University of Heidelberg

Department of Physics and Astronomy

Master Thesis in Physics

submitted by

Mathias Backes

born in Neunkirchen (Germany)

2022

Machine Learning Unfolding based on conditional Invertible Neural Networks using Iterative Training

This Master Thesis has been carried out by Mathias Backes at the
Kirchhoff Institute for Physics in Heidelberg
under the supervision of
Priv.-Doz. Dr. Monica Dunford.

Abstract

Throughout the last decades, matrix-based unfolding has been used in countless analyses to remove detector effects from measured data. The main bottleneck of these approaches is the exponential scaling of computational resources for multiple dimensions due to the introduced binning of the phase space. Hence, there is a demand for unbinned approaches, which use an event-by-event treatment of the data. In recent years, machine learning tools have thus been applied to the task of unfolding. One possibility is to use conditional Invertible Neural Networks (cINN).
This thesis proposes a method for comparrison of the matrix-based algorithms to cINN unfolding by implementing a *matrix-based single event unfolding*. In addition, an extension of cINN unfolding is proposed to iteratively reduce the model-dependency in the unfolding process. The performance of this *Iterative cINN unfolding* (IcINN) is demonstrated by unfolding pseudo-data with a $pp \to Z\gamma\gamma$ final state.

Abstract (in deutscher Übersetzung)

In den letzten Jahrzehnten wurden matrix-basierte Unfolding-Algorithmen eingesetzt um Detektoreffekte von gemessenen Daten zu entfernen. Das Hauptproblem dieser Ansätze ist das exponentielle Skalieren der benötigten Rechenleistung für höhere Dimensionen aufgrund der eingeführten Histogrammdarstellung des Phasenraumes. Infolgedessen existiert eine Nachfrage für Ansätze, welche auf diese Histogrammdarstellung verzichten und stattdessen die Daten auf Basis einzelner Events behandeln. In den letzten Jahren wurden daher Werkzeuge des maschinellen Lernens auf die Herausforderung des Unfoldings angewandt. Eine Möglichkeit ist hierbei die Verwendung von konditionalisierten invertierbaren neuronalen Netzen (cINN).
Diese Arbeit schlägt eine Möglichkeit vor, wie matrix-basierte Algorithmen durch die Implementierung eines *matrix-basiertes Unfolding von einzelnen Events* mit cINN Unfolding verglichen werden können. Des Weiteren wird eine Erweiterung des cINN Unfoldings vorgeschlagen, welches die Modellabhängigkeit im Unfolding-Prozess iterativ reduziert. Die Wirksamkeit dieses *Iterativen cINN Unfolding* (IcINN) wird anhand von Pseudo-Daten mit $pp \to Z\gamma\gamma$ im Endzustand demonstriert.

Contents

1. Introduction

The detectors used in modern particle colliders perform some of the most precise measurements throughout physics. In order to probe theoretical predictions from the Standard Model of Particle Physics, it is necessary to compare them to measured data. This can be a challenge as the uncertainty of the experimental data depends on the resolution of the detector. These uncertainties are so-called *detector effects*, which are not present in the theoretical predictions. To allow for comparisons in experimental searches, these detector effects can either be added to the theoretical predictions or removed from the experimental data. The latter is called *unfolding*. Both approaches require a precise knowledge of the detector's response to the measured phenomena.

Unfolding a data distribution is a non-trivial task due to the statistical nature of the data-taking process. In the last decades, mostly *matrix-based* unfolding has been used, which relies on the construction and (pseudo-)inversion of the detector response matrix [1]. A simple matrix inversion produces a mathematically correct, but physically undesirable result, because it includes large bin-to-bin fluctuations and uncertainties. Hence, it is necessary to implement a regularization [2]. A different approach to matrix-based unfolding are *iterative* matrix-based algorithms [3, 4]. These algorithms perform a bayesian pseudo-inversion of the detector response, which is based on an explicit prior assumption about the result of the unfolding.
The unfolding result carries a certain amount of bias towards this prior assumption. This problem is called *model dependency*: the final result depends on the model which has been used to describe the data. The prior is iteratively corrected, thus reducing the model dependency. Unfortunately, an increasing number of iterations increases the uncertainties. Hence, only a limited amount of iterations is beneficial. It is essential to choose the number of iterations such that a balance between the model dependency and the uncertainty is obtained.
The introduced binning in the matrix-based algorithms creates challenges: in a multi-dimensional unfolding, the number of entries in the response matrix increases exponentially with each further dimension. Therefore, the unfolding of the full phase space information, i.e. an unfolding of all observables measured by the detector, is computationally not feasible.

Due to this, it is desirable to introduce an unbinned unfolding based on machine learning tools. These tools treat the data on an event-by-event basis, which is easily scalable to higher dimensions. One possibility to implement a machine learning based unfolding is to use conditional invertible neural networks in the so-called *cINN unfolding* [5]. The basic idea of this approach is the implementation of an event-generation conditionalised on measured events. As a consequence, the cINN is not only able to unfold a full measured distribution, but also single events. Several applications of the cINN unfolding will be explored in this thesis.

After recapitulating several aspects of the Standard Model and a discussion of the ATLAS detector in Section 2, the basics of matrix-based unfolding as well as an analytic toy model for unfolding are introduced in Section 3. The subsequent Section 4 introduces the basics of machine learning. Section 5 explains how machine learning tools are currently used in unfolding problems.
Section 6 discusses a fundamentally new approach to predict *single event unfolded distributions, using the matrix-based unfolding algorithms*. Especially the iterative algorithms yield promising results. In Section 7 the limits of cINN unfolding are demonstrated and a new, iterative unfolding algorithm based on cINN unfolding is introduced: the *Iterative cINN unfolding* (IcINN). The performance of this algorithm is illustrated in Section 8 by unfolding pseudo-data of the process $pp \to Z\gamma\gamma$ [6].

Author's contribution
The first project was the generation of single event unfolded distribution for the matrix-based methods. To achieve this, I modified the standard matrix-based unfolding algorithms implemented in the package *RooUnfold* [7], which is provided by CERN. To demonstrate the proper functionality, I constructed an analytically solvable toy model to validate the performance.
The second project was the implementation of the IcINN algorithm and its deeper investigation. Starting from code provided by Anja Butter, I implemented the analytic toy model as well as the two-dimensional physical $Z\gamma\gamma$ example. For the latter, I first had to generate pseudo-data using *MadGraph5* [8], *Pythia8.308* [9] and *DELPHES 3.5.0* [10].

2. Modern Particle Physics

High energy physics is one of the most important research areas in modern physics. The underlying theory -the Standard Model of Particle Physics- has been well established over the last decades, predicting most of the observed effects to a high amount of precision. The success of this model is based on the thorough theoretical groundwork as well as the development of high precision detectors in experiments.
The Standard Model of Particle Physics includes three of the four fundamental forces: the electromagnetic force, the weak force and the strong force. It is in general formulated as a consistent Quantum Field Theory (QFT) which is required to fulfill several symmetries. This field theory predicts observables depending on constants like masses or coupling strengths. In order to obtain an estimation of how valid the theoretical assumptions actually are, a highly accurate measurement of these observables is necessary. This is thus the main objective of modern detector systems.
In recent years, circular colliders like the Large Hadron Collider (LHC) at CERN (Conseil européen pour la recherche nucléaire) contributed to significant discoveries of several types. There are four major experiments at the LHC: *ATLAS*, *LHCb*, *CMS* and *ALICE*. Currently (2022), the third run of data-taking has started.

2.1. The Standard Model of Particle Physics

The Standard Model is based on a Lagrangian which is symmetric under the Poincaré group of the special theory of relativity as well as the combined gauge groups

$$\mathrm{SU(3)_C} \times \mathrm{SU(2)_L} \times \mathrm{U(1)_Y}. \tag{2.1}$$

In addition to these symmetries, the final Lagrangian is required to produce a renormalizable theory, i.e. the predicted observables should be finite. A more involved discussion on how to derive the possible terms of the Lagrangian can be found in advanced literature [11, 12, 13].
The strong interaction is associated with the gauge group $\mathrm{SU(3)_C}$, the corresponding part of the standard model is called Quantum Chromodynamics [14, 15, 16]. The electromagnetic force and the weak force are unified in the Glashow-Salam-Weinberg model [17, 18, 19], their corresponding gauge group is $\mathrm{SU(2)_L} \times \mathrm{U(1)_Y}$. The Standard Model Lagrangian can in general be written down as

$$\mathcal{L}_{\mathrm{SM}} = \mathcal{L}_{\mathrm{Gauge}} + \mathcal{L}_{\mathrm{Fermion}} + \mathcal{L}_{\mathrm{Higgs}} + \mathcal{L}_{\mathrm{Yukawa}}. \tag{2.2}$$

The Gauge part of the Lagrangian is introduced to implement the gauge boson dynamics of each force. Gauge bosons are needed to preserve that the full Lagrangian is invariant under *local* gauge symmetries [11]. The gauge fields associated with each gauge symmetry can be introduced as:

- $\mathrm{SU(3)_C}$: G^a_μ with $a \in \{1, 2, ..., 8\}$,

- $\mathrm{SU(2)_L}$: W_μ^i with $i \in \{1,2,3\}$,
- $\mathrm{U(1)_Y}$: B_μ.

The field-strength tensors are defined as

$$\begin{aligned} G_{\mu\nu}^a &= \partial_\mu G_\nu^a - \partial_\nu G_\mu^a + g_s f^{abc} G_\mu^b G_\nu^c, \\ W_{\mu\nu}^a &= \partial_\mu W_\nu^a - \partial_\nu W_\mu^a + g\epsilon^{abc} W_\mu^b W_\nu^c \\ B_{\mu\nu} &= \partial_\mu B_\nu - \partial_\nu B_\mu. \end{aligned} \tag{2.3}$$

using the structure constant of $\mathrm{SU(3)_C}$ (f^{abc}) and the structure constant of $\mathrm{SU(2)_L}$ (ϵ^{ijk}) as well as the corresponding coupling constants g_s and g. The gauge part of the Standard Model Lagrangian can be written down as [20]

$$\mathcal{L}_{\text{Gauge}} = -\frac{1}{4} G_{\mu\nu}^a G^{a\mu\nu} - \frac{1}{4} W_{\mu\nu}^i W^{i\mu\nu} - \frac{1}{4} B_{\mu\nu} B^{\mu\nu}. \tag{2.4}$$

These terms implement gauge boson dynamics into the Standard Model. At this point it is instructive to have a closer look at the first term of the equation. Due to the non-abelian structure of the group $\mathrm{SU(3)_C}$ and its non-vanishing structure constant f^{abc}, the gauge boson dynamics of the gluon G_μ^a give rise to gluon-gluon self interactions, depicted in Figure 2.1. As explained later, the gluons couple to the fermions, which carry a color charge: the quarks. Experimentally it is observed that there are in general no free quarks. This behavior is explained by the hypothesis of *color confinement*, which states that colored objects are always confined to color singlet states [21]. This assumption is not yet analytically proven, but there is a qualitative explanation. The strong interaction between two quarks is exchanged via gluons. If they would be pulled apart, a "field tube" would form because of the attractive interactions of the gluons due to their self-interaction. This phenomenon is shown in Figure 2.2. In contrast to the electric field, where the field lines are spreading out, the energy density is constant inbetween the quarks [22]. This behavior can be modeled with a potential of the form

$$V(r) \propto r. \tag{2.5}$$

The consequence of this potential is a very large attractive force between any two unconfined quarks, regardless of separation. Hence, the observed hadronic states are all colorless combinations of quarks [21]. This color confinement leads to the effect of *hadro-*

Figure 2.1: Gluon-gluon self interaction for a field theory based on the non-abelian gauge group $\mathrm{SU(3)_C}$: on the left the triple gluon interaction vertex, on the right a quartic gluon interaction vertex.

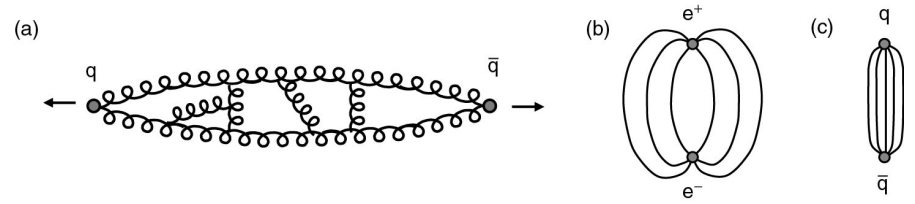


Figure 2.2: Qualitative explanation of color confinement. In Figure (a) it is shown how the gluon self-interactions cause attractive forces between the exchanged gluons of a $q\overline{q}$-pair. Figure (b) and (c) show the resulting field lines of the electromagnetic and strong force, respectively. While in (b) the field lines spread out, in (c) the color field is squeezed into a tube which causes a constant energy density and consequently color confinement. Source: [21].

nisation: by producing a $q\overline{q}$ pair, for example at an electron-positron-collider, single quarks are not observed directly, but only as a *jet*. Separating the $q\overline{q}$ pair leads to a strong field between them with enough energy to perform a pair production of further quarks. This happens repeatedly until the energy of the collision is stored in two jets of hadrons moving in different directions. Like the color confinement, this effect is only described phenomenologically.
An additional result of the gluon-gluon self-interactions is the asymptotic freedom of quarks. The strength of the coupling between quarks and gluons is determined by the strong coupling constant g_s. This value is not actually a constant but rather a running coupling, which is large at the low-energy scale and declines with increasing energy. This behavior is based on the fact that loop corrections need to be applied to the gluon propagator as depicted in Figure 2.3. For non-abelian gauge theories there is always a declining running coupling [11]. For abelian gauge theories (based e.g. on U(1) for photons) this is not the case because the missing self-interaction of the photon forbids the last two diagrams of Figure 2.3. An example for such an increasing coupling would be pure Quantumelectrodynamics (QED). The gauge boson dynamics of the symmetry groups $\mathrm{SU(2)_L}$ and $\mathrm{U(1)_Y}$ will be discussed separately in the context of the Higgs mechanism.

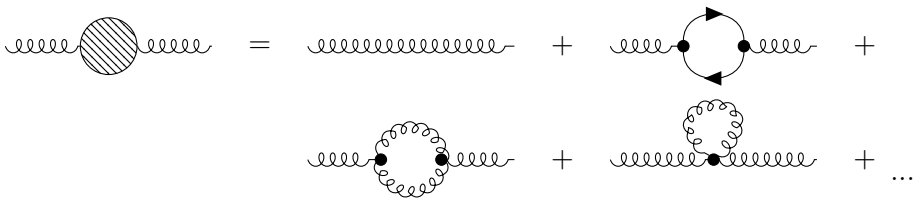


Figure 2.3: 0-loop and 1-loop contributions to the gluon propagator. The fermionic loop (upper right) contributes to an increasing running coupling, but the gluonic loops (lower row) dominate in their contribution to a declining running coupling for g_s. An explicit calculation of these effects can be found in reference [11].

The Fermion part of the Lagrangian implements interactions between the gauge bosons and the fermions which carry their respective charges, for example the strong interaction couples only to fermions carrying a color charge: the quarks. At this point it is implemented that only left-(right-)handed (anti-)fermions couple to the W-boson, this is represented by the group $\mathrm{SU(2)_L}$ and its conserved charge, the third component of the weak isospin I_W^3. Finally, the hypercharge Y is chosen to fulfill the Gell-Mann-Nishijima relation to the electric charge Q of the fermion [23]:

$$Q = I_W^3 + \frac{Y}{2}. \tag{2.6}$$

Experimentally it is found that there are three generations of fermions in the Standard Model, where corresponding fermions in each generation only differ by their masses. The generations of the Standard Model are summarised in Table 1, combined with their representation under the Standard Model symmetry gauge group and their respective charges. To determine the interactions between gauge bosons and fermions, the covariant derivative is constructed using the generators T^a and I_W^i of the gauge groups as well as the coupling constant g' of $\mathrm{U(1)_Y}$:

$$D_\mu = \partial_\mu - ig_s T^a G_\mu^a - igI_W^i W_\mu^i - ig'\frac{Y}{2}B_\mu. \tag{2.7}$$

Fields		Generation			Representation	Charges		
		1st	2nd	3rd		I_W^3	Y	Q
Leptons	$\mathbf{E}'_L$	$\begin{pmatrix}\nu'_e\\ e'\end{pmatrix}_L$	$\begin{pmatrix}\nu'_\mu\\ \mu'\end{pmatrix}_L$	$\begin{pmatrix}\nu'_\tau\\ \tau'\end{pmatrix}_L$	$(\mathbf{1},\mathbf{2})_{-1}$	$\frac{1}{2}$ $-\frac{1}{2}$	-1 -1	0 -1
	$\mathbf{e}'_R$	e'_R	μ'_R	τ'_R	$(\mathbf{1},\mathbf{1})_{\frac{1}{3}}$	0	-2	-1
Quarks	$\mathbf{Q}'_L$	$\begin{pmatrix}u'\\ d'\end{pmatrix}_L$	$\begin{pmatrix}c'\\ s'\end{pmatrix}_L$	$\begin{pmatrix}t'\\ b'\end{pmatrix}_L$	$(\mathbf{3},\mathbf{2})_{\frac{1}{3}}$	$\frac{1}{2}$ $-\frac{1}{2}$	$\frac{1}{3}$ $\frac{1}{3}$	$\frac{2}{3}$ $-\frac{1}{3}$
	$\mathbf{u}'_R$	u'_R	c'_R	t'_R	$(\mathbf{3},\mathbf{1})_{\frac{4}{3}}$	0	$\frac{4}{3}$	$\frac{2}{3}$
	$\mathbf{d}'_R$	d'_R	s'_R	b'_R	$(\mathbf{3},\mathbf{1})_{-\frac{2}{3}}$	0	$-\frac{2}{3}$	$-\frac{1}{3}$

Table 1: The fermions of the three generations of the Standard Model. The representation under the gauge group of the Standard Model is given by $(\mathrm{SU(3)_C}, \mathrm{SU(2)_L})_{\mathrm{U(1)_Y}}$. The weak isospin I_W^3 and the hypercharge Y combine to the electric charge Q as predicted in Equation (2.6). Source: [23].

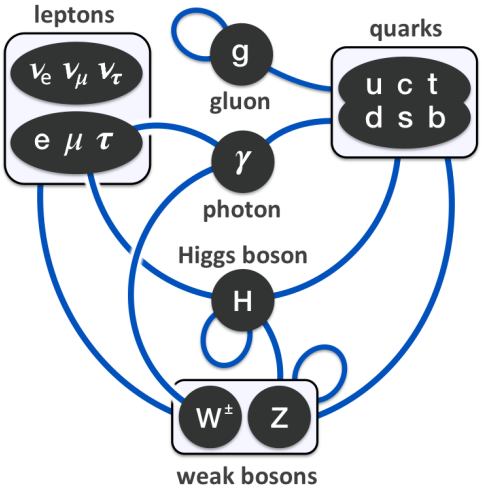


Figure 2.4: Display of the possible Standard Model interactions of gauge bosons with themselves and fermions. Not included is a possible interaction between the Higgs-boson and the neutrinos, since the mass generation process of neutrinos is still unclear. Source: [24].

