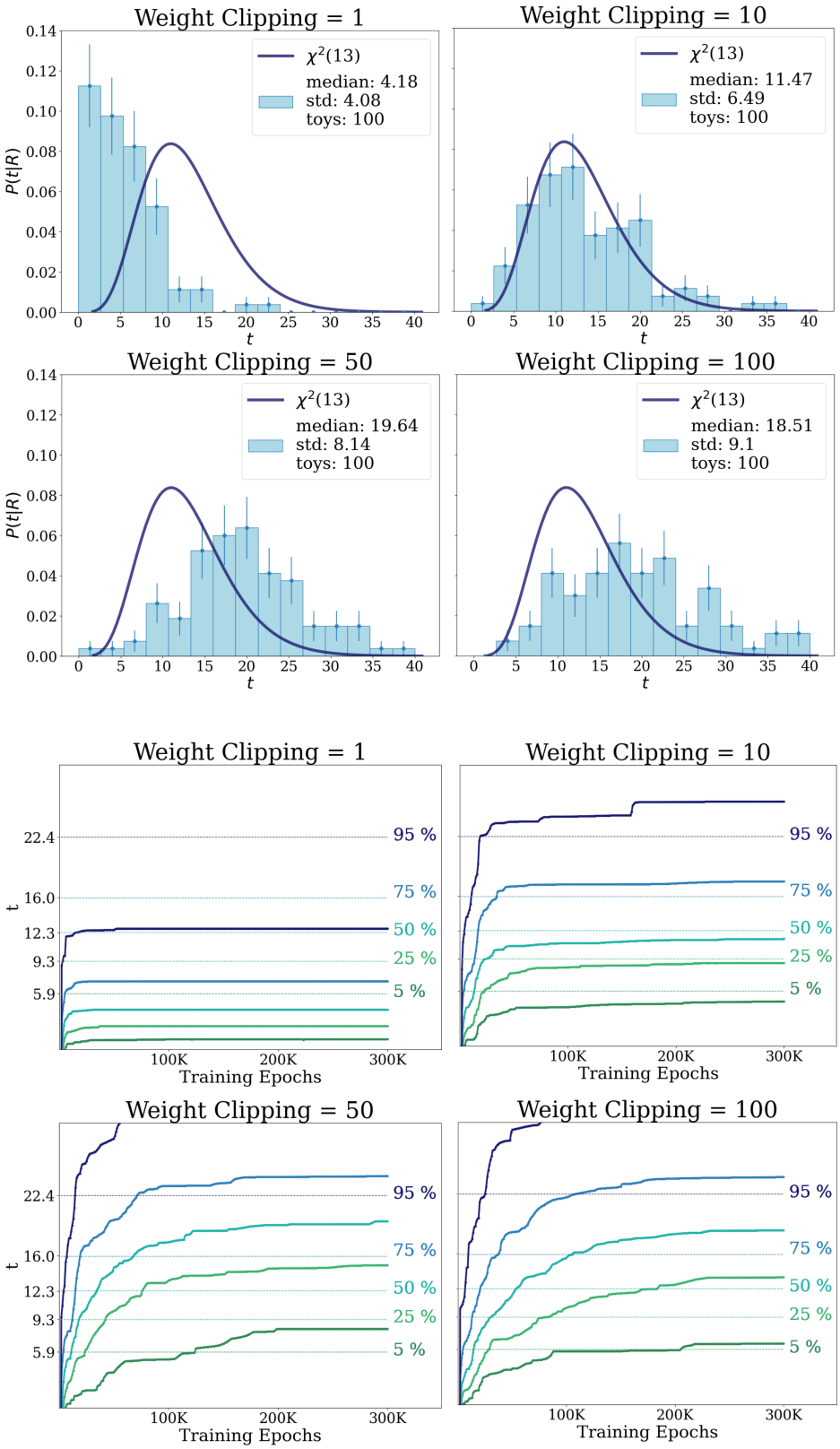


Figure 5.13: Empirical distribution of the test statistic t for different weight clipping parameters. The expected χ^2_{13} asymptotic limit is also shown (top). Corresponding evolution of the t percentiles during training, compared to the expected χ^2_{13} percentiles (bottom). For each weight clipping value, 100 toy experiments were performed.

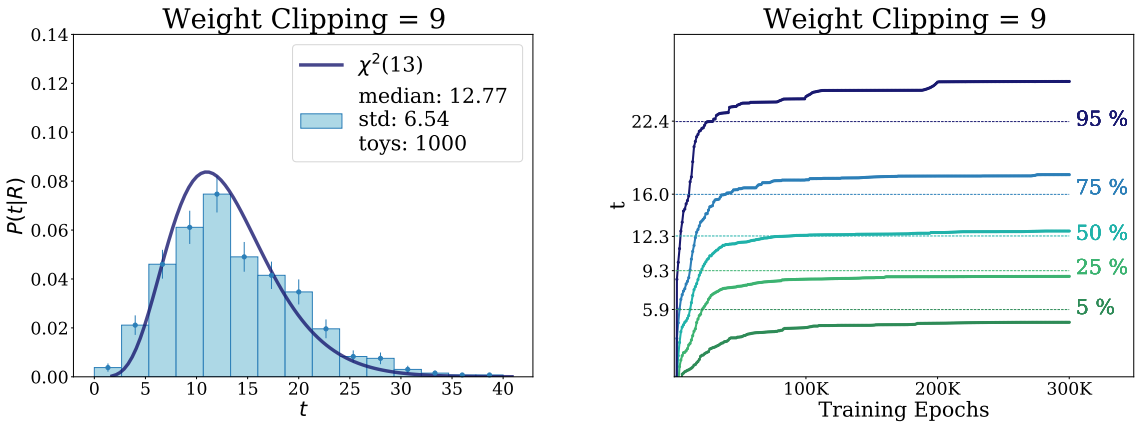


Figure 5.14: (left) Empirical distribution of the test statistic t for weight clipping parameter $wc = 9$, with the expected χ^2_{13} asymptotic limit show as a dark blue line. (right) Corresponding evolution of the t percentiles during training, compared to the expected χ^2_{13} percentiles. For training $1\,000$ toy samples were used.

Table 4: Results of the KS test for different weight clipping parameters for the architecture $(1, 4, 1)$. The training process lasted for $300\,000$ epochs, and used a varying input number of toy data sets. A weight clipping parameter of $wc = 9$ was found to be the most suitable, resulting in a KS test p-value of $\sim 10^{-4}$ and ~ 0.0002795 when, respectively, 500 and $1\,000$ toy samples are used.