In general, the fermionic part of the Lagrangian is written as

$$\mathcal{L}_{\text{Fermion}} = \sum_{i=1}^{3} \left(\overline{E}'_{L,i} \not{D} E'_{L,i} + \overline{Q}'_{L,i} \not{D} Q'_{L,i} + \overline{e}'_{R,i} \not{D} e'_{R,i} + \overline{u}'_{R,i} \not{D} u'_{R,i} + \overline{d}'_{R,i} \not{D} d'_{R,i} \right). \quad (2.8)$$

The representation for each fermion under each gauge group needs to be considered with respect to which terms appear in each covariant derivative D. Left handed electrons, muons or tauons are for example part of a color singlett, therefore no interaction with gluons is possible. This part of the Lagrangian gives rise to the fermion dynamics as well as the fermion interactions displayed in Figure 2.4 (exept the Higgs boson interaction which will be discussed in the next paragraph). The corresponding Feynman diagrams can be found in Appendix A.2.

The Higgs part of the Lagrangian is mainly motivated by the need for mass terms in the Lagrangian. In pure QED, naive mass terms like

$$\mathcal{L}_{\text{Mass}} = -m\overline{\psi}\psi = -m(\overline{\psi}_L \psi_R + \overline{\psi}_R \psi_L), \quad (2.9)$$

with specified left- and right-handed fermions are possible. In general, this naive formulation arises by considering that masses are associated with poles of the propagator and are therefore connected to quadratic terms of fields. This path-integral approach to QFT is further explained in [11, 12].
The main problem that arises with these types of mass terms are the different transformation properties of left- and right-handed fermions under SU(2)_L. This means that the

naive mass terms are no longer gauge invariant, which makes them invalid. The solution to this problem is the *Higgs Mechanism* connected with *Spontaneous Symmetry Breaking* [25, 26, 27].
The Higgs field is introduced as a scalar $\mathrm{SU}(2)_\mathrm{L}$ doublet which has the representation $(\mathbf{1}, \mathbf{2})_1$ under the Standard Model gauge group:

$$\Phi(x) = \begin{pmatrix} \phi^+(x) \\ \phi^0(x) \end{pmatrix}. \tag{2.10}$$

For this field the gauge dynamics and the Higgs potential are chosen as

$$\begin{aligned} \mathcal{L}_\mathrm{Higgs} &= \left|D_\mu \Phi\right|^2 - V(\Phi) \\ &= \left| \left(\partial_\mu - igI_W^i W_\mu^i - ig' \frac{1}{2} B_\mu \right) \Phi \right|^2 + \mu^2 (\Phi^\dagger \Phi) - \frac{\lambda}{2} (\Phi^\dagger \Phi)^2, \end{aligned} \tag{2.11}$$

with $\lambda, \mu^2 \in \mathbb{R}$. This Lagrangian is invariant under the gauge transformations of the Standard Model. However, the *vacuum expectation value* ("vev") Φ_0 of the Higgs field corresponds to the minimum of the potential which requires the condition

$$|\Phi_0|^2 = \frac{\mu^2}{\lambda} = \frac{v^2}{2}. \tag{2.12}$$

The vacuum expectation value is in general set to

$$\Phi_0 = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ v + h(x) \end{pmatrix}, \tag{2.13}$$

introducing small deviations from the vev in the form of $h(x)$. The next step is to insert the vev into the Higgs Lagrangian (2.11). The resulting Lagrangian will not be invariant under gauge transformations of $\mathrm{SU}(2)_\mathrm{L} \times \mathrm{U}(1)_\mathrm{Y}$ anymore: these symmetries are "broken". The first part of the Higgs Lagrangian can be calculated as

$$|D_\mu \Phi_0|^2 = \frac{1}{2} \frac{v^2}{4} \left(g^2 (W_\mu^1)^2 + g^2 (W_\mu^2)^2 + (-gW_\mu^3 + g' B_\mu)^2 \right) + \mathcal{O}(h). \tag{2.14}$$

As a result, mass terms for linear combinations of W_μ^i and B_μ are obtained, the mass eigenstates are

$$\begin{aligned} W_\mu^\pm &= \frac{1}{\sqrt{2}} (W_\mu^1 \mp i W_\mu^2), \\ Z_\mu &= \frac{1}{\sqrt{g^2 + g'^2}} (g W_\mu^3 - g' B_\mu), \\ A_\mu &= \frac{1}{\sqrt{g^2 + g'^2}} (g'\ W_\mu^3 + g B_\mu). \end{aligned} \tag{2.15}$$

The first eigenstates correspond to the charged $W^\pm$-boson and the Z-boson. The photon A_μ is chosen as an orthogonal state to Z_μ and fulfills the relation

$$\begin{pmatrix} A_\mu \\ Z_\mu \end{pmatrix} = \begin{pmatrix} \cos\theta_W & -\sin\theta_W \\ \sin\theta_W & \cos\theta_W \end{pmatrix} \begin{pmatrix} B_\mu \\ W_\mu^3 \end{pmatrix}, \tag{2.16}$$

where the weak mixing angle θ_W is defined as

$$\cos\theta_W = \frac{g}{\sqrt{g^2 + g'^2}}. \tag{2.17}$$

After the symmetry breaking, these bosons carry the masses

$$m_W = \frac{v}{2}g, \qquad m_Z = \frac{v}{2}\sqrt{g^2 + g'^2}, \qquad m_A = 0. \tag{2.18}$$

It can be shown that the operator

$$Q := I_W^3 + \frac{Y}{2}, \tag{2.19}$$

still imposes a valid symmetry operation on the new Lagrangian [11]. This comes as no surprise, since there is a conserved electric charge after the spontaneous symmetry breaking. The Higgs mechanism therefore implies the symmetry breaking

$$\mathrm{SU(2)_L} \times \mathrm{U(1)_Y} \Rightarrow \mathrm{U(1)_Q}. \tag{2.20}$$

The three broken symmetry degrees of freedom manifest themselves in the gauge boson masses of $W^\pm$ and Z, i.e. as their longitudinal polarisations. This is predicted in the Goldstone-Boson-Equivalence theorem [11].
In Equation (2.14) the terms containing the excitation of the Higgs field $h(x)$ were neglected. By including $h(x)$ interactions between the Higgs-boson and the massive electroweak gauge bosons are obtained. Additionally, inserting the vev into the Higgs potential raises the Higgs mass as well as the Higgs self-interaction.
At this point it is suitable to revisit the discussion on the gauge part of the Standard Model Lagrangian by plugging in the derived fields from Equation (2.16) into the corresponding part of the gauge Lagrangian (2.4). This procedure results in several interactions inbetween gauge bosons of the electroweak sector. The corresponding Feynman diagrams of the Higgs self-interaction and the electroweak boson interactions are shown in Appendix A.1.

The Yukawa part of the Lagrangian finally introduces the fermion masses in the context of the Higgs mechanism [12]. In a gauge invariant form it is defined as

$$\mathcal{L}_{\text{Yukawa}} = \sum_{i=1}^{3}\left(-\lambda_{e,i}\overline{E}_{L,i}\phi\, e_{R,i} - \lambda_{d,i}\overline{Q}_{L,i}\phi d_{R,i} - \lambda_{u,i}\epsilon^{ab}\overline{Q}_{L,i,a}\phi_b^\dagger u_{R,i}\right) + \text{h.c.}\,, \tag{2.21}$$

with the Yukawa mass couplings λ_i. The charge-conjugate of the Higgs field contains a Levi-Civita-Tensor ϵ^{ab} to preserve gauge invariance. Applying the Higgs mechanism on Equation (2.21), i.e. plugging in the vev, leads to a broken gauge symmetry as well as quadratic terms of fields which are interpreted as mass terms. Each fermion of the Standard Model (with the possible exception of neutrinos) has a non-vanishing Yukawa coupling which determines its mass. There is currently no explanation for the differences

in these couplings, this is known as the *hierarchy problem*. Every massive fermion consequently couples to the Higgs boson, the interaction vertex is given in Figure 2.5.
In general, there is no reason why the mass eigenstates E_L and Q_L of Equation (2.21) and the flavour eigenstates E'_L and Q'_L coupling to the electroweak gauge Bosons via Equation (2.8) should be identical [28]. Hence, a unitary transformation which connects a mass eigenstate f with a flavour eigenstate f' is needed:

$$f_{L,i} = \sum_{j=1}^{3} U^f_{L,ij} f'_{L,j}, \qquad f_{R,i} = \sum_{j=1}^{3} U^f_{R,ij} f'_{R,j}. \tag{2.22}$$

Since these transformations are unitary, they do not affect neutral current interactions. Since neutrino masses are not always considered a part of the Standard Model, the neutrino and lepton transformations are chosen to be equal. Therefore, the charged currents of neutrino-lepton interactions are not affected by this. This idea is of course incomplete since the detection of neutrino oscillations proved the existence of neutrino masses [29, 30]. Deeper theoretical models lead to the construction of the *PMNS-matrix* [31, 32, 33].
The charged current interactions of quarks are affected by these transformations. Since in general the transformation of up-type and down-type quarks according to Equation (2.22) are unequal $U^u_L \neq U^d_L$, the charged current Lagrangian as a part of $\mathcal{L}_{\text{Fermion}}$ reads

$$\mathcal{L}_{cc} \propto \sum_{(i,j)=1}^{3} \left(\overline{u}_{i,L} (U^u_L U^{d\dagger}_L)_{ij} d_{L,j} \not{W}^+ + \overline{d}_{L,i} (U^d_L U^{u\dagger}_L)_{ij} u_{L,j} \not{W}^- \right). \tag{2.23}$$

The additional matrix factor in the charged current can be summarized in the *CKM matrix* V_{CKM} defined as

$$V_{\text{CKM}} = U^u_L U^{d\dagger}_L. \tag{2.24}$$

It can be shown that the CKM matrix for three generations has four degrees of freedom: three rotation angles and one complex phase. This complex phase is the only source of $\mathcal{CP}$-violation in the Standard Model [28].

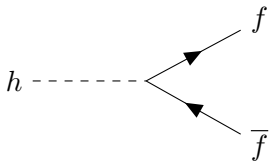


Figure 2.5: Interaction vertex of the Higgs boson with massive fermions. The strength of the coupling is proportional to the Yukawa coupling, i.e. the particle mass.

2.2. Effective Field Theory

The Standard Model of Particle Physics is one of the most successful physical theories ever probed. Nevertheless there are still loose ends which cannot be explained: dark matter [34], neutrino masses [29] or the strong CP problem [35], to mention a few. The LHC and especially the experiments ATLAS and CMS have majorly been designed to probe the electroweak sector of the Standard Model and discover the Higgs boson. After the discovery of the Higgs boson [36, 37] there were no signs of further fundamental particles in any direct searches, i.e. no resonant production of a heavy particle.
One reason for the missing signs of further particles could be their mass. Several theories like Supersymmetry [38] or the Seesaw mechanism [39] propose particles with masses way beyond the electroweak scale, i.e. current colliders cannot produce them. Nevertheless, these particles could have an influence on other processes, an *indirect search* is therefore possible. Such indirect knowledge is ambiguous in the way that it can be the low energy-implication of several models. The main reason for this is the *decoupling theorem* which states how the non-analytic structure of correlation functions due to heavy states are projected out onto a low energetic *effective field theory* (EFT) [40, 41].

A historic example for an EFT is the Fermi theory of weak interaction [21]. The W-boson propagator can be derived from the gauge part of the Lagrangian 2.4 to be

$$\frac{-i}{q^2 - m_W^2}\left(g_{\mu\nu} - \frac{q_\mu q_\nu}{m_W^2}\right). \tag{2.25}$$

In the low-energy approximation $|q^2| \ll m_W^2$, which is valid for most particle decays this simplifies to

$$i\frac{g_{\mu\nu}}{m_W^2}. \tag{2.26}$$

Physically, this simplification can be understood as the reduction to a point-like interaction as shown in Figure 2.6. This description by Fermi [42] therefore considered the weak interaction to be a force without range. As shown later with the success of electroweak unification and the subsequent discovery of the $W^\pm$- and Z-boson, this approach was too simplistic. The theory proved useful to calculate for example muon decay rates more precisely. It was hence beneficial up to the point where colliders reached higher energies.
As described earlier particle physics is currently in a similar position: there are no signs of potential heavy particles, so it might be worthwhile to look at their low-energy implications. This can be done by integrating out the heavy particles to obtain a correction to a Standard Model interaction or a completely new interaction.
The *Standard Model Effective Field Theory* (SMEFT) is an approach to extend the Standard Model in the most general way possible. The idea is to add terms in the Standard model Lagrangian $\mathcal{L}_{\text{SM}}$ which are constructed of gauge-invariant, $(d > 4)$-dimensional operators $Q_i^{(d)}$ built out of the Standard Model fields introduced earlier. Explicitly the Lagrangian is expanded as

$$\mathcal{L}_{\text{SMEFT}} = \mathcal{L}_{\text{SM}} + \mathcal{L}^{(5)} + \mathcal{L}^{(6)} + \mathcal{L}^{(7)} + \mathcal{L}^{(8)} + ..., \tag{2.27}$$

using

$$\mathcal{L}^{(d)} = \sum_{i=1}^{n_d} \frac{C_i^{(d)}}{\Lambda^{(d-4)}} Q_i^{(d)}, \qquad \text{for } d > 4. \tag{2.28}$$

In this equation the operators $Q_i^{(d)}$ are suppressed by $(d-4)$ powers of the cut-off scale Λ, which is well above the electroweak energy scale and marks the scale at which SMEFT breaks down [40]. The Wilson coefficients $C_i^{(d)}$ are determined by a theory containing heavy particles or mediators at higher energy scales than currently investigated. The aim is therefore to determine their values experimentally or at least find restrictions for them.
It can be shown [43] that the invariant operators constructed from the SM fields satisfy

$$\frac{1}{2}(\Delta B - \Delta L) = d \bmod 2, \tag{2.29}$$

with the change of the baryon number ΔB and the change of the lepton number ΔL for the interaction the operator is implying. For uneven d this implies that the interactions either violate lepton number conservation or baryon number conservation. There are ideas to implement neutrino masses in $\mathcal{L}^{(5)}$ but this is beyond the scope of this thesis.
For three generations there are already numerous contributions by the six-dimensional component $\mathcal{L}^{(6)}$: there are 2499 independent baryon-number conserving operators [44] and 546 baryon-number violating operators [45]. This formulation of SMEFT is the so-called *Warsaw basis* as introduced in reference [46]. For $\mathcal{L}^{(8)}$ there are 44 807 SMEFT operators, a complete list of them can be found in [47].

To estimate the Wilson coefficients processes sensitive to altered or new interactions introduced by $\mathcal{L}^{(6)}$ or $\mathcal{L}^{(8)}$ are studied. This thesis investigates the general process

$$pp \to Z\gamma\gamma, \qquad Z \to \mu^+\mu^-. \tag{2.30}$$

An example for a Feynman diagram at leading order in the pure Standard Model is given in Figure 2.7 on the left. The inclusion of additional SMEFT terms can have two effects: either an already existing interaction vertex gets an additional contribution or a completely new interaction vertex is introduced. Couplings that were not present in the

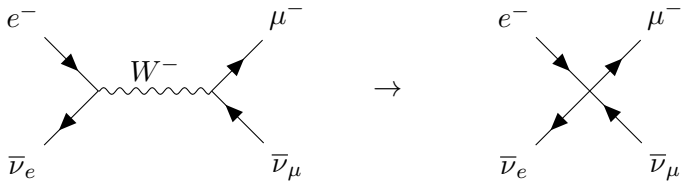


Figure 2.6: Visualisation of Fermi theory as an EFT of the weak interaction. The W-boson propagator is reduced to a point-like interaction. Source: [21].

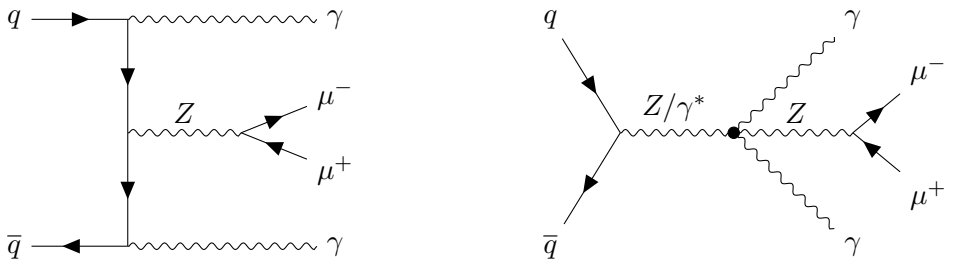


Figure 2.7: Feynman diagrams for the process $q\bar{q} \to Z\gamma\gamma$ with Z decaying into muons. On the left the Standard Model leading-order process is shown, on the right additional contributions that appear after including the EFT operator $Q_{T,8}^{(8)}$ enabling the anomalous quartic gauge couplings $ZZ\gamma\gamma$ and $Z\gamma\gamma\gamma$.

Standard model are called *Anomalous Gauge Couplings*, for dimension six or eight there are anomalous triple gauge couplings (aTGGs) or anomalous quartic gauge couplings (aQGCs).
For the process at hand the aTGC introduced by the dimension-six operators do not contribute to the production of $Z\gamma\gamma$. The aQGCs introduced via the dimension-six operators are also not contributing to this process since they do not give rise to purely neutral aQGCs [48].
Hence, to construct a larger SMEFT contribution a dimension-eight operator is needed

$$\mathcal{L}_{T,8} = \frac{C_{T,8}^{(8)}}{\Lambda^4} B_{\mu\nu} B^{\mu\nu} B_{\alpha\beta} B^{\alpha\beta}. \tag{2.31}$$

For simplicity all Wilson coefficients in the simulation are set to zero except $C_{T,8}^{(8)}$. $\mathcal{L}_{T,8}$ introduces aQGCs of the neutral electroweak gauge bosons: $ZZZZ$, $ZZZ\gamma$, $ZZ\gamma\gamma$, $Z\gamma\gamma\gamma$ and $\gamma\gamma\gamma\gamma$ [49, 50]. The last three interactions lead to additional Feynman diagrams for the production of $Z\gamma\gamma$, they are displayed in Figure 2.7 on the right.

2.3. The ATLAS Experiment at the LHC

The ATLAS (originally "A Toroidal LHC ApparatuS") experiment is one of the four main experiments at the LHC at CERN. The main purpose of the ATLAS detector was the discovery of the Higgs-boson, which was accomplished in 2012 [36, 37]. Today the ATLAS detector is used as a multi-purpose detector to investigate multiple research topics like precision measurements of masses, top quark properties or b-meson physics. One of the most recent discoveries was the measurement of a potential four-charm tetraquark [51]. In the following an overview over the LHC in general as well as the ATLAS detector in specific is given.

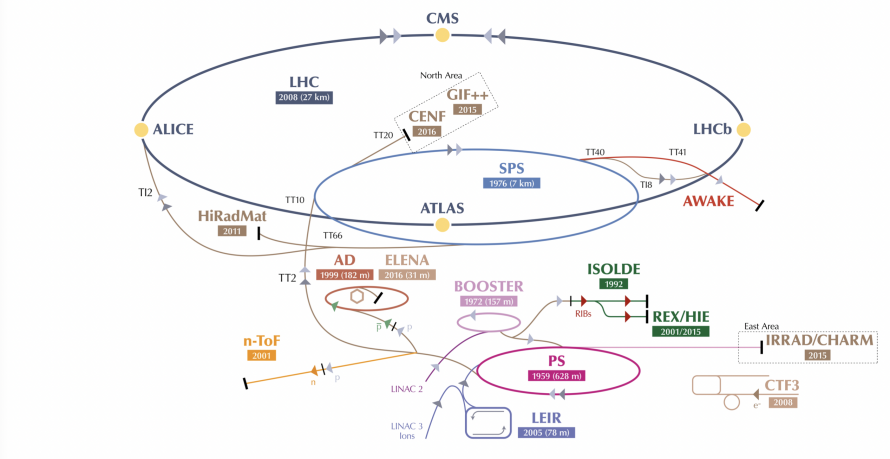


Figure 2.8: Overview of the experiments and the accelerator complex at CERN during Run 2. Source: [54].

2.3.1. The Large Hadron Collider (LHC)

The LHC [52] is the largest and most powerful particle accelerator ever built. The aim is to study processes of the Standard Model in great detail by colliding accelerated protons or heavy ions at center-of-mass-energies up to $\sqrt{s} = 13.6\,\text{TeV}$ and an instantaneous luminosity of $10^{34}\,\text{cm}^{-2}\text{s}^{-1}$. The structure of the accelerators located at CERN is shown in Figure 2.8. The LHC itself has a circumference of 26.659 km and is located $\approx 100\,\text{m}$ below the surface. To reach the center of mass energy of 13.6 TeV protons need to be pre-accelerated. As an example, during Run 2 the LINAC2 (LINear ACcelerator 2) started the acceleration chain by accelerating negative charged hydrogen ions to 160 MeV. After this the electrons are stripped of the hydrogen ions and the remaining protons are further accelerated in the Proton Synchrotron Booster (PSB), the Proton Synchrotron (PS) as well as the Super Proton Synchrotron (SPS). After reaching 450 GeV the protons are injected into the LHC beam pipe, where they are accelerated to their final energy. During Run 2 each beam consisted of about 2800 bunches with up to 10^{11} protons per bunch, since 2015 the collisions are timed 25 ns apart from each other [53].
The tunnel of the LHC is constructed with eight straight segments and eight curved ones and therefore not perfectly round. To keep the protons on this trajectory there are over 1200 superconducting dipole magnets which are operated at a current of 12 000 A and a temperature below 2 K to create a magnetic field of more than 8 T. To reach temperatures this low the magnets have to be cooled continuously using liquid nitrogen as a pre-cooling and superfluid helium to reach the final temperature. The strength of the magnets is the main limiting factor for the final kinetic energy. The beam pipes themselves contain a vacuum at approximately 10^{-13} bar [55].
The beams cross at the four interaction points where the experiments are placed. To connect the number of expected events N_{Events} for a specific process in one of the detectors with the instantaneous luminosity L, the cross-section σ is introduced, which is an ex-

pression of the underlying quantum field theoretical probability that such an interaction will occur. It can be shown [21] that

$$\frac{\mathrm{d}N_{\mathrm{Events}}}{\mathrm{d}t} = \sigma L. \tag{2.32}$$

The total number of events can be calculated by integrating over t.
The four main detector systems at CERN have all been designed for specific purposes. *ALICE* (A Large Ion Collider Experiment) [56] is designed to investigate quark-gluon-plasmas which are created by the collision of heavy nuclei, often lead (Pb). There are theories that similar conditions have existed in the early stage of the universe. The *LHCb* (Large Hadron Collider beauty) [52] is a proton-collision experiment to investigate heavy-flavour physics, especially b-mesons. Connected to this is precise measurement of the CKM-phase causing $\mathcal{CP}$-violation, which might be an explanation for the matter-antimatter asymmetry of the universe. *CMS* (Conducting Muon Solenoid) [57] represents together with the ATLAS experiment the general purpose detectors of the LHC. The main difference between them is the used detector technology. As already given by the name, CMS contains a huge solenoid magnet and a very precise muon measurement system. Its main purpose was the search for the Higgs-boson, today it is used for lots of analyses like precision measurements of the top-sector and the search for signs of supersymmetry.

2.3.2. The ATLAS Detector

The ATLAS experiment [58] is a cylindrical detector system with a length of 44 m and a diameter of 25 m. The main challenges for the detector systems are the high particle multiplicities in connection with the high interaction rates as well as the radiation doses the individual components are exposed to. During Run 2 of the LHC, the ATLAS detector recorded an integrated luminosity [58] of

$$\tilde{L} = \int \mathrm{d}t L = 139\,\mathrm{fb}^{-1}. \tag{2.33}$$

There are four main components of the ATLAS detector: the Magnet System, the Calorimeters, the Muon Chambers and the Inner Detector. An overview of the ATLAS detector is given in Figure 2.9.

The Magnet System [59] and its design is the fundamental choice that drives the rest of the detector design. A thin superconducting solenoid [60] surrounds the inner detector to create a magnetic field up to 2 T, which causes curved trajectories for charged particles. In addition, there are three large superconducting toroids (one barrel, two end-caps) arranged around the calorimeters with an eight-fold azimuthal symmetry. The magnetic field created by the toroid system has a strength up to 3.5 T. This is especially relevant for the measurement of muon momenta.

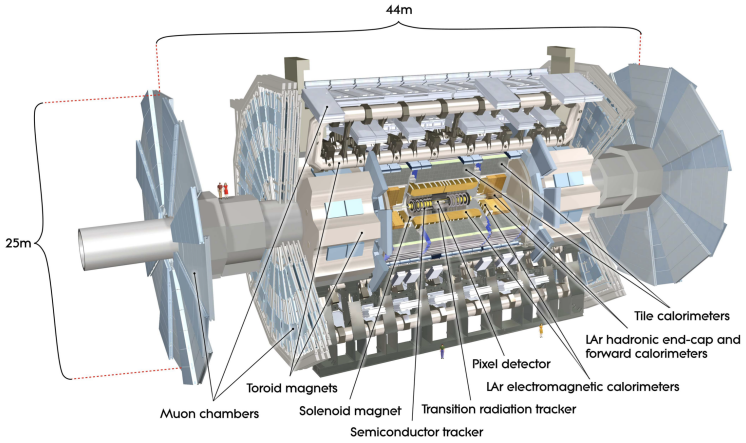


Figure 2.9: Overview of the ATLAS detector and its subcomponents. Source: [58].

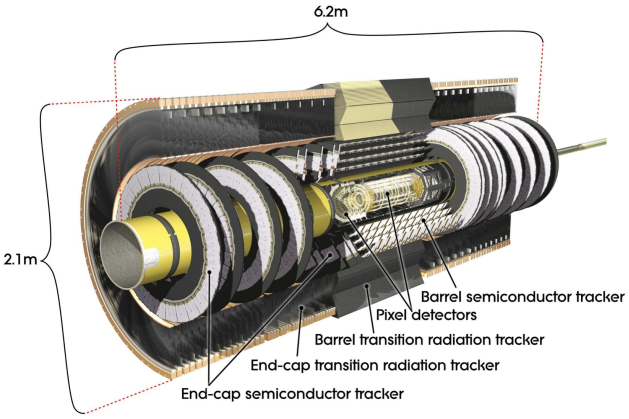


Figure 2.10: Cut-away view of the Inner Detector of the ATLAS experiment. Source:[58].

The Inner Detector [61] is designed to measure the track and the momentum of electrically charged particles. A schematic of the inner detector is shown in Figure 2.10. To achieve the required resolution, a semiconductor tracker (SCT) [62] consisting of pixels in the inner layers and a silicon strip detector in the outer layers are used, followed by a transition radiation tracker (TRT) [63]. The SCT achieves high granularity and precision, in addition it is easily replaceable in a long shutdown, which is advantageous because of the high radiation dose it is exposed to. The applied magnetic solenoid field causes a curved trajectory for charged particles which can be used for charge identification. The SCT can also be used for full particle identification since the differential energy $\mathrm{d}E/\mathrm{d}x$ can be determined and matched to a Bethe-Bloch-curve. The attached TRT helps to distinguish electrons and charged hadrons via the γ-factors of the transition radiation. This is especially interesting to distinguish electrons and charged pions.

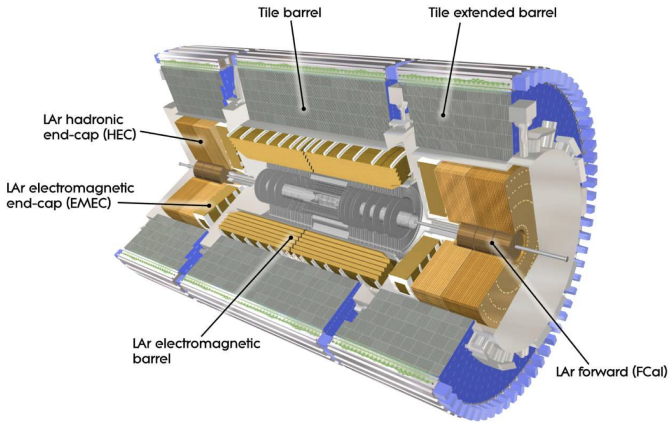


Figure 2.11: Cut-away view of the calorimeter system at ATLAS containing an electromagnetic and a hadronic calorimeter. Source: [58].

The Calorimeter System [64] of ATLAS contains an electromagnetic (EM) calorimeter as well as a hadronic calorimeter. Their setup is displayed in Figure 2.11. The main task of the calorimeters is to provide a reliable energy measurement of electromagnetic and hadronic showers as well as making sure that the majority of these particles are absorbed. Therefore, calorimeter depth is an important consideration. The calorimeters used in ATLAS are *sampling calorimeters*, i.e. they use a combination of absorbing layers to restrict the particle showers arising from collision, as well as active layers in which the signal is produced. The advantage of sampling calorimeters is that each material can be chosen to be well-suited for their task. For example an absorbing material with a high density can be used to produce a shower that evolves fast in a limited space, although the material might be unsuitable for measuring the deposited energy of the shower [65]. The EM calorimeter [66] uses lead as an absorbing and liquid argon as an active material. It consists of two endcaps as well as a central barrel calorimeter. For a better performance the electromagnetic calorimeter shares a vacuum vessel with the central solenoid. The lead plates are arranged in an accordeon-shape with the liquid argon inbetween. This special geometry provides a full ϕ symmetry without any azimuthal cracks. The main purpose of the EM calorimeter is a high resolution for photon and electron energies as well as the absorption of the mainly electromagnetic showers caused by them. To ensure this the total thickness of the EM calorimeter is approximately 22 radiation lengths X_0 in the barrel and 24 X_0 in the end-caps [58].
The hadronic calorimeter [67] is placed around the EM calorimeter and consists of several components as can be seen in Figure 2.11. The tile calorimeter (barrel and extended barrel) uses steel as absorber and scintillating tiles as an active material. In addition to these calorimeters, there are the hadronic end-cap calorimeters (HEC) and the forward calorimeters (FCal). HEC is placed directly behind the EM end-cap calorimeters and is therefore overlapping in its measurement region with the tile calorimeter and FCal. The absorbing material is copper, the active material is liquid argon. It is therefore

convenient to share a common liquid argon cryostat with the EM calorimeter endcaps. The FCal is placed very close to the beam itself and is therefore exposed to high energetic radiation. The consequence is a high-density choice for the absorbing material: copper and tungsten. The active material is again liquid argon. In general the hadronic calorimeters are designed to initiate and measure hadronic jets, as well as absorbing most of their components. The main difficulty compared to electromagnetic calorimeters is the unknown ratio of electromagnetic to hadronic particles in the measured jet. This is also the reason why electromagnetic calorimeters in general have a better energy resolution than hadronic calorimeters [65].
With this calorimeter design a nearly 4π-coverage of the calorimeter system is achieved. In addition, nearly all particles coming from the collision are absorbed, except of muons and neutrinos.

The Muon Spectrometer [68] is used to identify and measure the remaining muons using the magnetic field provided by the toroid magnet system. Its setup is shown in Figure 2.12. The toroid produces a magnetic field which is mostly orthogonal to the trajectory of the muons. The measurement of the deflection in trajectory implicitly allows a momentum estimation. To perform these measurements a combination of precision tracking detectors as well as trigger chambers are used. The precision tracking detectors are monitored drift tubes (MDT) and cathode strip chambers (CSC). The MDT and CDC mainly measure the track coordinates in the principal bending direction of the magnetic field. In addition to this, the orthogonal muon coordinate is measured with the trigger chambers consisting of resistive plate chambers (RPC) and thin gap chambers (TGC). Furthermore the trigger chambers provide bunch-crossing identification and well-defined p_T-thresholds. In general the muon spectrometer is optimized to provide a high-resolution muon momentum measurement with an almost 4π-coverage [58, 69].

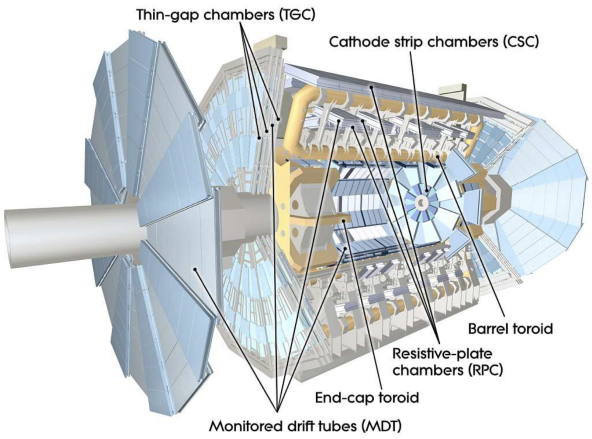


Figure 2.12: Cut-away view of the muon spectrometer system at ATLAS. Starting at a radius of 4.25 m and reaching out to the full radius of the detector it is the largest subdetector in terms of volume. Source: [58].

2.4. Observables

The interacting objects in a proton collider are the point-like constituents of the proton: the partons. The underlying idea is the parton model, which was Richard Feynman's attempt to explain the Bjorken scaling at the SLAC experiment in the 1960s [70]. In this model he considers the partons to be in an infinite momentum frame (masses can be neglected), where the whole parton energy comes from the proton momentum in beam direction. Every parton k carries a fraction x_k of the nucleons momentum p, i.e.

$$p_k = x_k p, \qquad \sum_k x_k = 1. \tag{2.34}$$

Today the partons of the proton are identified with the quarks and gluons. Since the quarks inside the proton can interact via gluons they are today described with a probability distribution rather than a fixed momentum. These probability distributions are called *parton distribution functions* (PDFs). The proton mainly consists of three valence quarks (uud). Nevertheless the gluon dynamics between them enables quark pair-productions, as a consequence more massive quarks (c, s) can be found in the proton. These quarks are called sea-quarks and are also described using PDFs.
The fundamental shapes of the PDFs depend on the detailed dynamics of the proton and are a priori not known, experimentally it has been shown that they depend on the amount of transfered momentum Q^2 (*scaling violations*) [21]. One way to construct the PDFs is by starting from a parametrisation of non-pertubative PDFs at a low Q^2-scale and fitting them to experimental data obtained from electron-proton scattering (e.g. HERA [71]). Although a prediction of the PDFs themselves from first principles is not possible, the *DGLAP* equations [72, 73, 74] describe their Q^2-dependence and therefore allow to calculate them for every Q^2. The resulting PDFs depend on multiple parameters, for example the chosen order of perturbation, the choice of the input data, the treatment of heavy quarks or the correlation between g_s and the PDFs. The DGLAP equations are based on parton splitting functions for the QCD processes $q \to qg$ and $g \to q\bar{q}$, their (rather involved) derivation can be found in [75].