Weight Clipping	**50 toys**		**100 toys**		**500 toys**	
	KS p-value	$\langle \bar{t} \rangle - 13$	KS p-value	$\langle \bar{t} \rangle - 13$	KS p-value	$\langle \bar{t} \rangle - 13$
1	$< 10^{-29}$	-9.19 ± 0.75	$< 10^{-42}$	-8.82 ± 0.51		
5	$< 10^{-3}$	-3.29 ± 0.89	$< 10^{-6}$	-3.14 ± 0.69		
6	$\sim 10^{-3}$	-1.62 ± 1.02	$< 10^{-4}$	-2.83 ± 0.74		
7	0.02	-3.01 ± 1.04	$< 10^{-3}$	-3.05 ± 0.69		
8	0.08	-1.90 ± 1.31	0.12	-1.76 ± 0.85	$< 10^{-4}$	-1.48 ± 0.36
9	0.27	0.13 ± 1.06	0.08	0.01 ± 0.83	$\sim 10^{-4}$	-0.23 ± 0.37
10	0.25	0.52 ± 1.28	0.35	-1.53 ± 0.81	$< 10^{-4}$	0.77 ± 0.37
15	$< 10^{-3}$	3.49 ± 1.36	$< 10^{-6}$	3.19 ± 0.89		
20	0.04	0.36 ± 1.22	$< 10^{-3}$	0.56 ± 0.85		
30	$< 10^{-7}$	6.17 ± 1.43	$< 10^{-10}$	4.73 ± 1.07		
50	$< 10^{-9}$	5.94 ± 1.38	$< 10^{-19}$	6.64 ± 1.02		
100	$< 10^{-6}$	5.73 ± 1.80	$< 10^{-15}$	5.62 ± 1.16		

Since all non-trainable hyper-parameters have been set, the next step is to implement the equivalent to the green workflow in Fig. 4.6, i.e. testing the alternative hypothesis. The inputs of the network are now the reference sample $\mathcal{R}$ and the toy samples $\mathcal{D}_S$, where S can stand for either the variation of $t\bar{t}$ processes, Higgs processes or Multi-Boson processes. The statistical power test was, once again, used to quantify the efficiency of the network to identify whether a toy sample was generated according to the reference hypothesis or not. For all three hypothesis testes, the results of the power test can be found in Table 5 and the corresponding test statistic distributions are shown in Fig. 5.15-5.17. Note that the critical test statistic t_c was calculated such that only 5% of the t values found by the network, when the toy samples follow the reference hypothesis, are higher than t_c. This resulted in a value of $t_c = 24.34$. The observations and conclusions for the sensitivity tests for all three alternative hypothesis are presented below.

Varying the cross section of $t\bar{t}$ processes

As it can be observed from Fig. 5.15, the network is only able to distinguish toy samples following SM expectation to those following the alternative hypothesis when the cross section of $t\bar{t}$ processes is scaled up, or down, by 0.5. When the scale factor is reduced to ± 0.2, the network cannot distinguish the toys samples anymore, since the statistical power is not sufficiently high.

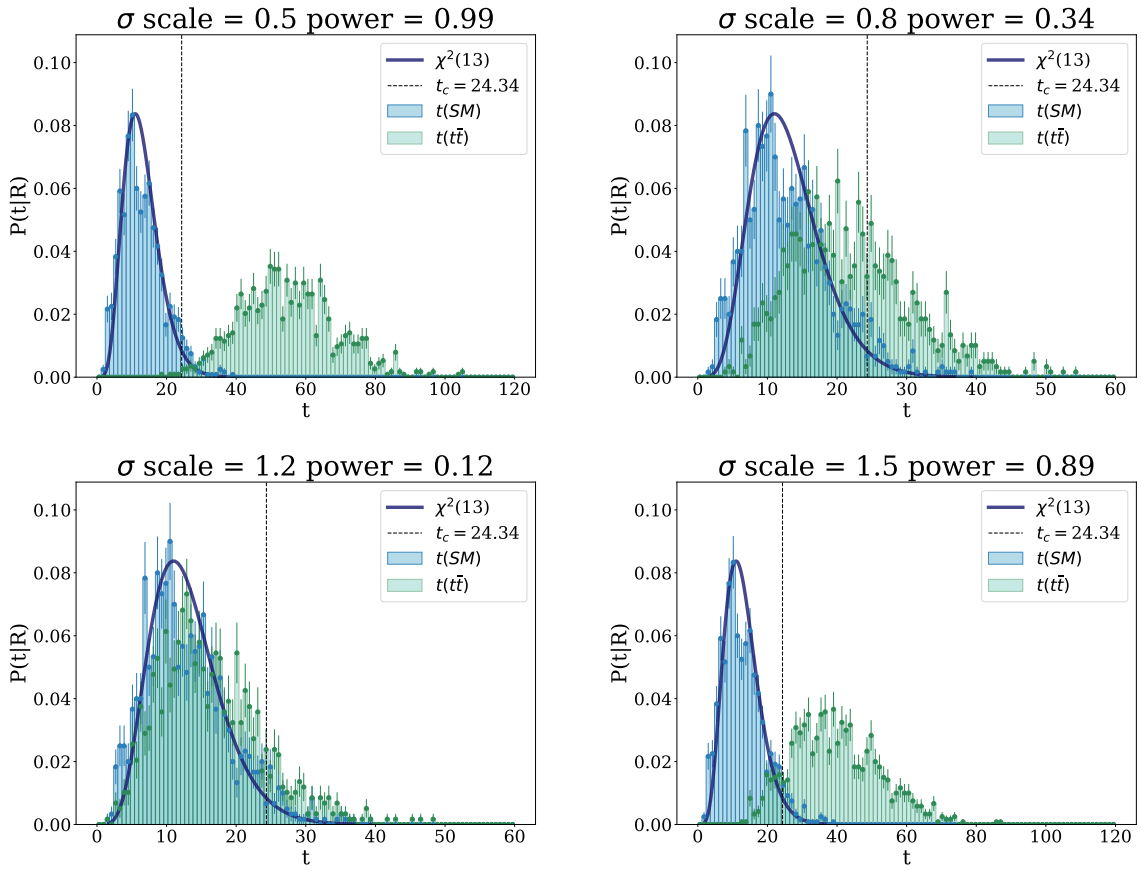


Figure 5.15: Statistical power test for different values of the cross section of $t\bar{t}$ processes. The scale factor applied is shown in each title, together with the calculated statistical power. The blue histogram results from the training of the network when the toy samples D_R are compared to the SM reference sample $\mathcal{R}$. The green histogram, instead, regards the results from comparing the toy signal samples D_S to $\mathcal{R}$.

Varying the cross section of Higgs processes

Fig. 5.16 shows that the network can only distinguish toy samples with a different SM expectation for Higgs events, when the cross section of such events is increased by a factor of 10. This result is not surprising, since Higgs processes only contribute 2.39 weighted events out of a total of 6075.59 weighted events in the reference hypothesis. Moreover, when Higgs events are completely removed (top right), the network is also unable to distinguish the reference from the alternative hypothesis.