At this point it is suitable to introduce several kinematic observables which are measured by the ATLAS detector. The 4-momentum $p_\mu = (E, \mathbf{p})$ of a measured particle is defined in a spherical coordinate system with z-axis in beam direction. Starting from the momentum $\mathbf{p} = (p_x, p_y, p_z)$ the transversal momentum is defined as

$$p_T = \sqrt{p_x^2 + p_y^2}, \tag{2.35}$$

as well as the azimuthal angle

$$\phi = \arctan\left(\frac{p_y}{p_x}\right). \tag{2.36}$$

In a collider process with two protons the actual interaction happens on *parton level*. Since each of the two colliding partons carry fractions x_1 and x_2 of the proton energy E_p,

the net longitudinal momentum of the colliding parton-parton system with respect to the center of mass system of the protons is given by $(x_1 - x_2)E_p$. Consequently the final state system is boosted along the beam axis. This boost is the reason to construct only observables invariant under these boosts in beam direction, which was easily possible with p_T and ϕ. For this reason the missing angle θ between beam direction and particle direction is often expressed in terms of the rapidity

$$y = \frac{1}{2} \ln\left(\frac{E + p_z}{E - p_z}\right). \tag{2.37}$$

It can be shown that differences in the rapidity are invariant under Lorentz transformations along the beam direction [21]. For high-energy jets where the jet mass is negligible the pseudorapidity

$$\eta = -\ln\left(\tan\left(\frac{\theta}{2}\right)\right), \tag{2.38}$$

is often used instead of the rapidity y.
Another important observable for this project is the p_T value of a Z-boson which decays into a muon-antimuon pair. It is possible to express p_T^Z as a function of the measured p_T- and ϕ-values of the muon and antimuon. Assuming energy and momentum conservation in the decay p_T^Z is written as

$$p_T^Z = \sqrt{(p_x^Z)^2 + (p_y^Z)^2} = \sqrt{(p_x^- + p_x^+)^2 + (p_y^- + p_y^+)^2}, \tag{2.39}$$

with $\mathbf{p}^-$ and $\mathbf{p}^+$ representing the muon and antimuon momenta. The x- and y-momentum can be reexpressed in terms of ϕ and p_T as

$$p_x = p_T \cos(\phi), \qquad p_y = p_T \sin(\phi). \tag{2.40}$$

Using this for the respective muon momenta in Equation (2.39) and with a trigonometrical theorem the final result reads

$$p_T^Z = \sqrt{(p_T^-)^2 + (p_T^+)^2 + 2\,(p_T^-)(p_T^+)\cos(\phi^- - \phi^+)}\,. \tag{2.41}$$

3. Unfolding

One of the central tasks of modern particle physics is to test the assumptions of the Standard Model by measuring its predictions. The predictions of the Standard Model are manifested in observables like particle distributions, particle energies or decay times. With Monte Carlo event generation (*MadGraph5* [8] and *Pythia8.308* [9]), it is possible to produce predictions for the event distributions of observables. The generated events are referred to as *truth-level* or *particle-level* events. Real experimental data cannot be measured arbitrarily precise because of the limited measuring abilities of detector systems. These detector effects exist due to resolution inaccuracies, non-linear responses of detector components, limited acceptance, missed particles or mislabeling. The measured events are referred to as *reconstruction-level* or *detector-level* events.
To probe a specific theory or model, it is necessary to compare the predicted and measured observables on the same level. To compare them on detector-level a detector simulation is applied to the generated events on truth-level, which is often referred to as *folding*, i.e. a convolution with the detector response function. This can be done in the case of experiments like ATLAS with *DELPHES 3.5.0* [10]. In general, this is the easier way to compare theory and experiment. To compare event distributions on truth-level a more complicated procedure is needed: the removal of the detector effects from the measured data, called *unfolding* (see Figure 3.1). This is done by finding a (pseudo-)inverse to the detector response function, which requires a deep knowledge of the detector and its effects. In general it is preferable to publish the measured data along with its detector response function, since a detector simulation is easier than an unfolding [1].

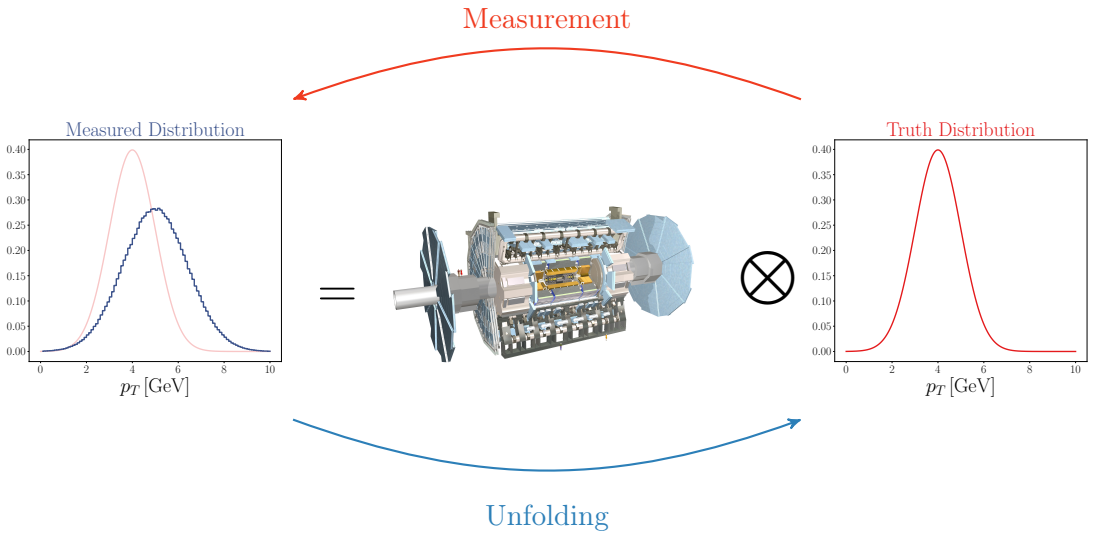


Figure 3.1: General idea of Unfolding as an inversion of a measurement. A truth distribution (red, right) is measured using a particle detector (blue, left). Unfolding inverts this measurement to reobtain the truth distribution [76].

Nevertheless, it is sometimes necessary to obtain truth-level data via unfolding, for example to compare results of different experiments with different response functions. Another reason to desire truth-level data is that modern theories in particle physics usually contain free parameters. Some parameters do not have obvious limitations, a scan of the model parameter space is therefore necessary. These scans are, in general, better applicable on truth-level. A third reason for unfolding is the complexity of modern detector response functions, which might be used in a wrong way by future generations of scientists. In general, it is an individual decision whether or not to unfold, a more involved discussion can be found in more involved literature [1].
With the assumption that it is decided to unfold the experimental data, there are several algorithms that have proven to be useful. On the one hand there are the "classical", matrix-based algorithms which have been used for several years [2, 3, 4], on the other hand there are machine learning algorithms to unfold experimental data which have been developed more recently [5, 77, 78].

3.1. Matrix-based Algorithms

As a start a naive mathematical formulation of the unfolding (following [1, 79]) is introduced. Let $f(t)$ be the true underlying function ehich determines the event distribution. Instead of observing $f(t)$ directly, only a measured distribution of events $g(r)$ is observed, which is the result of convoluting the true function with an abstract detector response function $R(r|t)$.

$$g(r) = \int R(r|t) f(t)\, \mathrm{d}t. \tag{3.1}$$

The detector response function R has to be constructed with a Monte Carlo simulation, which is based on real test beam measurements. The goal of unfolding is to obtain the best estimate for $f(t)$. Formally, the true distribution can be obtained from the measured distribution via

$$f(t) = \int \tilde{R}(t|r) g(r)\, \mathrm{d}r, \tag{3.2}$$

where $\tilde{R}(t|r)$ is a (pseudo-) inverse detector response function, i.e. the conditional probability density at truth-level given a measured event t. To obtain an inverted detector response function there are several unfolding algorithms.

In the standard unfolding methods the distributions $f_{\text{true}}(t)$ and $g_{\text{meas}}(r)$ are converted into histograms $\mathbf{t}$ and $\mathbf{r}$, which introduces a binning. The size of this binning is already an important choice. Too small bins lead to large bin-to-bin correlations, while too large bins do not resolve interesting structures. Equation (3.1) can be rewritten as

$$r_i = \sum_j R_{ij} \cdot t_j, \tag{3.3}$$

where r_i and t_j are the number of events in the i'th and j'th bin of the respective histogram. i is always a detector-level index, j always a truth-level index. The detector response matrix R_{ij} is a binned representation of the continuous response function $R(r|t)$. The response matrix is constructed using a Monte Carlo simulation with truth-level events $\tilde{\mathbf{t}}$, which are mapped to detector-level events $\tilde{\mathbf{r}}$. With no inefficiencies in the model a generated event will appear in a certain bin j on truth-level as well as in a bin i on detector-level. Since this is a statistical process there are events which contribute to the same bin on truth-level but to different bins on detector-level and vice versa. Given that there are enough Monte Carlo events, it is possible to construct the migration matrix A. A single entry of this matrix, A_{ij}, counts the number of events which correspond to truth-level bin j and detector-level bin i in the Monte Carlo simulation. The detector response matrix is characterized by its probabilistic interpretation as

$$R_{ij} = P(\text{observed in bin } i \,|\, \text{true value in bin } j). \tag{3.4}$$

Without inefficiencies, the sum over all possible bins of the observed value is normalized

$$\sum_i R_{ij} = P(\text{observed in any bin} \,|\, \text{true value in bin } j) = 1. \tag{3.5}$$

The detector response matrix is derived by normalizing the migration matrix for a truth-level event as

$$R_{ij} = \frac{A_{ij}}{\tilde{t}_j} = \frac{A_{ij}}{\sum_k A_{kj}}. \tag{3.6}$$

The detector response matrix is dependent on the Monte Carlo simulation. To unfold a given measured distribution, the detector response matrix needs to be inverted and applied to the measured histogram.

Simple Matrix Inversion considers the inversion to be a purely mathematical problem: the inversion of the detector response reduces to the inversion of the detector response matrix. The corresponding binned equation for Equation (3.2) can be written as

$$r_j = \sum_i R_{ji}^{-1} \cdot t_i. \tag{3.7}$$

The idea of a simple matrix inversion has several flaws. In general, it is possible that small differences between simulation and experiment introduce large bin-to-bin correlations and fluctuations. This phenomenon is called the *high frequency problem* [1]. A visualization of the phenomenon is illustrate in Figure 3.2. The truth distribution (left) has two sharp peaks in a generic observable (here p_T), while a detector measurement (center) smears the distribution and the peaks are less sharp. The effect of the detector response is therefore to smear out any fine structure. However, its inverse will consequently introduce fine structure. The distribution measured in experiments is in general different to the Monte

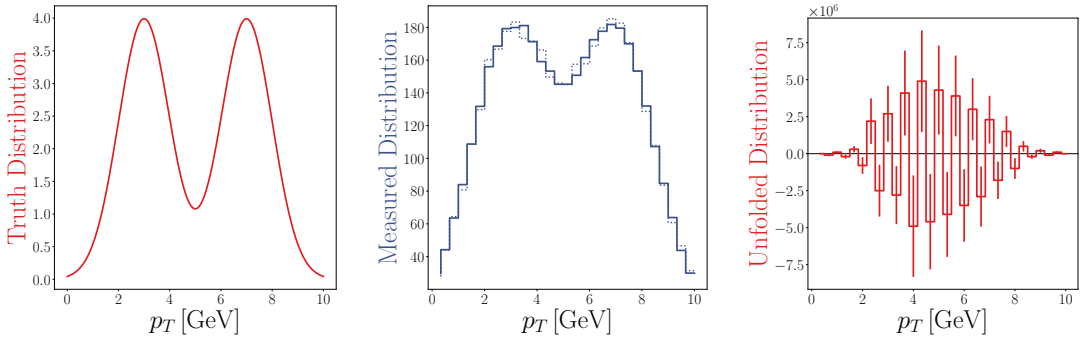


Figure 3.2: Example for a high-frequency problem. A truth distribution with two peaks (red, left) is measured with a detector which smears out fine structure (blue dotted line, center). The response matrix was constructed upon a Monte Carlo detector-level distribution which looked slightly different than the detector simulation (blue solid line, center). The application of the inverse of the response matrix to the measured distribution results in a highly fluctuating distribution with huge uncertainties (red, right). Source: [1].

Carlo simulation since there are statistical fluctuations. Applying the inverted detector response matrix to the experimentally obtained distribution (dotted in the center) leads to a heavily fluctuating unfolded distribution (right) with large uncertainties. Due to the introduction of fine structure through the inverted detector response, the unfolded distribution shows such a behavior. It is problematic that the negative entries of the unfolded distribution are unphysical, they appear because a true inversion of the detector response matrix needs to contain negative entries. Although this unfolding example is mathematically correct it is obviously not the desired outcome. Nevertheless, the unfolded distribution has two desirable features: it has zero bias and a minimum variance. A proof of this can be found in [1]. Using a regularization is possible to smooth out the distributions or to reduce the variances, but this will always result in a bias towards the Monte Carlo simulation. This trade-off between a small bias and an acceptable uncertainty is managed with the strength of the regularization.

Singular Value Decomposition (SVD) [2] is one of the most-used unfolding algorithms. The solution to the simple matrix inversion is the maximum likelihood solution if independent Poissonian fluctuations are assumed in each bin. In this case the log-likelihood function is

$$\ln L(\hat{\mathbf{t}}) = \sum_i \left(r_i \ln(\textstyle\sum_j R_{ij}\hat{t}_j) - \ln(\textstyle\sum_j R_{ij}\hat{t}_j) - \ln(r_i!) \right), \tag{3.8}$$

with the response matrix R_{ij}, the measured distribution r_i and the estimator of the bin means of the true histogram $\hat{t}_j$. To maximise the log-likelihood, its derivative with respect to each $\hat{t}_j$ has to be zero. The solution for the maximum likelihood estimator is

given as

$$r_i = \sum_j R_{ij}\hat{t}_j \qquad \Leftrightarrow \qquad \hat{t}_j = R_{ji}^{-1} r_i, \tag{3.9}$$

under the assumption that R is invertible. To avoid the high frequency problem, a term is added to the maximum likelihood in order to achieve smoothness of the resulting unfolded distribution. The new maximum likelihood estimator reads

$$\phi = \ln L(\hat{\mathbf{t}}) + \tau S(\hat{\mathbf{t}}), \tag{3.10}$$

with a regularization parameter τ and the Tikhonov regularization [80]

$$S(\hat{\mathbf{t}}) = \sum_j \left[(\hat{t}_{j+1} - \hat{t}_j) - (\hat{t}_j - \hat{t}_{j-1})\right]^{-2}. \tag{3.11}$$

This additional term keeps "the integrated square of the second derivative of $\hat{t}$ constrained" [81]. Therefore, the unfolded distribution is required to be smooth and the oscillating behavior is reduced. The strength of the regularization is determined by the parameter τ: if τ is chosen too small, the unfolded distribution will oscillate too much, if τ is too large, the result will be strongly biased towards the Monte Carlo simulation. The actual implementation of this regularized log-likelihood is efficiently executed using the algorithm by Höcker and Kartvelishvili [2]. In this algorithm an additional rescaling as well as the name-giving *singular value decomposition* (SVD) of the rescaled response matrix are used. A SVD is a factorisation of a matrix in two unitary and a diagonal matrix which contains the eigenvalues. Further information about this can be found in their paper [2].

Iterative Bayesian Unfolding (IBU) [3] is another possibility to obtain a biased unfolded distribution with reasonable uncertainties. If the detector response function is interpreted as a conditional probability $R_{ij} = P(r_i|t_j)$, the posterior distribution calculated with Bayes' theorem can be written as

$$\tilde{R}_{ji} = P(t_j|r_i) = \frac{P(r_i|t_j)P(t_j)}{P(r_i)} = \frac{P(r_i|t_j)P(t_j)}{\sum_k P(r_i|t_k)P(t_k)}, \tag{3.12}$$

using Equation (3.1) and defining the probability distributions of the true and reco distribution as

$$P(t_j) = \frac{t_j}{\sum_k t_k} \qquad \text{and} \qquad P(r_i) = \frac{r_i}{\sum_l r_l}. \tag{3.13}$$

With no inefficiencies each event on truth level corresponds to one event on reco level. Therefore it is possible to use $\sum_k t_k = \sum_l r_l$ to rewrite

$$\tilde{R}_{ji} = \frac{R_{ij}t_j}{\sum_k R_{ik}t_k}. \tag{3.14}$$

In this algorithm the dilemma of ill-defined inverse problems can be seen: in order to obtain the posterior, knowledge about the true underlying distribution is needed in the first place. The basic idea of Iterative Bayesian Unfolding is to assume a distribution for t_j, which is based on a prior belief of the unfolding result. A usual choice of this *prior* is the truth component of the Monte Carlo simulation $\tilde{t}_j$. Using Equation (3.6), $\tilde{R}_{ji}$ is reexpressed as

$$\tilde{R}_{ji} = \frac{R_{ij}\tilde{t}_j}{\sum_k R_{ik}\tilde{t}_k} = \frac{A_{ij}}{\sum_k A_{ik}} = \frac{A_{ij}}{\tilde{r}_i}. \tag{3.15}$$

The resulting formula shows a similar shape as Equation (3.6). The posterior distribution can be used to propagate events from reco level to truth level, i.e. the unfolded distribution u_j is obtained to be

$$u_j = \sum_i \tilde{R}_{ji} r_i = \sum_i \frac{A_{ij}}{\tilde{r}_i} r_i. \tag{3.16}$$

If the experimental data (t_j, r_i) and the Monte Carlo simulation ($\tilde{t}_j$, $\tilde{r}_i$) are equal, the unfolding result will be perfect. In an example where experiment and simulation are very different, the resulting unfolded distribution u_j carries a strong bias towards the Monte Carlo simulation. This *model dependency* of the unfolding procedure can be reduced by iterating the calculation several times with the unfolded distribution as a new prior. The complete algorithm for Iterative Bayesian Unfolding is given in Algorithm 1 [82].
The crucial parameter in Iterative Bayesian Unfolding is the number of iterations N. Although biases decrease with higher number of iterations, statistical uncertainties increase. For an infinite number of iterations the unfolding result converges towards the

Algorithm 1 Iterative Bayesian Unfolding

Choose prior for the unfolded distribution to be the Monte Carlo truth distribution:

$$u_j^{(0)} = \tilde{t}_j. \tag{3.17}$$

for training iteration $n \in [1, 2, ..., N]$ **do**

Calculate the current posterior distribution as

$$\tilde{R}_{ji}^{(n)} = \frac{R_{ij} u_j^{(n-1)}}{\sum_k R_{ik} u_j^{(n-1)}}. \tag{3.18}$$

Recalculate the unfolded distribution as

$$u_j^{(n)} = \sum_i \tilde{R}_{ji}^{(n)} r_i. \tag{3.19}$$

end for

result of the simple matrix inversion [3]. As before, the art of unfolding consists in finding a trade-off between an acceptable uncertainty and a minimum bias towards the Monte Carlo simulation.

Iterative Dynamic Stabilising (IDS) [4] is based on the same principles as Iterative Bayesian Unfolding. Nevertheless, this algorithm is far more involved with several new ideas taken into account. This section provides a brief introduction into the mathematical foundation of this algorithm, further details can be found in [4, 83].
First, a regularization function f is defined. The main idea of this function is to provide an information of the significance of an absolute deviation between two bins Δx with respect to its corresponding error σ. In general, the function can be any smooth function with limits $[0, 1]$. The function used in this project is

$$f(\Delta x, \sigma, \lambda) := 1 - \exp^{-\left(\frac{\Delta x}{\lambda \sigma}\right)^2}, \tag{3.20}$$

where the free parameter λ denotes the strength of the regularization: small λ lead to higher values of the function, hence deviations are interpreted as more significant; a large λ has the opposite effect. The function is displayed in Figure 3.3 on the left.
In general, the Monte Carlo simulations are build to be normalized to the same value as the detector-level data which is unfolded. This approach does not consider the fact that the detector-level data might contain additional structures which are not present in the simulation. An example of such a case is shown in Figure 3.3 (right). To gain a better normalization for the Monte Carlo distribution, the additional unknown structures need to be ignored. If this problem was dismissed, the result could generate fake differences between the two spectra in regions where the data is well described by the simulation [4].
The first estimation of the number of data events corresponding to structures simulated in the Monte Carlo $N_{\mathrm{MC}}^{(0)}$ is the number of measured data events

$$N_{\mathrm{D}}^{(0)} = \sum_i r_i. \tag{3.21}$$

A better estimation of $N_{\mathrm{D}}^{(0)}$ can be calculated as

$$N_D^{(1)} = N_D^{(0)} + \sum_i \left[1 - f(|\Delta r_i|, \hat{\sigma}(r_i), \lambda_N)\right] \cdot \Delta r_i \,, \tag{3.22}$$

with

$$\Delta r_i = r_i - \frac{N_D^{(0)}}{N_{\mathrm{MC}}} \tilde{r}_i \,, \tag{3.23}$$

$$\hat{\sigma}(r_i) = \sqrt{\sigma^2(r_i) + \left(\frac{N_D^{(0)}}{N_{MC}}\right)^2 \sigma^2(\tilde{r}_i)} \,, \tag{3.24}$$

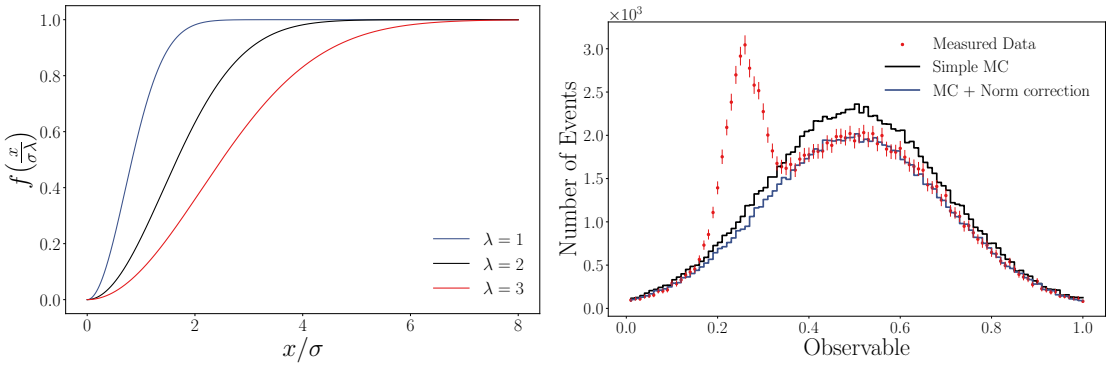


Figure 3.3: The regularization function (left) assigns individual weights depending on the ratio of differences and uncertainties [4]. The function is plotted for different parameters λ. The toy example (right) shows that a regularization in the calculation of the Monte Carlo norm correction N_D/N_{MC} is necessary if there is additional structure which is present only in the data. A usual fitting function (black) is not able to model the data (red). The normalisation correction accounts for the additional structure [84].

using the total number of Monte Carlo Events N_{MC} and the uncertainties of the detector-level data $\sigma(r_i)$ and Monte Carlo $\sigma(\tilde{r}_i)$. The new introduced parameter λ_N determines the strength of the norm regularization.
Equation (3.22) can be used for $N_{\mathrm{iter,norm}}$ iterations to get updated values for N_D, obtaining a better normalization factor N_D/N_{MC}. Equation (3.22) is not only used to derive correction factors between r_i and $\tilde{r}_i$, but also for a comparison between an unfolded distribution u_j and the Monte Carlo truth component $\tilde{t}_j$. After calculating the normalization factor it is possible to start the unfolding itself. In the case of identical initial and final binning the unfolded distribution is given as [4]

$$u_j = \frac{N_D}{N_{\mathrm{MC}}} \cdot \tilde{t}_j + \sum_i \Big[f(|\Delta r_i|, \hat{\sigma}(r_i), \lambda_U) \cdot \Delta r_i \cdot \tilde{R}_{ji} + (1 - f(|\Delta r_i|, \hat{\sigma}(r_i), \lambda_U)) \Delta r_i \delta_{ij} \Big]. \tag{3.25}$$

The first term of this equation is the Monte Carlo truth component with a corrected norm. The second part (in square brackets) adds the structures which are only visible in the data and not covered by the simulation. Only a fraction f of these events is unfolded using the current unfolding probability matrix $\tilde{R}_{ji}$, the rest of the events are kept in their respective bin. Reducing the unfolding parameter λ_U reduces the number of events which are unfolded.
In analogy to Equation (3.15) an updated migration matrix A'_{ij} is used. This migration matrix is calculated as

$$A'_{ij} = A_{ij} + f(|\Delta t_j|, \hat{\sigma}(t_j), \lambda_M) \Delta t_j \frac{N_{\mathrm{MC}}}{N_D} R_{ij}, \tag{3.26}$$

with

$$\Delta t_j = t_{j,\text{unf}} - \frac{N_D}{N_{\text{MC}}} \tilde{t}_j \,, \tag{3.27}$$

$$\hat{\sigma}(t_j) = \sqrt{\sigma^2(t_{j,\text{unf}}) + \left(\frac{N_D}{N_{MC}}\right)^2 \sigma^2(\tilde{t}_j)} \,. \tag{3.28}$$

A relative weighting factor f is introduced again to regularize the size of the updates. The strength of the regularization is again determined with a parameter λ_M. If the function f is set to one, the updating algorithm of IBU is reobtained. Regardless of the update of the migration matrix, the original detector response is not changed throughout this process; the detector response matrix R_{ij} stays the same.
The full Iterative Dynamic Stabilising method is summarised in Algorithm 2 [83]. Additionally, there is an extended algorithm which discusses how to treat background subtractions, more information about this can be found in [4]. The parameters λ_N, $N_{\text{iter,norm}}$, λ_U, λ_M and N_{iter} usually need to be determined by using different pseudo-experiments to judge the introduced biases and sensitivity to injected signals [85].

Algorithm 2 Iterative Dynamic Stabilising

Choose prior for the unfolded distribution to be the Monte Carlo truth distribution:

$$u_j^{(0)} = \tilde{t}_j. \tag{3.29}$$

for training iteration $n \in [1, 2, ..., N]$ **do**
Calculate the updated migration matrix

$$A_{ij}^{(n)} = A_{ij}^{(n-1)} + f(|\Delta t_j|, \hat{\sigma}(t_j), \lambda_M) \Delta t_j \frac{N_{\text{MC}}}{N_D} R_{ij}, \tag{3.30}$$

to obtain the unfolding probability matrix

$$\tilde{R}_{ji}^{(n)} = \frac{A_{ij}^{(n)}}{\sum_k A_{ik}^{(n)}}. \tag{3.31}$$

Recalculate the unfolded distribution

$$u_j = \frac{N_D}{N_{\text{MC}}} \cdot \tilde{t}_j + \sum_i \Big[f(|\Delta r_i|, \hat{\sigma}(r_i), \lambda_U) \cdot \Delta r_i \cdot \tilde{R}_{ji} + (1 - f(|\Delta r_i|, \hat{\sigma}(r_i), \lambda_U)) \Delta r_i \delta_{ij} \Big] \,. \tag{3.32}$$

end for
The needed Monte Carlo normalization factors are iteratively determined according to Equation (3.22).

3.2. Inefficiencies, Fakes and Misses

Up to this point, each event on truth-level corresponds to one event on detector-level. In reality, it is possible that an event is missed, misidentified or not accepted because of certain selection criteria. This leads to the possibility of a particle only appearing on one level. An event which appears on truth-level but does not appear on detector-level is labeled as a *miss*; a detector-level event that does not appear on truth-level is labeled as a *fake*. This influence of fakes and misses needs to be taken into account [85].
In addition to the information on how detectors measure events and the strength of their smearing, it is required to have prior information about the probability of an event appearing on truth- and detector-level. Therefore, the distribution of the fakes and misses already needs to appear in the Monte Carlo simulation of the detector measurement. This is illustrated in Figure 3.4. The easiest approach to consider misses and fakes in the unfolding is to apply efficiencies to the observed data r_i and the unfolding result u_j. With the distribution of fakes f_i and the distribution of misses m_j the needed efficiencies are defined as

$$\epsilon_i^{(fake)} = \frac{f_i}{f_i + \tilde{r}_i} \qquad \text{and} \qquad \epsilon_j^{(\text{miss})} = \frac{m_j}{m_j + \tilde{t}_j}. \tag{3.33}$$

using the truth-level Monte Carlo $\tilde{t}_j$ and the detector-level Monte Carlo $\tilde{r}_i$. The measured distribution is corrected by multiplying bin-wise with the fake efficiency $\epsilon_i^{(fake)}$, this is known as a *purity correction*. The unfolding result is corrected by dividing with the miss efficiency $\epsilon_j^{(\text{miss})}$, this is commonly referred to as *stability correction* [86].

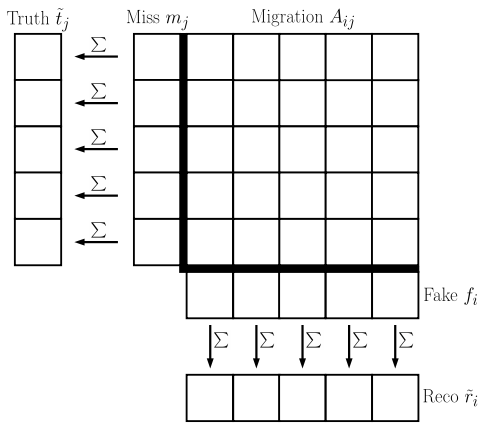


Figure 3.4: Visualization of the treatment of fakes and misses in matrix-based unfolding algorithms. The distribution of misses/fakes is appended in an extra column/row to the migration matrix. The summation over columns/rows returns the detector-level/truth-level distribution of the Monte Carlo simulation. Source: [85].

3.3. Uncertainties

After the choice for a specific unfolding algorithm, an unfolded distribution u_j is obtained. The evaluation of uncertainties is difficult since the resulting distribution for u_j cannot be assumed to be Poissonian anymore. For example, non-zero off-dimensional elements of $\tilde{R}_{ji}$ introduce some degree of correlations between unfolded bins. Therefore, it is wrong to calculate the square root of the number of events in one bin to evaluate its uncertainty [3]. The calculation of covariance matrices is possible for some of the unfolding algorithms, nevertheless this thesis sticks to a more general form of uncertainty estimation: *Bootstrapping* [87].
The idea of bootstrapping is to unfold a measured distribution several times while making small changes in the measured distribution r_i as well as the Monte Carlo simulated migration matrix A_{ij}. The value of each measured bin r_i is replaced by a number drawn from a Poisson distribution $\mathcal{P}(r_i)$. In analogy, the entries of the migration matrix are fluctuated (including the fakes and misses) by drawing a new entry from $\mathcal{P}(A_{ij})$. These fluctuations will consequently impact the unfolded distribution u_j. These "toy fluctuations" are performed several times to calculate the covariance matrix

$$\mathrm{cov}_{ij} = \frac{1}{N_{\mathrm{toys}}} \sum_{n=1}^{N_{\mathrm{toys}}} (u_i^{(n)} - \overline{u}_i)(u_j^{(n)} - \overline{u}_j), \tag{3.34}$$

with the number of toys N_{toys} and the mean value

$$\overline{u}_i = \frac{1}{N_{\mathrm{toys}}} \sum_{n=1}^{N_{\mathrm{toys}}} u_i^{(n)} \,. \tag{3.35}$$

The standard deviation in each bin and the correlation matrix can be calculated as

$$\sigma_j = \sqrt{\mathrm{cov}_{jj}}, \qquad \mathrm{cor}_{ij} = \frac{\mathrm{cov_{ij}}}{\sigma_i \sigma_j}. \tag{3.36}$$

3.4. Analytic Toy Example

To demonstrate the unfolding algorithms it is beneficial to construct an example which is analytically solvable. Such an example can be constructed if every relevant distribution is a Gaussian. The detector response function is set to be a Gaussian shift and smearing with the parameters $\mu_s = -6$, $\sigma_s = 3$. The analytic detector response function to connect distributions on truth-level t with their corresponding distributions on detector-level r reads

$$p(r|t) = \frac{1}{\sqrt{2\pi\sigma_s^2}} \exp\left(-\frac{(r - (t + \mu_s))^2}{2\sigma_s^2}\right). \tag{3.37}$$

With the generic Gaussian function $G(x, \mu, \sigma)$ the truth-level distributions are defined as

- Monte Carlo simulation at truth-level: $\tilde{f}(t) = G(t,\ \mu_t = 4,\ \sigma_t = 4)$,

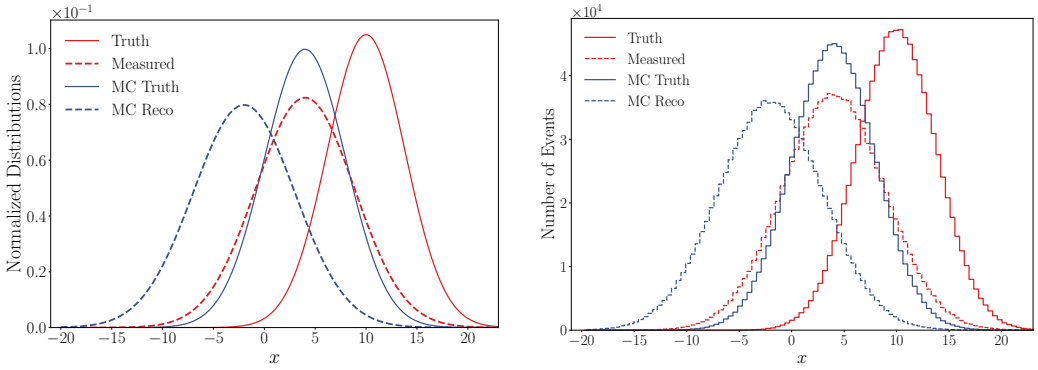


Figure 3.5: The toy example used in this section. All distributions are Gaussian, for illustration purposes there are huge differences between the simulation and the pseudo-experiment. The continuous functions are shown on the left, their binned version with 10^6 sampled events on the right. There graphs show the truth-level data (red, solid), the detector-level data (red, dashed), the truth-level Monte Carlo (blue, solid) and the detector-level Monte Carlo (blue, dashed).

- pseudo-experimental data at truth-level: $f(t) = G(t,\ \mu_{\text{Truth}} = 10,\ \sigma_{\text{Truth}} = 3.8)$.

The corresponding detector-level distributions can be calculated by evaluating the convolution integral of the truth distribution with the detector response function

$$g(r) = \int \mathrm{d}t\, p(r|t) f(t). \tag{3.38}$$

Since a Gaussian smearing is applied to a Gaussian function, the result is a new Gaussian function with summed means and variances:

- Monte Carlo simulation at detector-level: $\tilde{g}(r) = G(r,\ \mu_r = \mu_t + \mu_s,\ \sigma_r^2 = \sigma_t^2 + \sigma_s^2)$,
- pseudo-experimental data at detector-level: $g(r) = G(r,\ \mu_M = \mu_{\text{Truth}} + \mu_s,\ \sigma_M^2 = \sigma_{\text{Truth}}^2 + \sigma_s^2)$.

The "data" used in this toy example is of course not measured by an experiment. A more precise expression would be "generated pseudo-data", for simplicity the term "data" is used in the following.
The distributions mentioned above are shown in Figure 3.5, both as continuous functions (left) as well as in a binned histogram (right). It can clearly be seen that this toy example shows a mismodeling situation where the Monte Carlo simulation (blue) is in no way close to the experimental data (red). In addition to this mismodeling there is a huge detector response with both a smearing and a shift, hence this toy example produces a worst-case scenario. It is possible to calculate an analytic expectation for the iterative algorithms. By using Bayes theorem and assuming the truth-level Monte Carlo as a prior for the

unfolded distribution $\tilde{f}(t) = \tilde{f}_{\text{prior},0}(t)$, the pseudo-inverted function $p(t|r)$ is calculated as

$$\begin{aligned} p(t|r) &= \frac{p(r|t)\ \tilde{f}_{\text{prior},0}(t)}{\tilde{g}(r)} \\ &= \frac{1}{\sqrt{2\pi}} \sqrt{\frac{\sigma_r^2}{\sigma_t^2 \sigma_s^2}} \exp\left(-\frac{(r-(t+\mu_s))^2}{2\sigma_s^2} - \frac{(t-\mu_t)^2}{2\sigma_t^2} + \frac{(y-\mu_r)^2}{2\sigma_r^2} \right). \end{aligned} \tag{3.39}$$

The unfolded distribution after one iteration $f_{u,1}(t)$ can be calculated by convoluting this pseudo-inverse with the measured distribution $g(r)$:

$$\begin{aligned} f_{u,1}(t) = \int p(t|r) g(r) \mathrm{d}r = \frac{1}{2\pi} \sqrt{\frac{\sigma_r^2}{\sigma_t^2 \sigma_s^2 \sigma_M^2}} \cdot \\ \int \mathrm{d}r \exp\left(-\frac{(r-(t+\mu_s))^2}{2\sigma_s^2} - \frac{(t-\mu_t)^2}{2\sigma_t^2} + \frac{(r-\mu_r)^2}{2\sigma_r^2} - \frac{(r-\mu_M)^2}{2\sigma_M^2} \right). \end{aligned} \tag{3.40}$$

The result is an updated prior function, a Gaussian distribution with parameters

$$\mu_{t,1} = \frac{\mu_M \sigma_t^2 + \mu_t \sigma_s^2 - \mu_s \sigma_t^2}{\sigma_s^2 + \sigma_t^2}, \qquad \sigma_{t,1} = \frac{\sqrt{\sigma_t^2 \sigma_M^2 + \sigma_t^2 \sigma_s^2 + \sigma_s^4}}{\sigma_s^2 + \sigma_t^2} \sigma_t. \tag{3.41}$$

This is everything needed for analytic predictions of the unfolded distribution in each iteration of an iterative unfolding algorithm. To obtain the parameters of the distribution after the second iteration, a recalculation of Equation (3.41) with the values $\{\mu_{t,1}, \sigma_{t,1}\}$ instead of $\{\mu_{t,0}, \sigma_{t,0}\}$ is needed. To make sure that this is a consistent result, it can be shown that in the limit of infinite iterations the mean value and the standard deviation approach the exact result $\{\mu_{\text{Truth}}, \sigma_{\text{Truth}}\}$. This is derived in Appendix A.3.

3.5. Unfolding the Toy Example

At this point the classical matrix-based unfolding algorithms can be applied to the toy example described above. Since there are strong differences between the Monte Carlo and the data, it is expected that a simple inversion of the response matrix will not succeed to obtain a proper unfolding result. Hence, more sophisticated matrix-based algorithms will be needed. Especially for the iterative algorithms it is interesting to compare the unfolded distribution after each iteration with the analytic prediction. In general, this is more a sanity check whether the algorithms are implemented properly, since the Iterative Bayesian Unfolding executes the exact same steps used for the analytic derivation with a binned phase space.
The results for the simple matrix unfolding are shown in the upper left of Figure 3.6. It behaves as expected: due to a lack of regularization it is unable to obtain an unfolding result which obeys to a physical smoothness. Instead there are large bin-to-bin fluctuations, which imply strong correlations between the bins. This demonstrates the necessity of regularized unfolding algorithms.