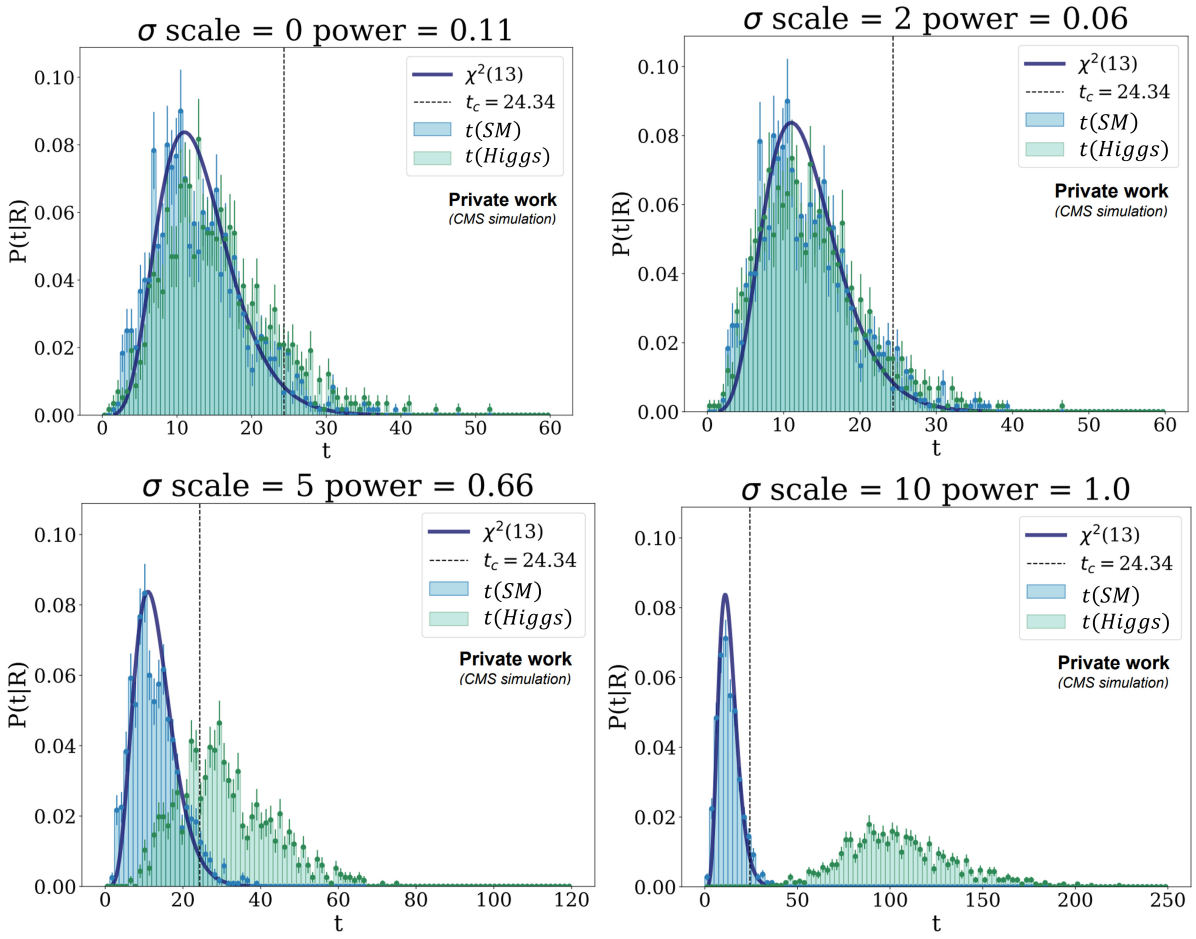


Figure 5.16: Statistical power test for different values of the cross section of Higgs processes. The scale factor applied is shown in each title, together with the calculated statistical power. The blue histogram results from the training of the network when the toy samples $\mathcal{D}_S$ are compared to the SM reference sample $\mathcal{R}$. The green histogram, instead, regards the results from comparing the toy signal samples $\mathcal{D}_S$ to $\mathcal{R}$.

Varying the cross section of Multi-Boson processes

As Fig. 5.17 shows in the bottom right and upper left plots, the cross section of Multi-Boson processes must be scaled up, or down, by at least 10% for the network to gain a significant statistical power, which was observed to be equal to 0.98 in both scenarios.

Moreover, when the cross section is varied by 5% (top right and bottom left plots), the network loses it capacity to identify the alternative hypothesis. In this case, the statistical power decreased by more than half when compared to the 10% scaling.

Lastly, for a small variation of 2%, the toy signal samples are effectively indistinguishable from toy samples following the reference hypothesis, as it is shown by the overlap between the histograms in the middle left and middle right plots.

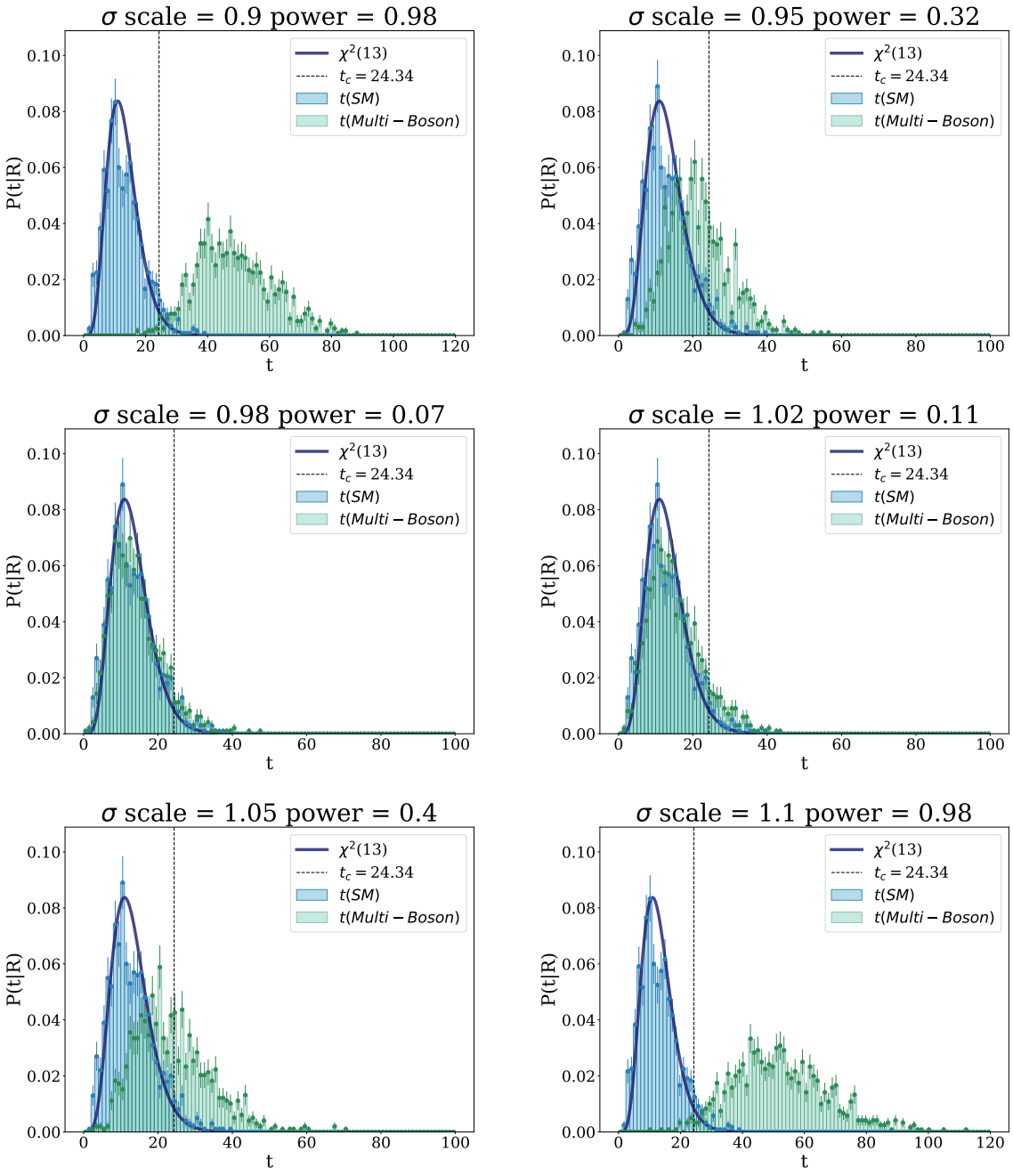


Figure 5.17: Statistical power test for different values of the cross section of Multi-Boson processes. The scale factor applied is shown in each title, together with the calculated statistical power. The blue histogram results from the training of the network when the toy samples D_R are compared to the SM reference sample $\mathcal{R}$. The green histogram, instead, regards the results from comparing the toy signal samples D_S to $\mathcal{R}$.

Further Remarks

A summary of the statistical power of the network for the different alternative hypothesis can be found in Table 5. Note that, for $t\bar{t}$ and Multi-Boson sensitivity tests, there is a degree of partial symmetry between the power resulting from toy signal samples which were scaled up or down by the same scale factor. This is not surprising, but is a good consistency check that NPLM is sensitive to both the excess or absence of events.

Also interesting to mention is, when the cross section of Higgs processes is scaled up by a factor of 10 (upper plot of Fig. 5.11) or when the cross section of Multi-Boson processes are scaled up by a factor of 0.05 (upper plot of Fig. 5.12), the expected number of toy data events is nearly identical at 6097.13 and 6097.73, respectively. However, even with similar number of weighted events, the network shows the corresponding statistical powers of 1.0 and 0.40. This can be explained by the fact that Higgs events have a summed transverse momentum localized at values lower than ~ 500 GeV, while Multi-Boson events can be found throughout the whole kinematic distribution.

Table 5: Statistical power test results for different the Higgs, $t\bar{t}$ and Multi-Boson processes, with a scaled cross-section.

$\mathbf{t\bar{t}}$								
σ scale factor	0.5	0.8	1.2	1.5				
Statistical Power	0.99	0.34	0.12	0.89				
Higgs								
σ scale factor	0.0	2.0	5.0	10.0				
Statistical Power	0.11	0.06	0.66	1.0				
Multi-Boson								
σ scale factor	0.8	0.9	0.95	0.98	1.02	1.05	1.1	1.2
Statistical Power	1.00	0.98	0.32	0.07	0.11	0.40	0.98	1.00

Chapter 6

Conclusions & Outlooks

MUSiC analysis provide a complementary strategy to dedicated searches, by looking for new physics phenomena without being bound to a particular theory model. The MUSiC algorithm analysis hundreds of final states, by assigning each event to several event classes, namely, to a single *exclusive* event class, and to several *inclusive* and *jet-inclusive* event classes. The kinematic variables analysed are the invariant mass, the summed transverse momentum and the missing transverse momentum. Up until the point of writing, MUSiC analysis have employed a region of interest scanning algorithm to search for discrepancies between Monte Carlo SM simulation and CMS measured data, based on binned histograms. After reviewing the advantages of neural networks as function approximants over the binned histogram method, this thesis introduces the first ever implementation of machine learning methods to MUSiC analysis. It was proposed that the current scanning algorithm be replaced by the recently published *New Physics Learning Machine* (NPLM) algorithm [65, 66]. Moreover, the work on this thesis is the second ever application of NPLM to any CMS analysis.

The NPLM algorithm was first tested on a simple case study to validate its suitability to weighed data sets. Different weight distributions were tested, including varying percentages of negatively weighted events. This preliminary check showed that the performance of NPLM is not affected by the increased complexity of the weight distribution. NPLM was then employed to the event class $1 + 1\mu + p_T^{miss}$ for 2017 data. This choice was based on 2016 results, as well as the number of total events in the class. Three alternative hypothesis were tested, namely, varying the cross section of $t\bar{t}$ processes with $M_{t\bar{t}} > 700$ GeV, of Higgs processes and of Multi-Boson processes. NPLM was able to detect this variations when the scale factors applied were ± 0.5, 10 and ± 0.1, respectively. The corresponding statistical powers were 0.99/0.89 for $t\bar{t}$, 1.0 for the Higgs and 0.98 for the Multi-Boson.

Overall, it was concluded that NPLM is a robust algorithm for anomaly detection and shows promising applications to MUSiC analysis. Note that, although this thesis did not include any considerations of systematic uncertainties, NPLM is fully capable of this option. Thus, to fully evaluate whether or not NPLM shows an higher sensitivity than the current MUSiC scanning algorithm, it is suggested that further work be done in this domain. Moreover, it is suggested that NPLM be employed to more challenging event classes, which might contain reduced statistics.

Abbreviations

ALICE	A Large Ion Collider Experiment
ANNs	Artificial Neural Networks
ATLAS	A Toroidal LHC Apparatus
BSM	Beyond Standard Model
CERN	European Organization for Nuclear Research
CMS	Compact Muon Solenoid
ECAL	Electromagnetic Calorimeter
FNN	Feed-forward Neural Network
HCAL	Hadron Calorimeter
HDF5	Hierarchical Data Format
LEP	Large Electron-Positron Collider
LHC	Large Hadron Collider
LHCb	The Large Hadron Collider beauty Experiment
LINAC4	Linear Accelerator 4
MC	Monte Carlo
MUSiC	Model Unspecific Search in CMS
NPLM	New Physics Learning from a Machine
QCD	Quantum Chromodynamics
QED	Quantum Electrodynamics
QFTs	Quantum Field Theories
SM	Standard Model of Particle Physics
WLCG	Worldwide LHC Computing GRID

Appendices

Appendix A

Unit Convention

The natural units convention is used throughout this thesis, where the physical constants c (speed of light), $\hbar$ (reduced Planck constant) and ϵ_0 (electric permittivity) take the value of unity. Therefore, following units from the International System of Units units are modified to:

- $[m] =$ GeV (mass);
- $[t] = 1/$GeV (time);
- $[s] = 1/$GeV (length).