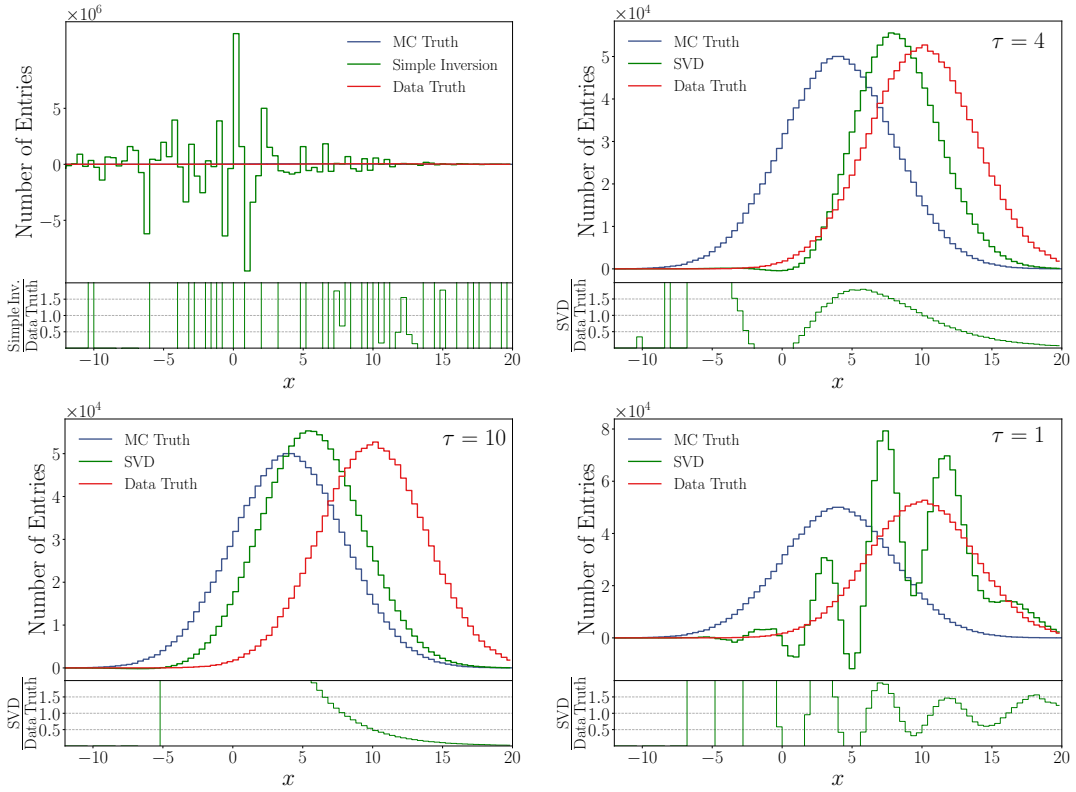


Figure 3.6: Unfolding the toy model using different matrix-based algorithms. The figures show the truth-level Monte Carlo (blue), the truth-level data (red) as well as the unfolding result (green). In the upper left graph the simple matrix inversion is applied, the result is an expected high-frequency spectrum. This demonstrates that the problem requires some kind of regularization. In the upper right and the two lower plots the SVD algorithm is applied using several different strengths of the regularization parameter τ. In the lower left the regularization is too strong, i.e. the unfolded distribution is biased towards the truth-level Monte Carlo simulation. In the lower right the regularization is too weak leading to a fluctuating result. In the upper right a compromise between bias and fluctuations is shown. This compromise still significantly deviates from the true distribution.

The SVD unfolding with a Tikhonov regularization depends on the regularizing parameter τ. As explained in Section 3.1, if τ is too low the regularization is too weak to suppress the oscillations. If τ is too high the unfolded distribution will have a bias towards the truth-level Monte Carlo simulation. This behavior is demonstrated in the other three plots of Figure 3.6. It can be seen that the SVD algorithm does not succeed in unfolding the distributions properly. This is due to the fact that the toy example was designed specifically to see an iterative improvement, the plain non-iterative SVD cannot correct

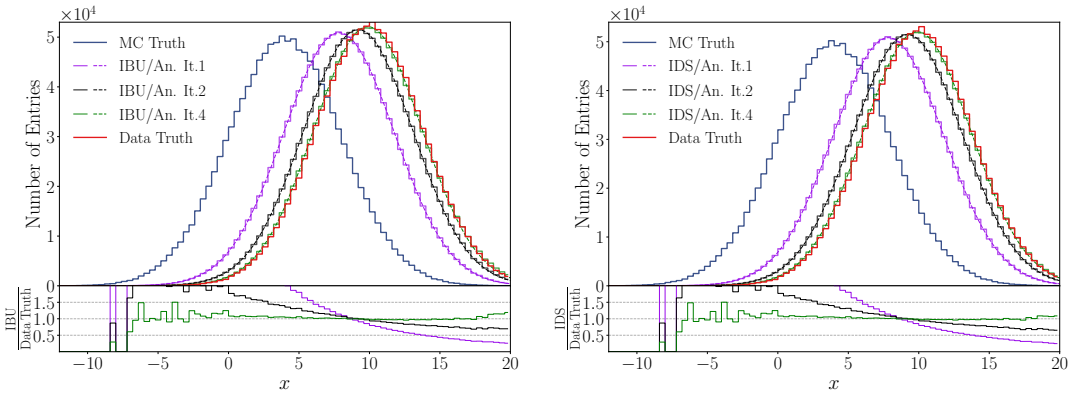


Figure 3.7: Unfolding the toy model with iterative matrix-based algorithms. On the left the result for IBU, on the right the result for IDS with the unfolded distribution are shown after each iteration. The unfolded distributions (purple, black and green solid line) fit the analytic expectations (respective dashed lines) in the region of high statistics. The ratios with the experimental truth-level data (red) show that the bias towards the truth-level Monte Carlo (blue) is iteratively reduced.

for the strong shape difference.
The iterative IBU and IDS algorithms fulfill the expectations. After each iteration it is possible to compare the analytic prediction with the unfolding result. The results up to iteration $N_{\text{iter}} = 4$ are shown in Figure 3.7. The parameters for Iterative Dynamic Stabilising are set to $\lambda_N = 0$, $N_{\text{iter,norm}} = 5$ and $\lambda_U = \lambda_M = 0.5$.
The choice skips the correction of the normalization. This can be done because there are no local structures in this toy model which might have an impact on the normalization. The values for λ_U and λ_M are standard values for these parameters. Because of this rather conservative choice it is not surprising that the IDS algorithm approaches nearly the same result as the IBU algorithm.

The main result of this section is that the iterative algorithms succeed in unfolding this toy example, whereas the non-iterative do not. The example can therefore be used in Section 7 to demonstrate the limits of a non-iterative cINN unfolding, and to show that the result can be improved by adapting an iterative approach.

4. Neural Networks

Artificial neural networks are an approach of deep learning inspired by the neurons of the human brain. The biological neuron is constructed to release a chemical transmitter once a certain action potential in form of an electrical impulse is reached [88]. This release induces another electrical impulse which is passed to other neurons and so on. The idea of interacting neurons is the basis on which an artificial neural network is constructed.

4.1. Basic Structure

The simplest possible neural network is a single neuron (also called perceptron) where an output y is calculated using several inputs x_i. The neuron multiplies weights w_i with the inputs, adds a bias b and finally applies a non-linear activation function f. Mathematically this is expressed as

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right). \tag{4.1}$$

This is visualised in Figure 4.1. The idea of machine learning is to optimize these weights to produce a desired outcome, for example to learn a certain pattern or mapping. To approach more difficult problems, the neurons can be organised in layers, where they are evaluated in parallel. This is visualised in Figure 4.2. In a *fully-connected* neural network the inputs of a neuron in each layer is given by the previous layer neurons' combined output. There are three kinds of layers: the input layer at the beginning, the output layer at the end and the hidden layers in between [89].

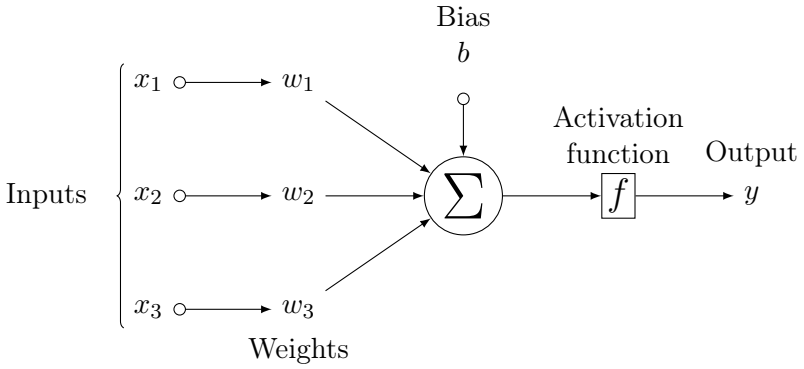


Figure 4.1: Structure of an artificial neuron. The input is multiplied with weights w_i, summed up with an additional bias b and a non-linear activation function f is applied.

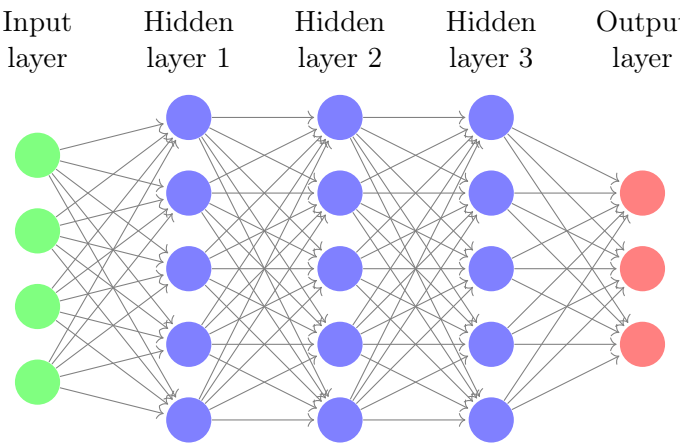


Figure 4.2: Exemplary structure of a generic fully-connected neural network. Each circle illustrates one neuron. The green layer on the left side is called the input layer, the blue layers in the center are called the hidden layers and the red layer on the right is called the output layer.

The input layer takes the input information and passes it to the first hidden layer. At this stage it is possible to implement a preprocessing of the data which simplifies the task of the neural network.
Each neuron of the first hidden layer calculates outputs according to Equation (4.1). The weights are not assigned to a single neuron, but their magnitudes express the importance of a connection between two neurons of consecutive layers (arrows in Figure 4.2). After the first hidden layer, the calculated output is passed on to the next hidden layer and so on. In simple neural networks, the hidden layers usually have more neurons per layer than the input or output layer to guarantee that no input information is compressed in the neural network itself. The number of neurons in one hidden layer will be called units. The output layer requires the exact number of neurons to fit the dimension of the output, the dimension of input and output do not have to agree. In addition, special activation functions can be applied to the output layers in comparison to other layers to restrict the output to a certain range of values.
At this point it is crucial to understand why a non-linear activation function is needed if a non-linear distribution should be learned. If there was be no activation function at all or if a linear activation function was chosen, Equation (4.1) would be a composition of linear functions and thus linear itself. The consequence of this linearity would be that all the hidden layers can be replaced by one single hidden layer. Therefore, in order to construct a *deep* neural network which is able to learn non-linear distributions, non-linear activation functions are needed. The *Universal Approximation theorem* [90] states that an arbitrarily good representation of a distribution by a neural network is only possible due to this non-linearity [88]. There are several types of activation functions. An overview can be found in [91]. In the following, the activation functions important

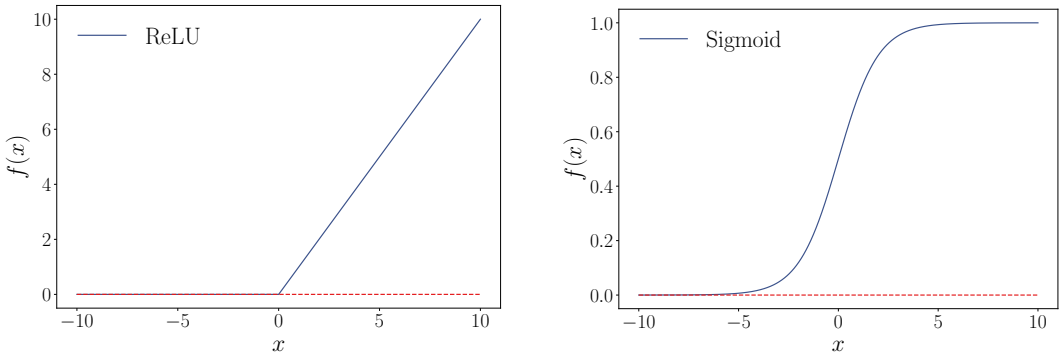


Figure 4.3: Two commonly used activation functions. The figure on the left shows the nearly linear ReLU activation function. The figure on the right shows a sigmoid activation function.

for this thesis are discussed. They are shown in Figure 4.3.
The *ReLU* activation function (**Re**ctified **L**inear **U**nit) has been the most used activation function since its proposal in 2010 [91]. It is defined by

$$\mathrm{ReLU}(x) = \max\{0, x\} = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}. \tag{4.2}$$

This nearly linear activation function is especially useful due to its simplicity. Because of this, the computation is much faster compared to other activation functions. Problems with this activation function can occur with "dying" neurons, i.e. the output of such a neuron is constantly zero due to its weight configuration. To solve this problem, the LeakyReLU activation function can be used, which introduces a small slope for $x < 0$.
The *Sigmoid* activation function is defined as

$$\mathrm{Sigmoid}(x) = \frac{1}{1 + e^{-x}}. \tag{4.3}$$

It is often used in the output layer since it restricts the output to the interval $[0, 1]$. This is a possibility to ensure that the network output is a probability [92].

4.2. Training

Training a neural network is a synonym for an advantageous optimization of the network parameters (i.e. the weights and biases) that leads to the network learning the structure of an input dataset. An example is the case where a mapping $y = g(x)$ should be learned. Doing this requires training data $(x, \hat{y})$ defining the relation g.
Prior to the optimization, all weights are initialised randomly, usually according to a Gaussian distribution with mean $\mu = 0$ and standard deviation $\sigma = 1$. The goal of the training process is to adjust the weights of a neural network by applying it to the input training samples x and comparing the resulting output y with the desired result $\hat{y}$. To

do this, it is necessary to introduce a differential learning objective, the *loss function*, which provides a feedback about the network performance.

The loss function is designed to increase for large differences between y and $\hat{y}$ and to decrease if they become very similar. One way to implement such a function is the mean squared error [88]

$$L_{\text{MSE}} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2, \tag{4.4}$$

with n being the number of samples to evaluate. The value of the loss function can only be modified by changing the network parameters w. The goal of the network training is hence to find the perfect parameter configuration w_∞ which minimises the loss L to its global minimum [88]:

$$w_\infty = \arg\min_w L(w). \tag{4.5}$$

Gradient descent is used find this minimum numerically. After initialising the network with random parameters w_0, the network is applied to the training data to obtain a prediction and calculate the loss $L(w_0)$. After this the network parameters are updated as

$$w_1 = w_0 - \alpha \nabla_{w_0} L(w_0), \tag{4.6}$$

with the learning rate α. The gradients of the loss function are calculated with *backpropagation*, which is a repeated application of the chain rule to the loss function. More information about backpropagation and its implementation can be found in [89].
This procedure of updating the network parameters is iteratively repeated to improve the parameters towards their desired values w_∞. The learning rate α quantifies the size of the step. It is important to find a good value for α: if it is to low, the neural network might get stuck in a local minimum. If it is to high, the network will never converge to the global minimum. This behavior is shown in Figure 4.4.
The calculation of the loss function gradient faces another challenge: especially for large datasets, it becomes impossible to compute the gradient. Instead, it is evaluated over a small, randomly chosen subset of the training data. This method is called stochastic gradient descent; the subsets of the training data are called minibatches or batches. The stochastic noise introduced by these batches further stabilises the convergence of the training process [93].
Since the loss function can have local minima, it needs to be ensured that the network finds the global minimum. This task is called exploration. After finding this minimum, the neural network needs to approach it in well chosen steps. This task is called exploitation. Gradient descent is a good starting point to approach these challenges, but there are several more techniques to optimize the training, for example learning rate schedulers or optimizers.

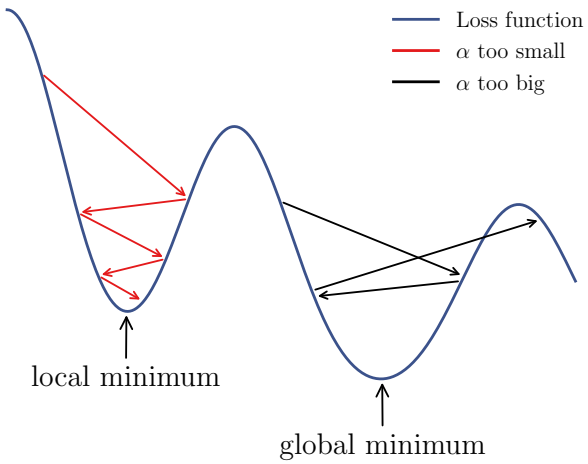


Figure 4.4: Learning process of a neural network with different learning rates. The learning rate is an externally adjustable *hyperparameter* of the network.

A learning rate scheduler can be used to change the learning rate continuously. The networks presented in this thesis use a *one-cycle learning rate* [94], i.e. they start the training with a learning rate value α_0, increase it over several epochs to α_{max} and finally decrease it to zero in the final epochs. The idea is to first locate the global minimum with the enhanced learning rate and use the smaller learning rate afterwards to converge. In addition to the changes in the learning rate an optimizer, which replaces the standard gradient descent, can be used.