Appendix B

ROOT to HDF5 Conversion

```
import numpy as np
import h5py, glob, math, uproot, os, ROOT, argparse

data_labels = ['SingleMuon', 'SingleElectron', 'SinglePhoton', 'DoubleMuon','DoubleEG']

def Create_dataset(dsetName,datasetName,dataName):
    if len(dataName)!=0:
        dsetName  = f.create_dataset(datasetName, data=dataName,chunks=(len(dataName),),
    maxshape=(None,))
        return
    else:
        return

def Create_string_dataset(dsetName,datasetName,dataName):
    if len(dataName)!=0:
        string_dt = h5py.special_dtype(vlen=str)
        dsetName  = f.create_dataset(datasetName, data=dataName,chunks=(len(dataName),),
    maxshape=(None,), dtype=string_dt)
        return
    else:
        return

def Append_to_dataset(dsetName,datasetName,dataName):
    if len(dataName)!=0:
        dsetName = f[datasetName]
        dsetName.resize((dsetName.shape[0]+len(dataName),))
        new_shape   = dsetName.shape[0]
        entry_point = new_shape-len(dataName)
        dsetName[-len(dataName):] = dataName
        return
    else:
        return

def Read_Branches_from_classname(fileup,classname):
    tree     = fileup[classname]
    branches = tree.arrays()
    if branches['SumPt'].shape[0]==0:
        print('Empty file!')
    else:
        return branches

def Read_ListTree_from_file(fileup):

    allClasses  = fileup.keys()
    nonClasses  = ["ProcessName","EvCounts","TotalEvents","TotalEventsUnweighted"]
    classtree   = []

    for i in range(len(allClasses)):
        if any(j in allClasses[i] for j in nonClasses):
            continue
        else:
            classtree.append(allClasses[i].replace(";1",""))
    return classtree

```

```
53
54 if __name__ == '__main__':
55
56     parser = argparse.ArgumentParser()
57     parser.add_argument('-in','--inputDir',     type=str, help="Directory where the process
       folders are stored.", required=True)
58     parser.add_argument('-out','--outputDir',   type=str, help="Directory to store the h5
       files.", required=True)
59     args = parser.parse_args()
60
61     inputDir  = args.inputDir
62     outputDir = args.outputDir
63
64     Lumi = 41480.
65
66     processDirList = [x[0] for x in os.walk(inputDir)]
67     processDir     = []
68     select_folders = ["AnalysisOutput"]
69     temp_listdir1  = [ item for item in os.listdir(inputDir) if os.path.isdir(os.path.join(
       inputDir,item))]
70
71     processList = []
72
73     for i in temp_listdir1:
74         inputdir = os.path.join(inputDir,i)
75         for x in os.listdir(inputdir):
76             processList.append(x)
77
78     # Printing all process names, which are read from the folder names inside the input
       directory
79     print "All process names are :"
80     print ''
81     for i in processList:
82         print i
83
84     # Creating a list with the grid paths
85     for i in range(len(processDirList)):
86         if any(j in processDirList[i] for j in select_folders):
87             processDir.append(processDirList[i])
88             processDir = [x for x in processDir if "Event-lists" not in x]
89         else:
90             continue
91
92     dic_evCounts = {}
93     dic_tEv      = {}
94     dic_tEvUnweighted  = {}
95
96     # Looping over grid folders
97     for i in range(len(processDir)):
98
99         subdir = processDir[i]
100        processname = ''
101        for j in range(len(processList)):
102            if processList[j] in subdir:
103                processname = processList[j]
104
105        # Checking if is a MC or data sample
106        label = 0
107        if any(i in subdir for i in data_labels):
108            label = 1
109        else:
110            label = 0
111        if len(glob.glob(subdir+"/ECP*")) == 0:
112            continue
113
114        # Opening the ROOT file
115        for fileROOT in glob.glob('%s/ECP*'%(subdir)):
116
117            inFile            = ROOT.TFile.Open(fileROOT,"READ")
118            processName       = inFile.Get("ProcessName")
119            evCount           = inFile.Get("EvCounts")              # total number of
       accepted and unaccepted events
120            totalEv           = inFile.Get("TotalEvents")           # sum of the total
```

```
    weighted events
121         totalEvUnweighted = inFile.Get("TotalEventsUnweighted")  # sum of the Monte Carlo
    generator weights
122         inFile.Close()
123
124         pName         = str(processName[0])
125         evCounts      = evCount[0]
126         tEv           = totalEv[0]
127         tEvUnweighted = totalEvUnweighted[0]
128
129         if dic_evCounts.has_key(pName)==False:
130             dic_evCounts[pName]      = []
131             dic_tEv[pName]           = []
132             dic_tEvUnweighted[pName] = []
133
134         dic_evCounts[pName].append(evCounts)
135         dic_tEv[pName].append(tEv)
136         dic_tEvUnweighted[pName].append(tEvUnweighted)
137
138         # Looping over event classes inside the ROOT file
139         with uproot.open(fileROOT) as fileup:
140             ListTree_classes  = []
141             ListTree_classes  = Read_ListTree_from_file(fileup)
142
143             ML_Classes = ['_2Ele_1bJet', '_1Ele_1Muon_1MET', '_1Ele_1Muon_1Jet','
    _1Ele_1Muon_1Jet_1MET']
144
145             for i in range(len(ListTree_classes)):
146                 classname = ListTree_classes[i]
147                 if classname not in ML_Classes:
148                     continue
149
150                 branches = []
151                 branches = Read_Branches_from_classname(fileup,classname)
152
153                 if branches is not None:
154                     entries_SumPt = len(branches['SumPt'])
155                 else:
156                     print('! This file is empty. No entries appended to hdf5 file !')
157                     print('')
158
159                 fileH5_Dir = outputDir+'H5'+classname+'.h5'
160
161                 # Initializing the arrays to be stored inside the hdf5 file
162                 SumPt      = []
163                 InvMass    = []
164                 MET        = []
165                 weights    = []
166                 PName      = []
167
168                 # Reading variables from branches to fill arrays
169                 for i in range(entries_SumPt):
170                     if branches is not None:
171                         SumPt.append(branches['SumPt'][i])
172                         MET.append(branches['MET'][i])
173                         InvMass.append(branches['InvMass'][i])
174                         PName.append(pName)
175                         weights.append(branches['Weight'][i] * tEvUnweighted/tEv )
176
177                 # Creating the hdf5 files
178                 if os.path.exists(fileH5_Dir)==False:
179                     with h5py.File(fileH5_Dir, 'w') as f:
180                         Create_dataset('sumpt','SumPt',SumPt)
181                         Create_dataset('invmass','InvMass',InvMass)
182                         Create_dataset('met','MET',MET)
183                         Create_dataset('Weights','weights',weights)
184                         Create_string_dataset('processname','ProcessName', PName)
185
186                         fileH5_Dir   = ''
187                         classname    = ''
188
189                 else:
190                     with h5py.File(fileH5_Dir, 'a') as f:
```

```
                            Append_to_dataset('sumpt','SumPt',SumPt)
                            Append_to_dataset('invmass', 'InvMass',InvMass)
                            Append_to_dataset('met','MET',MET)
                            Append_to_dataset('Weights','weights',weights)
                            Append_to_dataset('processname','ProcessName',PName)

                            fileH5_Dir   = ''
                            classname    = ''

    # Calculating the process-specific N_MC and updating the MC weights
    if label == 0 :

        for classname in ML_Classes:
            outfile = ROOT.TFile(classname+'.root', 'RECREATE')
            hist    = ROOT.TH1D('SumPt'+classname, 'SumPt'+classname, 10000,0,10000)

            fileH5_Dir = outputDir+'H5'+classname+'.h5'

            with h5py.File(fileH5_Dir, 'r+') as f:
                sumpt      = f['SumPt']
                weights1   = f['weights']
                procName   = f['ProcessName']
                NewWeights = []
                NewWeights_lastweight = []
                outtxt     = open('processList_ML.txt','w')
                alreadyin  = []

                for entry in procName:
                    if entry not in alreadyin:
                        alreadyin.append(entry)

                for i in range(len(sumpt)):

                        N_MC = sum(dic_tEvUnweighted[procName[i]])
                        NewWeights.append(weights1[i] * Lumi / N_MC)
                        hist.Fill(sumpt[i], weights1[i]* Lumi / N_MC)

                Create_dataset('newweights','NewWeights',NewWeights)
                hist.Write()

                for process in alreadyin:
                    outtxt.write(process+'\n')

        outfile.Close()
```

Bibliography

[1] *The Standard Model of Elementary Particles*. Apr. 2023. URL: https://en.wikipedia.org/wiki/File:Standard_Model_of_Elementary_Particles.svg.

[2] C. S. Wu et al. "Experimental Test of Parity Conservation in Beta Decay". In: *Physical Review* 105.4 (1957), pp. 1413–1415. DOI: 10.1103/PhysRev.105.1413.

[3] Peter W. Higgs. "Broken Symmetries and the Masses of Gauge Bosons". en. In: *Physical Review Letters* 13.16 (Oct. 1964), pp. 508–509. ISSN: 0031-9007. DOI: 10.1103/PhysRevLett.13.508. URL: https://link.aps.org/doi/10.1103/PhysRevLett.13.508 (visited on 04/06/2023).

[4] F. Englert and R. Brout. "Broken Symmetry and the Mass of Gauge Vector Mesons". en. In: *Physical Review Letters* 13.9 (Aug. 1964), pp. 321–323. ISSN: 0031-9007. DOI: 10.1103/PhysRevLett.13.321. URL: https://link.aps.org/doi/10.1103/PhysRevLett.13.321 (visited on 04/06/2023).

[5] S. Chatrchyan et al. "Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC". en. In: *Physics Letters B* 716.1 (Sept. 2012), pp. 30–61. ISSN: 03702693. DOI: 10.1016/j.physletb.2012.08.021. URL: https://linkinghub.elsevier.com/retrieve/pii/S0370269312008581 (visited on 04/10/2023).

[6] The CMS Collaboration. "Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC". In: *Physics Letters B* 716.1 (Sept. 2012). arXiv:1207.7235 [hep-ex], pp. 30–61. ISSN: 03702693. DOI: 10.1016/j.physletb.2012.08.021. URL: http://arxiv.org/abs/1207.7235 (visited on 04/10/2023).

[7] D.E. Nagle. "The Delta: The First Pion Nucleon Resonance, Its Discovery and Applications". In: *LALP-84-27 Los Alamos National Laboratory* (July 1984).

[8] Lyndon Evans and Philip Bryant. "LHC Machine". In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08001–S08001. ISSN: 1748-0221. DOI: 10.1088/1748-0221/3/08/S08001. URL: https://iopscience.iop.org/article/10.1088/1748-0221/3/08/S08001 (visited on 04/10/2023).

[9] The CMS Collaboration et al. "The CMS experiment at the CERN LHC". In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08004–S08004. ISSN: 1748-0221. DOI: 10.1088/1748-0221/3/08/S08004. URL: https://iopscience.iop.org/article/10.1088/1748-0221/3/08/S08004 (visited on 03/16/2023).

[10] The CMS Collaboration. *Detector - CMS Experiment*. URL: https://cms.cern/detector.

[11] Izaak Neutelings and Alexandros Tsagkaropolulos. *TikZ.net*. URL: https://tikz.net/neural_networks/.

[12] Martina Ressegotti and On behalf of the CMS Collaboration. "Overview of the CMS Detector Performance at LHC Run 2". en. In: *Universe* 5.1 (Jan. 2019), p. 18. ISSN: 2218-1997. DOI: 10.3390/universe5010018. URL: https://www.mdpi.com/2218-1997/5/1/18 (visited on 02/27/2023).

[13] The CMS Collaboration. “The CMS hadron calorimeter project: Technical Design Report”. In: *CERN-LHCC-97-03; CMS-TDR-2.* (June 1997). URL: http://cds.cern.ch/record/357153.

[14] L et al. Bayatian G L et al. “Detector Performance and Software”. In: *CMS Physics: Technical Design Report Volume 1*. CERN, Geneva, 2006. URL: http://cds.cern.ch/record/922757..

[15] The CMS collaboration. “The performance of the CMS muon detector in proton-proton collisions at $\sqrt{s}$ = 7 TeV at the LHC”. In: *Journal of Instrumentation* 8.11 (Nov. 2013), P11002–P11002. ISSN: 1748-0221. DOI: 10.1088/1748-0221/8/11/P11002. URL: https://iopscience.iop.org/article/10.1088/1748-0221/8/11/P11002 (visited on 04/10/2023).

[16] Collaboration CMS. “Performance of muon reconstruction including Alignment Position Errors for 2016 Collision Data.” In: (Nov. 2016).

[17] The CMS Collaboration. “CMS TriDAS project: Technical Design Report, Volume 1: The Trigger Systems”. In: *CERN-LHCC-2000-038; CMS-TDR-6.1.* (Dec. 2000). URL: http://cds.cern.ch/%20record/706847.

[18] The CMS Collaboration. “CMS TriDAS Project: Technical Design Report, Volume 2: Data Acquisition and High-Level Trigger”. In: *CERN-LHCC-2002-026; CMS-TDR-6.2.* (Dec. 2002). URL: http://cds.cern.ch/record/578006.

[19] CMS Collaboration. *MUSiC: a model-unspecific search for new physics in proton-proton collisions at $\sqrt{s} = $ 13 TeV*. 2021. DOI: 10.1140/epjc/s10052-021-09236-z.

[20] C. Hof. ““Implementation of a Model-Independent Search for New Physics with the CMS Detector exploiting the World-Wide LHC Computing Grid””. PhD Thesis. III. Physikalisches Institut A, RWTH Aachen University, Oct. 2009.

[21] E. Dietz-Laursonn. ““Model Unspecific Search for New Physics with b-Hadrons in CMS””. Diploma Thesis. III. Physikalisches Institut A, RWTH Aachen University, Oct. 2010.

[22] H. Pieta. ““MUSiC - A Model Unspecific Search in CMS based on 2010 LHC data””. PhD Thesis. III. Physikalisches Institut A, RWTH Aachen University, 2012.