ADAM [95] is an optimizer which replaces the standard gradient descent. It adds an additional term in the update of the parameters, called *momentum* [96]:

$$m_{t+1} = \beta_1 m_t + (1 - \beta_1)\nabla_{w_t} L, \tag{4.7}$$

$$w_{t+1} = w_t - \alpha m_{t+1}. \tag{4.8}$$

with $\beta_1 \in [0, 1]$ to add a continuous average of the previous update steps to the next update. This additional term smooths out the updates, helps to prevent getting stuck in local minima and is useful in areas in which there are big differences between the gradients in several directions. In such a case, neural networks without momentum tend to oscillate instead of going straight to the minimum.
The second concept implemented in ADAM is *RMSProp* ("root mean square radius propagation") [97]. The idea is to implement an individual correction factor to the learning rate applied to each batch, since the gradients may vary from layer to layer. For layers with a continuously small gradient no progress is made, while layers with continuously large updates might overstep the minimum. To prevent this one can introduce a correc-

tion factor v_t which is a moving average of the squared gradient updates:

$$v_{t+1} = \beta_2 v_t + (1 - \beta_2)(\nabla_{w_t} L)^2, \tag{4.9}$$

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{v_{t+1}} + \epsilon} \nabla_{w_t} L, \tag{4.10}$$

with a factor $\beta_2 \in [0, 1]$ as well as a regularization parameter ϵ to avoid divergences.
ADAM combines these two approaches. In addition, it introduces correction factors for the momentum and the RMSProp, since they are biased towards the gradients of the first batches. The full ADAM algorithm can be written as

$$m_{t+1} = \beta_1 m_t + (1 - \beta_1) \nabla_{w_t} L, \tag{4.11}$$

$$v_{t+1} = \beta_2 v_t + (1 - \beta_2)(\nabla_{w_t} L)^2, \tag{4.12}$$

$$\hat{m}_{t+1} = \frac{m_{t+1}}{1 - \beta_1^t}, \tag{4.13}$$

$$\hat{v}_{t+1} = \frac{v_{t+1}}{1 - \beta_2^t}, \tag{4.14}$$

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{\hat{v}_{t+1}} + \epsilon} \hat{m}_{t+1}, \tag{4.15}$$

where β_i^t is β_i to the power of t. The values used for ADAM in this project are $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$.

Overfitting is a problem that can occur while training neural networks. Since the training data is limited, several training iterations with the same data cannot be avoided. This risks overspecialisation or overfitting [98], meaning that the neural network learns details about the training data which are specific only to the training data. An example for this is given in Figure 4.5. The network no longer learns an underlying structure, but the exact distribution of the training data. Bigger networks with more parameters are more likely to overfit. Nevertheless, some problems require big networks and therefore need to deal with this problem explicitly.
One possibility to prevent overfitting is to implement a *dropout* of neurons [99]. The idea is to omit each hidden neuron with a dropout probability, for example 0.25. In this configuration the output of approximately one in four neurons is set to zero in an epoch of training. This procedure can be seen as "a very efficient way of performing model averaging with neural networks" [99], as there are different combinations of active neurons in every training epoch, but all these "subnetworks" share the same weights. These new combinations prevent the network from focusing too much on the training data itself and shift its focus to the underlying nature of the data. An appropriate choice of the dropout probability is crucial, since a balance between preventing overfitting and still enabling an effective training procedure has to be ensured.
Another possibility to prevent overfitting is an *early stopping* of the learning process [100]. During the training procedure, the loss of the training sample is continuously evaluated. In addition, a loss of a so-called test sample is calculated, which is not used

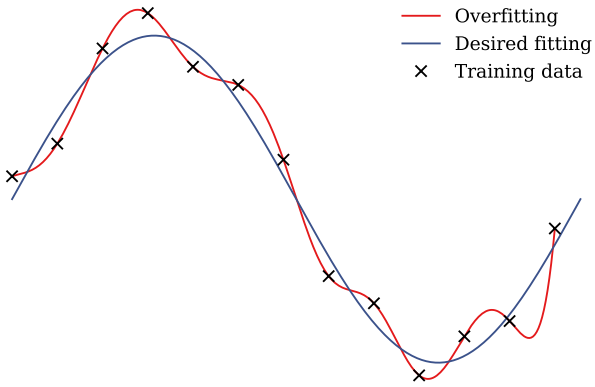


Figure 4.5: Overfitting of a neural network shown in a graphic example. The underlying structure is a sine function.

during training but follows the same underlying distribution. As soon as the loss of the test sample starts to increase again, the network starts overfitting and the training is stopped.

4.3. Discriminative Neural Networks

Neural networks can be used to approach *classification* problems. A well-known example is the MNIST-problem: the data contains images of handwritten digits and their corresponding labels. The task to recognize the number with a machine learning algorithm was long used for benchmarking neural network performances. Modern neural network architectures need more involved problems to distinguish performances [89]. Networks that are designed to label data are in general called *classifiers* or *discriminators*.
In the context of this thesis it is suitable to consider a binary version of the MNIST problem, in which there are only pictures of zeros and ones. The easiest way to approach the classification problem is to use a fully connected neural network. The number of input neurons matches the number of pixels in each picture, the input values are the brightnesses of one pixel. The output layer contains only one single neuron with a sigmoid activation function, to restrict the output values y to $[0, 1]$. Each picture of a zero and a one in the training data is paired with its correct label $\hat{y} \in \{0, 1\}$. These (picture, label) samples are given to the neural network to perform a training as described in Section 4.2. The used loss function is a binary cross entropy loss

$$L_{BCE} = -\sum_{i=1}^{n} \hat{y}_i \log(y_i) + (1 - \hat{y}_i) \log(1 - y_i). \tag{4.16}$$

Since $\hat{y} \in \{0, 1\}$ for each training point, only one part of the sum, which sanctions the deviations from the true label logarithmically, is relevant.
Since there is only limited training data available and the number of network parameters

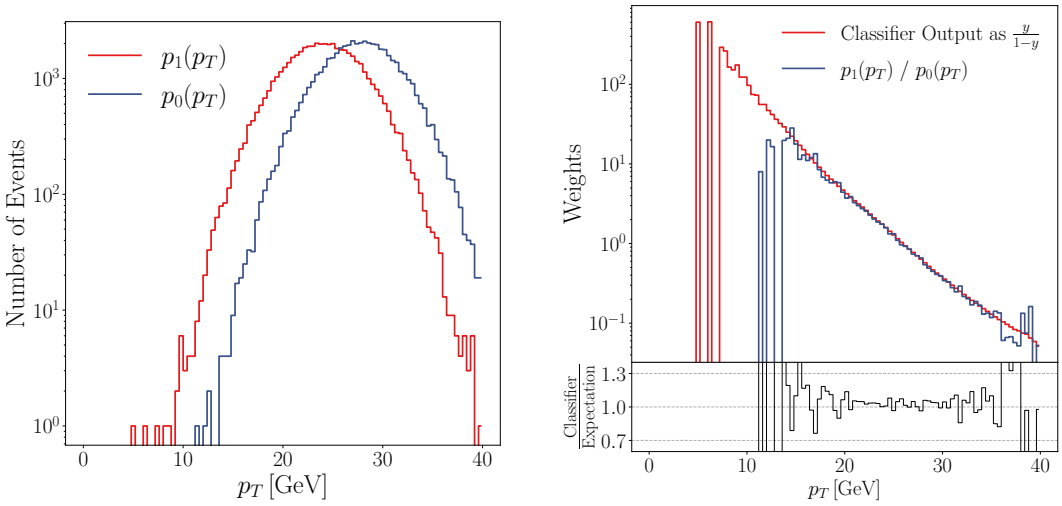


Figure 4.6: Example for a reweighting problem. The left image shows two distributions, where p_0 (blue) should be reweighted on an event-by-event basis to match p_1 (red). The right image shows the result of a trained classifier, where the binned ratio p_1/p_0 (blue) is approximately equal to the ratio of the mean probabilities assigned by the classifier to the events of each bin, $y/(1-y)$ (red).

is limited, a perfect performance in which the classifier output is always exactly one or zero will never be reached. Instead, a picture of a one is for example evaluated to $y = 0.95$. This shows that the network concludes the picture is much more likely a one than a zero. It can be shown that the network which converges to the minimum of the binary cross entropy loss actually learns the probability ratio of a picture being a one or a zero [93]. A picture which is equally close to a one as to a zero will thus return an output which is close to 0.5.

This learning of probabilities can be used to *reweight* distributions towards each other [101]. An example for a reweighting problem is given in Figure 4.6, left. The task is to assign each event of the distribution p_0 a weight such that the resulting reweighted distribution is similar to distribution p_1. It is possible to divide the two histograms bin by bin, assigning each event the bin ratio p_1/p_0 of its respective bin as a weight.

Instead of introducing a binning, the machine learning reweighting algorithm takes an event-by-event approach, i.e. each point of the distribution p_0 is assigned an individual weight. This can be achieved by a neural network which is trained to distinguish between samples of the distributions. Hence, the training labels $\hat{y} = 1$ are assigned to the points of the distribution p_1 and $\hat{y} = 0$ to points of p_0. As can be seen on the left of Figure 4.6. the distributions are overlapping. In the region where p_1 and p_0 are of similar size, the classifier output is again expected to be around $y \approx 0.5$ since a clear assignment to either p_1 or p_0 is not possible in this region. By design the neural network therefore learns the underlying probability distributions. The likelihood ratio of the distributions is therefore

well-approximated as

$$L[p_T, p'_T] = \frac{p_1(p_T)}{p_0(p'_T)} \approx \frac{y}{1-y}, \tag{4.17}$$

with the classifier output y. Both ratios are calculated in a binned way. This relation will not hold to arbitrary precision due to limited training statistics. Nevertheless, it gives a good estimation if the classifier has learned to distinguish the distributions in principal. A plot of these ratios for a trained network is given in Figure 4.6 on the right. This concept of reweighting is used in the iterative unfolding algorithms introduced in Section 5 and 7.

4.4. Generative Neural Networks

In general, discriminative models are designed to draw boundaries in data spaces, i.e. they try to predict the correct labels for data. In contrast to this, generative models model the distribution of the data itself. A trained generative model is able to sample completely new data which behaves like the training data.
In this thesis a *normalizing flow* is used which learns a mapping between a simplistic *latent space* and a complicated physical phase space density. In unfolding problems, it is suitable to use *conditional Invertible Neural Networks*, a specialised version of *Invertible Neural Networks*.

Normalizing Flows [102] are used to model complex densities precisely and sample from them in a controlled manner. The basic idea is to connect events of a complicated target distribution $y \sim p_Y(y)$ to a random variable $z \sim p_Z(z)$

$$g_\theta : z \to y, \tag{4.18}$$

using a bijective (and therefore invertible) mapping g_θ which depends on network parameters θ. The concatenation of invertible mappings is invertible again. Therefore, multiple bijective mappings can be combined to form a normalizing flow. The forward evaluation of Equation (4.18) is the *generative* direction of the normalizing flow. The inverted direction

$$\overline{g}_\theta \equiv g_\theta^{-1} : y \to z, \tag{4.19}$$

normalizes the complicated target distribution. The latent density $p_Z(z)$ is connected to the target density $p_Y(y)$ as

$$p_Y(y) = p_Z(z)\left|\frac{\partial g_\theta(z)}{\partial z}\right|^{-1} = p_Z(\overline{g}_\theta(y))\left|\frac{\partial \overline{g}_\theta(y)}{\partial y}\right|, \tag{4.20}$$

with the Jacobian $\partial g_\theta(z)/\partial z$.
Given a sample z from the latent distribution, a trained normalizing flow with mapping g_θ can be used to generate a sample of the target distribution. Alternatively, the density of

a sample y can be computed using the inverse direction. This bijective mapping provides a powerful generative pipeline to model complicated distributions. In order for this to be an effective approach, the latent space distribution needs to be known and simple enough to allow for an efficient sample generation. One possibility is to choose $p_Z(z)$ to be a multivariate Gaussian with mean zero and an identity matrix as the covariance. In addition, g_θ is required to be flexible enough to learn a non-trivial transformation. Furthermore its Jacobian determinant needs to be efficiently computable.

Invertible Neural Networks (INN) [103] are a special type of a normalizing flow networks, which fulfill these requirements by providing a bijective mapping and an easily calculatable Jacobian in both directions. The INN uses affine coupling layers as building blocks [104, 105]. The input vector z is split in half, $z = (z_1, z_2)$, which allows to compute the output $y = (y_1, y_2)$ of the layer as

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} z_1 \odot e^{s_2(z_2)} + t_2(z_2) \\ z_2 \odot e^{s_1(y_1)} + t_1(y_1) \end{pmatrix} \Leftrightarrow \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} (y_1 - t_2(z_2)) \odot e^{-s_2(z_2)} \\ (y_2 - t_1(z_1)) \odot e^{-s_1(y_1)} \end{pmatrix}, \tag{4.21}$$

with arbitrary functions $s_{1,2}$ and $t_{1,2}$ and the element-wise product $\odot$. This bijective mapping works independent of the functional form of $s_{1,2}$ and $t_{1,2}$, i.e. these functions can be chosen to be a fully connected neural network with several layers. A deep INN is composed of a sequence of these building blocks. To increase the model capacity it is advantageous to permute the dimensions between coupling blocks. This enhances interaction among the individual variables [103, 106].
The vector notation in Equation (4.21) can be misleading as one coupling block contains two concatenated layers. The first one calculates y_1 from z_1 and z_2, the second one calculates y_2 from z_2 and y_1 [107, 108]. The Jacobian of this mapping is the product of two triangular matrices

$$\frac{\partial g_\theta(z)}{\partial z} = \begin{pmatrix} \mathbb{1} & 0 \\ \text{finite} & \operatorname{diag} e^{s_1(y_1)} \end{pmatrix} \cdot \begin{pmatrix} \operatorname{diag} e^{s_2(z_2)} & \text{finite} \\ 0 & \mathbb{1} \end{pmatrix}, \tag{4.22}$$

i.e. its determinant is the product of the diagonal entries

$$\left| \frac{\partial g_\theta(z)}{\partial z} \right| = \prod e^{s_2(z_2)} \prod e^{s_1(y_1)}. \tag{4.23}$$

The network can be trained with a maximum likelihood loss. This relies on the assumption that there is a dataset which encodes the complicated distribution $p_Y(y)$. To fit a model distribution $p_{\text{model}}(y, \theta)$ via g_θ, the log-likelihood loss needs to be minimised

$$L_{\text{INN}} = -\langle \log p_{\text{model}}(y, \theta) \rangle_{y \sim p_Y}, \tag{4.24}$$

where $\langle . \rangle$ implies the calculation of a mean value. Using Equation (4.20) this can be rewritten as

$$L_{\text{INN}} = -\left\langle \log p_Z(\overline{g}_\theta(y)) + \log \left| \frac{\partial \overline{g}_\theta(y)}{\partial y} \right| \right\rangle_{y \sim p_Y}. \tag{4.25}$$

The first term ensures that the latent representation remains Gaussian, while the second term constructs the correct transformation to the complicated distribution. Both terms can be computed efficiently. It is possible to derive this maximum likelihood approach by minimizing a *Kullback-Leibler-divergence* [109] between the $p_Y(y)$ and $p_{\text{model}}(y)$, further details about this can be found in [93].
The INN provides a powerful generative model of an underlying data distribution. Additionally, its invertible structure seems advantageous for unfolding. Nevertheless, it has been shown that an INN is not the best network architecture to approach unfolding problems, because a naive INN implements a deterministic mapping between detector-level and truth-level events. This contradicts the statistical nature of unfolding and does not account for possible hidden degrees of freedom. Hence, it is advantageous to adapt a conditional setup instead [5, 78].

conditional Invertible Neural Networks (cINN) [110] are a modified version of the INNs discussed in the last paragraph. The main difference in the general setup is that the previous fully connected neural networks $s_{1,2}$ and $t_{1,2}$ from Equation (4.21) are now conditionalised, i.e. they obtain an external input. This structure is called a *conditional coupling block*. The cINN models the mapping g_θ between a complicated distribution $y \sim p_Y$ and a Gaussian latent space $z \sim p_Z$ under the condition $x \sim p_X$. Therefore only z and y need to have the same dimension. It is again possible to sample from the latent space for event generation after a successful training.
The cINN is trained with a loss function similar to Equation (4.24): the probability distribution for the network weights θ conditional on x and y needs to be maximised. The negative log-likelihood for the network parameters reads

$$L_{\text{cINN}} = -\left\langle \log p_{\text{model}}(\theta|x,y) \right\rangle_{x\sim p_X, y\sim p_Y} . \tag{4.26}$$

Bayes' theorem can be used to turn the probability for θ into a likelihood with y as the argument

$$\begin{aligned} L_{\text{cINN}} &= -\left\langle \log \frac{p_{\text{model}}(y|x,\theta) p_{\text{model}}(\theta|x)}{p_{\text{model}}(x|y)}) \right\rangle_{x\sim p_X, y\sim p_Y} \\ &= -\left\langle \log p_{\text{model}}(y|x,\theta) \right\rangle_{x\sim p_X, y\sim p_Y} - \log p_{\text{model}}(\theta) + \text{const.} \end{aligned} \tag{4.27}$$

Constant terms which are irrelevant for the normalization are dropped in the following. It is again possible to apply the coordinate transformation of Equation (4.20) to obtain a trainable Jacobian. In addition, a Gaussian prior for the weight distribution $p_{\text{model}}(\theta)$ with $1/2\sigma_\theta^2 := \lambda$ is assumed. Hence, the full loss function reads

$$L_{\text{cINN}} = -\left\langle \log p_Z(\overline{g}_\theta(y|x)) + \log \left| \frac{\partial \overline{g}_\theta(y|x)}{\partial y} \right| \right\rangle_{x\sim p_X, y\sim p_Y} - \lambda\, \theta^2. \tag{4.28}$$

The last term is a weight regularization preventing the weights to become too large. It is generally often referred to as a *weight decay*. The first two terms are the usual maximum

likelihood loss, but conditional on reconstruction-level events and trained on event pairs. At this point it becomes clear why a *Gaussian* latent space with mean zero and variance one is chosen: up to a constant, the first term is easily computed as

$$\log p_Z(\overline{g}_\theta(y|x)) = -\frac{\| \overline{g}_\theta(y|x) \|_2^2}{2}. \tag{4.29}$$

5. Machine Learning in Unfolding

In recent years, machine learning tools have been applied to the unfolding problem described in Section 3 [5, 77, 78]. The main advantage of the machine learning methods over the "classic" matrix-based algorithms is their computational scaling behavior in higher dimensions. The matrix-based algorithms introduce a binning and are based on a pseudo-inversion of the response matrix. For a multidimensional unfolding the number of necessary matrix cells scales exponentially. This scaling behavior makes a high-dimensional unfolding computationally unfeasible. Most analysis which use a matrix-based algorithm unfold only up to three dimensions simultaneously.
The fact that the matrix-based methods only unfold a limited amount of observables poses several problems. First, the detector response might be sensitive to so-called *hidden observables* which are not unfolded themselves. Especially in complex modern particle detectors, this dependency on hidden observables is realistic. This problem can be solved by unfolding the full phase space, i.e. every observable obtained from the detector system. In addition, this full phase space unfolding would make it possible to construct every other observable out of the unfolded ones. This is especially interesting for future theories which might show only in exotic variables [111, 112].
Another limitation of matrix-based algorithms is the information loss associated with the introduced binning itself, the exact position of the event inside the bin is lost. This is even more pressing, as the size of the bin is very important: too broad bins may lose fine structure resolved by the event distribution, too narrow bins introduce large bin-to-bin correlations.
These problems do not appear when using machine learning methods, because they introduce no binning and apply the training on an event-by-event basis. This unbinned approach enables the unfolding of all detector observables simultaneously, i.e. avoiding the problem of hidden observables entirely. In the following two popular machine learning unfolding algorithms are discussed: *OmniFold* [77] and *cINN unfolding* [5].