[23] M. Brodski. ““Model Unspecific Search in CMS with tau Leptons””. MA thesis. III. Physikalisches Institut A, RWTH Aachen University, Sept. 2012.

[24] P. Papacz. ““Model Unspecific Search for new Physics in CMS Based on 2011 Data””. PhD Thesis. III. Physikalisches Institut A, RWTH Aachen University, May 2014.

[25] A. A. E. Albert. ““Extension of the Model Unspecific Search in CMS to Final States with Jets using 2012 Data””. MA thesis. III. Physikalisches Institut A, RWTH Aachen University, Nov. 2015.

[26] D. Durchardt. ““MUSiC: A Model Unspecific Search for New Physics Based on CMS Data at s = 8 TeV””. PhD Thesis. III. Physikalisches Institut A, RWTH Aachen University, 2017.

[27] CMS Collaboration. ““MUSiC, a Model Unspecific Search for New Physics, in pp Collisions at s = 8 TeV””. In: *Technical Report CMS-PAS-EXO-14-016 CERN* (2017).

[28] J. Roemer. ““Model Unspecific Search for New Physics with pp Collisions at s = 13 TeV with the CMS Experiment””. MA thesis. III. Physikalisches Institut A, RWTH Aachen University, Feb. 2017.

[29] S. Knutzen. ""A software for the reinterpretation of model independent search results and constraining theories beyond the Standard Model"". PhD Thesis. III. Physikalisches Institut A, RWTH Aachen University, 2017.

[30] The CMS Collaboration. "CMS luminosity measurement for the 2017 data-taking period at $\sqrt{s} = 13~\mathrm{TeV}$". In: *CMS-PAS-LUM-17-004* (2018). URL: https://inspirehep.net/literature/1677076.

[31] CMS Monte Carlo Validation Group . *MC-validation twiki*. URL: https://twiki.cern.ch/twiki/bin/viewauth/CMS/GeneratorMain.

[32] S. Bailey et al. "Parton distributions from LHC, HERA, Tevatron and fixed target data: MSHT20 PDFs". In: *The European Physical Journal C* 81.4 (Apr. 2021). arXiv:2012.04684 [hep-ex, physics:hep-ph], p. 341. ISSN: 1434-6044, 1434-6052. DOI: 10.1140/epjc/s10052-021-09057-0. URL: http://arxiv.org/abs/2012.04684 (visited on 01/11/2023).

[33] 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". en. In: *Journal of High Energy Physics* 2014.7 (July 2014), p. 79. ISSN: 1029-8479. DOI: 10.1007/JHEP07(2014)079. URL: http://link.springer.com/10.1007/JHEP07(2014)079 (visited on 02/14/2023).

[34] Torbjörn Sjöstrand et al. "An introduction to PYTHIA 8.2". en. In: *Computer Physics Communications* 191 (June 2015), pp. 159–177. ISSN: 00104655. DOI: 10.1016/j.cpc.2015.01.024. URL: https://linkinghub.elsevier.com/retrieve/pii/S0010465515000442 (visited on 02/13/2023).

[35] Simone Alioli et al. "A general framework for implementing NLO calculations in shower Monte Carlo programs: the POWHEG BOX". en. In: *Journal of High Energy Physics* 2010.6 (June 2010), p. 43. ISSN: 1029-8479. DOI: 10.1007/JHEP06(2010)043. URL: http://link.springer.com/10.1007/JHEP06(2010)043 (visited on 02/13/2023).

[36] Enrico Bothmann et al. "Event generation with Sherpa 2.2". In: *SciPost Physics* 7.3 (Sept. 2019), p. 034. ISSN: 2542-4653. DOI: 10.21468/SciPostPhys.7.3.034. URL: https://scipost.org/10.21468/SciPostPhys.7.3.034 (visited on 02/13/2023).

[37] S. Agostinelli et al. "Geant4—a simulation toolkit". en. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 506.3 (July 2003), pp. 250–303. ISSN: 01689002. DOI: 10.1016/S0168-9002(03)01368-8. URL: https://linkinghub.elsevier.com/retrieve/pii/S0168900203013688 (visited on 02/14/2023).

[38] The NNPDF collaboration et al. "Parton distributions for the LHC run II". en. In: *Journal of High Energy Physics* 2015.4 (Apr. 2015), p. 40. ISSN: 1029-8479. DOI: 10.1007/JHEP04(2015)040. URL: http://link.springer.com/10.1007/JHEP04(2015)040 (visited on 02/13/2023).

[39] Rene Brun et al. *root-project/root: v6.18/02*. Aug. 2019. DOI: 10.5281/ZENODO.3895860. URL: https://zenodo.org/record/3895860 (visited on 02/08/2023).

[40] *Physics eXtension Library*. URL: https://vispa.physik.rwth-aachen.de/pxl.

[41] RWTH Aachen University III. Physics Institute A. *Three A Physics Analysis Software (TAPAS)*. URL: https://gitlab.cern.ch/aachen-3a.

[42] Glen Cowan et al. “Asymptotic formulae for likelihood-based tests of new physics”. In: *The European Physical Journal C* 71.2 (Feb. 2011). arXiv:1007.1727 [hep-ex, physics:physics], p. 1554. ISSN: 1434-6044, 1434-6052. DOI: 10.1140/epjc/s10052-011-1554-0. URL: http://arxiv.org/abs/1007.1727 (visited on 02/23/2023).

[43] Adrian E. Bayer and Uroš Seljak. “The look-elsewhere effect from a unified Bayesian and frequentist perspective”. In: *Journal of Cosmology and Astroparticle Physics* 2020.10 (Oct. 2020), pp. 009–009. ISSN: 1475-7516. DOI: 10.1088/1475-7516/2020/10/009. URL: https://iopscience.iop.org/article/10.1088/1475-7516/2020/10/009 (visited on 02/21/2023).

[44] Mike Folk et al. “An overview of the HDF5 technology suite and its applications”. en. In: *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*. Uppsala Sweden: ACM, Mar. 2011, pp. 36–47. ISBN: 978-1-4503-0614-0. DOI: 10.1145/1966895.1966900. URL: https://dl.acm.org/doi/10.1145/1966895.1966900 (visited on 04/04/2023).

[45] Warren S. McCulloch and Walter Pitts. “A logical calculus of the ideas immanent in nervous activity”. en. In: *The Bulletin of Mathematical Biophysics* 5.4 (Dec. 1943), pp. 115–133. ISSN: 0007-4985, 1522-9602. DOI: 10.1007/BF02478259. URL: http://link.springer.com/10.1007/BF02478259 (visited on 02/28/2023).

[46] T. M. Mitchell. *Machine Learning*. New York: McGraw-Hill Science/Engineering/Math, 1997.

[47] Catherine F. Higham and Desmond J. Higham. “Deep Learning: An Introduction for Applied Mathematicians”. In: (2018). Publisher: arXiv Version Number: 1. DOI: 10.48550/ARXIV.1801.05894. URL: https://arxiv.org/abs/1801.05894 (visited on 02/28/2023).

[48] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. URL: http://www.deeplearningbook.%20org.

[49] *Mathematica*. URL: https://www.wolfram.com/mathematica/.

[50] Izaak Neutelings. *Tikz.net/neural_networks/*. URL: https://tikz.net/neural_networks/.

[51] Chigozie Nwankpa et al. “Activation Functions: Comparison of trends in Practice and Research for Deep Learning”. In: (2018). Publisher: arXiv Version Number: 1. DOI: 10.48550/ARXIV.1811.03378. URL: https://arxiv.org/abs/1811.03378 (visited on 02/28/2023).

[52] Johannes Fürnkranz and International Machine Learning Society, eds. *Proceedings, Twenty-Seventh International Conference on Machine Learning: held June 21 - June 25 in Haifa, Israel*. eng. Meeting Name: ICML. S.l.: ACM, 2010. ISBN: 978-1-60558-907-7.

[53] Lars Stietz. “Artificial Neural Networks for Individual Tracking and Characterization of Wake Vortices in LiDAR Measurements”. MA thesis. University of Hamburg, July 2022. URL: https://elib.dlr.de/189820/1/Masterthesis-Stietz.pdf.

[54] C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[55] Md Zahangir Alom et al. *The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches*. arXiv:1803.01164 [cs]. Sept. 2018. URL: http://arxiv.org/abs/1803.01164 (visited on 03/01/2023).

[56] Luc Vinet and Alexei Zhedanov. “A "missing" family of classical orthogonal polynomials”. In: *Journal of Physics A: Mathematical and Theoretical* 44.8 (Feb. 2011). arXiv:1011.1669 [math], p. 085201. ISSN: 1751-8113, 1751-8121. DOI: 10.1088/1751-8113/44/8/085201. URL: http://arxiv.org/abs/1011.1669 (visited on 03/01/2023).

[57] M. A. Cauchy. “Méthode générale pour la résolution des systèmes d’équations simultanées”. In: *Comptes Rendus Hebd. Séances Acad. Sci.* 25 (1847), pp. 536–538.

[58] Léon Bottou. “Stochastic Gradient Descent Tricks”. en. In: *Neural Networks: Tricks of the Trade*. Ed. by Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller. Vol. 7700. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 421–436. ISBN: 978-3-642-35288-1 978-3-642-35289-8. DOI: 10.1007/978-3-642-35289-8_25. URL: http://link.springer.com/10.1007/978-3-642-35289-8_25 (visited on 03/01/2023).

[59] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. *Learning internal representations by error propagation*. Tech. rep. California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[60] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: (2014). Publisher: arXiv Version Number: 9. DOI: 10.48550/ARXIV.1412.6980. URL: https://arxiv.org/abs/1412.6980 (visited on 03/01/2023).

[61] Geoff Hinton. *Lecture notes Neural Networks for Machine Learning (Lecture 6)*. URL: http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf..

[62] Vladik Ya. Kreinovich. “Arbitrary nonlinearity is sufficient to represent all functions by neural networks: A theorem”. en. In: *Neural Networks* 4.3 (Jan. 1991), pp. 381–383. ISSN: 08936080. DOI: 10.1016/0893-6080(91)90074-F. URL: https://linkinghub.elsevier.com/retrieve/pii/089360809190074F (visited on 03/02/2023).

[63] G. Cybenko. *Approximation by Superpositions of a Sigmoidal Function*. 303-314. Math. Control. Signals, Syst., 1989.

[64] Kurt Hornik. “Approximation capabilities of multilayer feedforward networks”. en. In: *Neural Networks* 4.2 (1991), pp. 251–257. ISSN: 08936080. DOI: 10.1016/0893-6080(91)90009-T. URL: https://linkinghub.elsevier.com/retrieve/pii/089360809190009T (visited on 03/02/2023).

[65] Raffaele Tito D’Agnolo and Andrea Wulzer. “Learning New Physics from a Machine”. In: *Physical Review D* 99.1 (2019), p. 015014. DOI: 10.1103/PhysRevD.99.015014.

[66] Raffaele Tito d’Agnolo et al. *Learning New Physics from an Imperfect Machine*. arXiv:2111.13633 [hep-ph]. Nov. 2021. DOI: 10.48550/arXiv.2111.13633.

[67] J. Neyman and E. S. Pearson. “On the Problem of the Most Efficient Tests of Statistical Hypotheses”. en. In: *Breakthroughs in Statistics*. Ed. by Samuel Kotz and Norman L. Johnson. Series Title: Springer Series in Statistics. New York, NY: Springer New York, 1992, pp. 73–108. ISBN: 978-0-387-94037-3 978-1-4612-0919-5. DOI: 10.1007/978-1-4612-0919-5_6. URL: http://link.springer.com/10.1007/978-1-4612-0919-5_6 (visited on 03/31/2023).

[68] S. S. Wilks. “The Large-Sample Distribution of the Likelihood Ratio for Testing Composite Hypotheses”. en. In: *The Annals of Mathematical Statistics* 9.1 (Mar. 1938), pp. 60–62. ISSN: 0003-4851. DOI: 10.1214/aoms/1177732360. URL: http://projecteuclid.org/euclid.aoms/1177732360 (visited on 04/03/2023).

[69] A. N. Kolmogorov. “Sulla Determinazione Empirica di Una Legge di Distribuzione”. In: *Giornale dell’Istituto Italiano degli Attuari* 4 (1933), pp. 83–91.

[70] George E. Forsythe. “Von Neumann’s Comparison Method for Random Sampling from the Normal and Other Distributions”. In: *Mathematics of Computation* 26.120 (Oct. 1972), p. 817. ISSN: 00255718. DOI: 10.2307/2005864. URL: https://www.jstor.org/stable/2005864?origin=crossref (visited on 04/05/2023).

[71] George Casella, Christian P. Robert, and Martin T. Wells. “Generalized Accept-Reject sampling schemes”. en. In: *Institute of Mathematical Statistics Lecture Notes - Monograph Series*. Beachwood, Ohio, USA: Institute of Mathematical Statistics, 2004, pp. 342–347. ISBN: 978-0-940600-61-4. DOI: 10.1214/lnms/1196285403. URL: http://projecteuclid.org/euclid.lnms/1196285403 (visited on 04/03/2023).

[72] Radford M. Neal. “Slice sampling”. In: *The Annals of Statistics* 31.3 (June 2003). ISSN: 0090-5364. DOI: 10.1214/aos/1056562461. URL: https://projecteuclid.org/journals/annals-of-statistics/volume-31/issue-3/Slice-sampling/10.1214/aos/1056562461.full (visited on 04/04/2023).

[73] Joseph S. Rossi. “Statistical Power Analysis”. en. In: *Handbook of Psychology, Second Edition*. Ed. by Irving Weiner. Hoboken, NJ, USA: John Wiley & Sons, Inc., Sept. 2012, hop202003. ISBN: 978-0-470-61904-9 978-1-118-13388-0. DOI: 10.1002/9781118133880.hop202003. URL: https://onlinelibrary.wiley.com/doi/10.1002/9781118133880.hop202003 (visited on 04/04/2023).