5.1. OmniFold

OmniFold [77] is using an iterative reweighting algorithm to obtain an estimator for the truth-level distribution of the data. One of the key concepts of OmniFold is the likelihood ratio

$$L[(w, X), (w', X')](x) = \frac{p_{(w,X)}(x)}{p_{(w',X')}(x)}, \tag{5.1}$$

with $p_{(w,X)}(x)$ being the probability density of x estimated from samples X with empirical weights w. As already discussed in Section 4.3 it is possible to approximate this likelihood ratio using a classifier to distinguish (w, X) from (w', X'). To include the training weights $\hat{w} \in \{w, w'\}$ into the training procedure of a classifier, the binary cross entropy loss from

Equation (4.16) can be adjusted to

$$L_{BCE} = -\sum_{i=1}^{n} \frac{\hat{w}_i}{\langle \hat{w} \rangle} \left(\hat{y}_i \log(y_i) + (1 - \hat{y}_i) \log(1 - y_i) \right), \tag{5.2}$$

where $\langle \hat{w} \rangle$ is the mean value of the weights which is introduced to avoid training instabilities. The approach is equivalent to training on the unweighted distribution. This can be shown explicitly in the case of Generative Adversarial Networks (GANs) and generalises to this case [113, 114].
The OmniFold algorithm is visualised in Figure 5.1. As in Section 3, the samples of the Monte Carlo simulation are labeled as $\tilde{t}$ for the truth- and $\tilde{r}$ for the detector-level distribution, while the experimentally measured samples are labeled as r with their associated ground truth t. Using Equation (5.1), the OmniFold iteration steps can be written down as

$$1. \quad w_n(\tilde{r}) = \nu_{n-1}(\tilde{r}) L[(1, r), (\nu_{n-1}, \tilde{r})], \tag{5.3}$$

$$2. \quad \nu_n(\tilde{t}) = \nu_{n-1}(\tilde{t}) L[(w_n, \tilde{t}), (\nu_{n-1}, \tilde{t})], \tag{5.4}$$

where w_n and ν_n are weights in the respective iteration. The first step calculates weights w_1 to reweight the detector-level Monte Carlo distribution to look like the detector-level data. The initial weights of the Monte Carlo simulation are usually set to $\nu_0 = 1$. The Monte Carlo events are paired on detector- and truth-level, w_1 can thus be pulled to truth-level. The second step ensures that the pulled weight distribution is valid, i.e. the truth-level Monte Carlo with weights ν_0 is assigned new weights ν_1 to match the new weighted Monte Carlo with weights w_1. This has the effect of smoothing out the weight distribution. After this first iteration, ν_1 is used as a new starting point for further iterations. This algorithm will continuously reweight the Monte Carlo simulation until a satisfying result is reached. As in the case of the classical unfolding algorithms, a bias towards the Monte Carlo in the final result will always appear, but is reduced with more iterations. Too many iterations on the other hand will increase the uncertainties. As always for iterative unfolding algorithms, it is crucial to find a good balance between bias and uncertainties.
The main challenge in the application of OmniFold is the choice of the phase space covered in the Monte Carlo simulation. The whole algorithm is a continuous reweighting. It is thus not easy to ensure that every structure present in the measured data will be transferred to the Monte Carlo. One example is an observable which is only defined on high jet multiplicities with more than ten jets. If there are ten-jet events in the data, but not in the Monte Carlo simulation, this observable will not be unfolded correctly bacause there is nothing to reweight the measured data to. The claim of OmniFold that it "simultaneously unfolds all possible observables" [77] is therefore an overstatement.
Nevertheless, in a clean environment OmniFold is a strong unfolding tool. It has already been applied to the experimental data of the H1 experiment, which took place at HERA [115]. The number of observables unfolded in this analysis shows clearly the potential of machine learning approaches over the classical matrix-based approaches. The data of

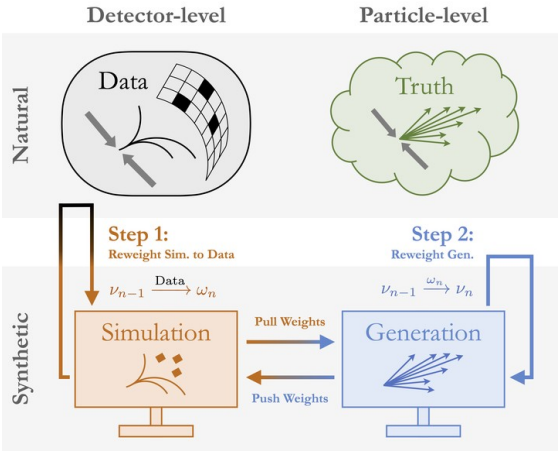


Figure 5.1: An illustration of the OmniFold algorithm. As a first step, starting from prior weights ν_0, the detector-level Monte Carlo is reweighted to match the detector-level measured data. The resulting weights w_1 are pulled back to induce weights on the truth-level Monte Carlo. As a second step, the initial Monte Carlo is reweighted to match the new weighted Monte Carlo. The resulting weights ν_1 are pushed forward to induce a new simulation, and the process is iterated. Source: [77].

the H1 detector is chosen because of the non-diverse possible final states and the overall very clean environment, to avoid the problem of a misjudgement of the phase space. In a fully involved ATLAS analysis this problem needs to be addressed in a more rigorous way, because the larger center of mass energy results in a broader overall phase space for the final states.

5.2. cINN Unfolding

Although OmniFold already offers a lot of possibilities in unfolding, it is not able to unfold single events. In addition, the developement of different methods is desirable to implement cross-checks. Hence, at this point it is suitable to introduce the cINN unfolding [5], which will be the main method used throughout this thesis.
The basic structure of a cINN was introduced in Section 4.4. The latent space z again follows a Gaussian distribution. Furthermore the detector-level data r serves as conditions while the particle-level data t is the network output. This idea is visualised in Figure 5.2.
During training, paired samples of detector- and particle-level events are passed through the network to the latent space. The cINN is still invertible in the sense that it includes a bi-directional training from Gaussian random numbers to truth-level events and back, but the invertible nature is not used to invert the detector simulation [93]. Once the training has converged, it is possible to sample from the latent space under the condition of a specific detector-level event to generate a distribution of truth-level events. Without

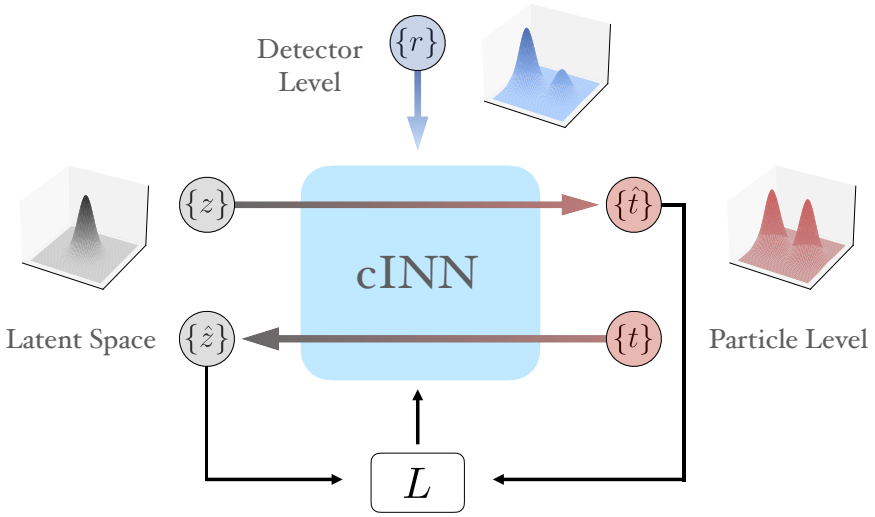


Figure 5.2: Structure of the conditional INN. Random numbers $\{z\}$ are mapped to truth-level events $\{t\}$ under the condition of a detector-level event $\{r\}$. The loss L follows Equation (4.28). A hat indicates a cINN-generated event. The training is performed in both directions [5].

the conditionality, the trained network would become an event generator which would reproduce the underlying distribution of particle-level events in the training data set.
In contrast to OmniFold the cINN unfolding obtains a probabilistic unfolded distribution for one single detector-level event, whereas Omnifold reweights only the distributions of events. This contrast in the approaches is desirable for later experimental purposes, in a sense that it motivates checking one with the other. An example of a successful cINN unfolding can be found in [5]. This example already shows the potential of cINN unfolding. To further check the validity of the cINN unfolding, it is interesting to observe its performance while introducing large differences between the experimental data and the Monte Carlo simulation, i.e. in a "worst-case" scenario. This idea is investigated further in Section 7.1.

6. Matrix-based Unfolding of Single Events

The main difference between machine-learning-based and matrix-based unfolding algorithms is the representation of the data: event-by-event or as a histogram with an introduced binning. Up to this point it was not possible to validate the single event unfolded distributions of the cINN unfolding. The goal of this section is to construct single event unfolded distributions using the matrix-based algorithms, to obtain a benchmark for the cINN single event unfolding.

6.1. Analytic Predictions

In the context of the toy model of Section 3.4 it can be analytically predicted how a single event unfolded distribution should look like. The contribution to the data distribution from a single measured event at r_m can be expressed with a delta distribution

$$g(r) = \delta(r - r_m), \tag{6.1}$$

on detector-level r. This distribution is unfolded to truth-level t by calculating

$$f_{u,1}(t) = \int p(t|r)g(r)\mathrm{d}r, \tag{6.2}$$

where $p(t|r)$ is still defined as the pseudo-inverted detector response function in Equation (3.39). The result is again a Gaussian function with parameters

$$\mu_{\text{single}} = \frac{\sigma_s^2\mu_t - \sigma_t^2(\mu_s - r_m)}{\sigma_s^2 + \sigma_t^2}, \qquad \sigma^2_{\text{single}} = \frac{\sigma_s^2\sigma_t^2}{\sigma_s^2 + \sigma_t^2}, \tag{6.3}$$

with the smearing parameters (μ_s, σ_s), and the parameters of the current prior (μ_t, σ_t). This result can be derived directly from Equation (3.41) by modifying $\mu_M \to r_m$ and $\sigma_M \to 0$. According to Section 3.4, the prior parameters (μ_t, σ_t) can be updated iteratively. The result from above can therefore be used to predict a single event unfolded distribution in each iteration by plugging in the corresponding updated prior. By inserting the parameters of the truth-level data, $\mu_t = \mu_{\text{Truth}}$ and $\sigma_t = \sigma_{\text{Truth}}$, it is possible to derive a result for an infinite number of iterations. The iterative predictions are used for IBU and IDS, whereas the exact result serves as a benchmark for SVD.

6.2. Weighting Approach

For matrix-based unfolding approaches, it is only possible to calculate one single event unfolded distributions for each bin, i.e. it is still not possible to treat all the events individually. This is due to the loss of the exact value of each event, which is connected to the introduced binning. The bin size can be decreased only to a certain point until large bin-to-bin correlations appear. Nevertheless, it is still advantageous to unfold one single bin, especially for distributions with small gradients. In this section the Monte Carlo simulation is again labeled as $(\tilde{t}_j, \tilde{r}_i)$ and the experimental data is labeled as (t_j, r_i),

on truth-level t and detector-level r respectively. It is possible to unfold r_i by using the unfolding algorithms described in Section 3.
A naive way to obtain a single event distribution is the use of a small perturbation: the unfolding procedure is performed once with the original distributions, once with a slightly changed detector-level distribution r_i. A single bin i_s of the measured distribution is modified by multiplying it with a weighting factor $w > 1$. The single event unfolded histogram $e_j(i_s)$ is defined as the normalized difference between the unfolded distributions

$$e_j(i_s) = \frac{1}{w-1}\left(u'_j - u_j\right), \tag{6.4}$$

using the original unfolding result u_j of the original detector-level distribution as well as the unfolding result u'_j derived from the reweighted detector-level data. This algorithm can be applied to the analytic toy model.
The results are shown in Figure 6.1. The parameters for IDS were chosen as in Section 3.4. Due to their conservative choice it is not surprising that IBU and IDS behave approximately identically. In the first iteration the single event unfolded distribution matches its analytic prediction. The single event distributions of the second and fourth iteration do not match their analytic predictions, instead there is a small shift as well as negative entries. This behavior can be explained by looking at the algorithms more closely: the first iteration is the pseudo-inversion of the response matrix using Bayes' theorem. This pseudo-inversion is based solely upon the Monte Carlo simulation, which was unchanged in both the weighted and not weighted case. The pseudo-inverse, i.e. the unfolding matrix, is therefore not impacted by the changed detector-level data and the extracted single event unfolded distribution is exactly a column of this matrix. This procedure seems to obtain nearly perfect results for both algorithms.
After this first iteration the unfolded (potentially weighted) detector-level data is set to be the new prior for the pseudo-inversion which means the unfolding matrix is impacted by the changed detector-level data distribution. In the examples of Figure 6.1 a factor of $w = 1.01$ was used, thus the height of one bin of the detector-level data was increased by one percent. To compensate this increase during the iterative unfolding, the values of each column in the response matrix are slightly shifted towards the increased bin. Since one column still needs to be normalized, the values of the other bins are slightly decreased, which leads to negative entries after the calculation of the difference. Since the distribution is still required to be smooth, the bins next to the reweighted one are impacted stronger than bins far away.
There are two obvious problems with the single event distributions of the iterative approaches beyond the first iteration: their non-positivity as well as the strong shape difference to the analytic prediction. In addition, the sum of all the individual unfolded event distributions is not perfectly equal to the full detector-level data distribution unfolded. This mismatch is founded in the fact that this approach is only an approximation of the single event distribution, which can be improved with $w \to 1$. This behavior, as well as the result of the first iteration, shows that it might be a better approach to extract a particular column of the unfolding matrix after each iteration instead.
The single event distribution of the SVD algorithm is shown in the lower part of Figure

6.1. The dashed line is the true single event distribution, which was calculated analytically with Equation (6.3). It can be seen that the unfolded single event distribution is far away from the analytic prediction and introduces asymmetric negative entries as well. Since the SVD algorithm is not able to unfold the full toy spectrum (see Section 3.5), it cannot be expected to obtain a proper single event unfolded distribution.
It can be concluded that the techniques presented in this project are not sufficient to derive a single event unfolding for the SVD algorithm in this specific example. It is therefore decided to drop the SVD algorithm at this point, since the possibility of comparing iterative machine learning algorithm to the iterative matrix-based algorithms remains.

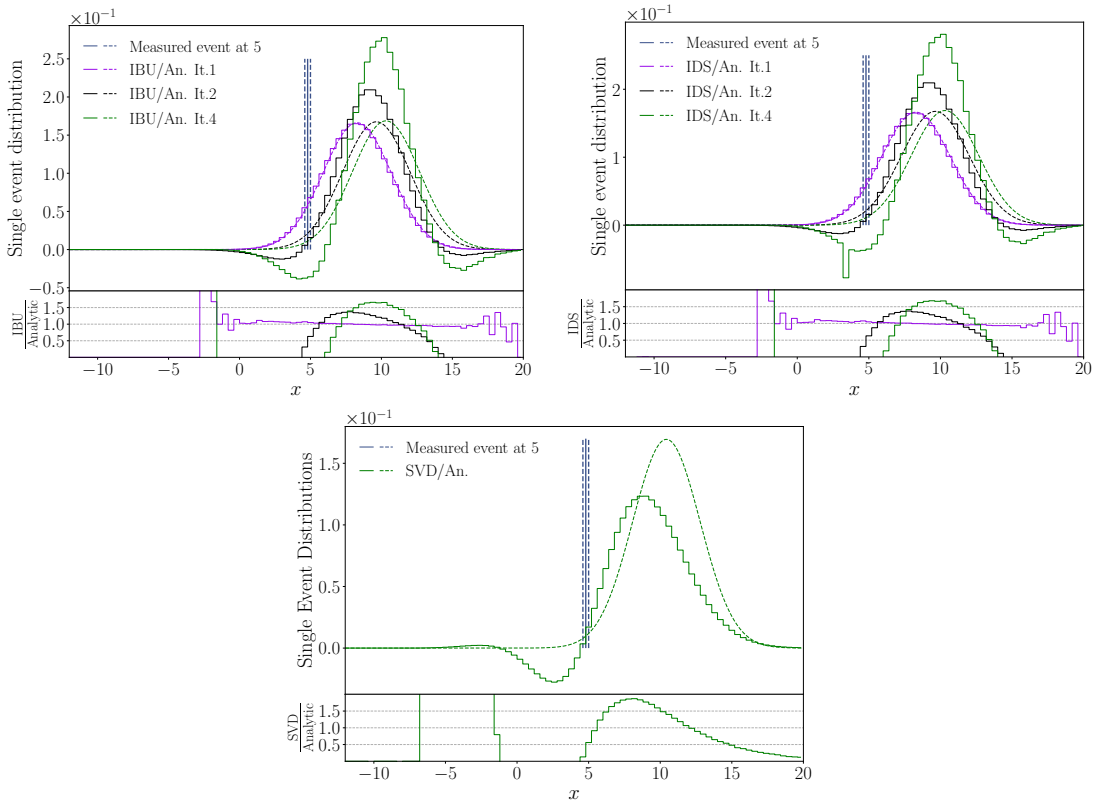


Figure 6.1: Single event unfolding using a weighting approach. In the upper left the result for IBU is shown, on the upper right the result for IDS. The blue bin is unfolded. The dashed lines (violet, black, green) are the analytic predictions for each iteration, the solid lines (violet, black, green) are the unfolding results of the single event distributions after each iteration. In the first iteration the single event unfolded distributions match their analytic predictions. For higher iterations, there is a significant bias. The lower plot shows the result for SVD. The analytic prediction (dashed green) and the unfolding result (solid green) do not agree. A weighting factor $w = 1.01$ is used for these examples.

6.3. True Single Event Distributions

As seen in the last section, the reweighting approach to obtaining single events unfolded is not working perfectly, because the summed single event unfolded distributions differ slightly from the full histogram unfolded distribution. In addition, the single event distributions can contain negative entries and differ from their analytic predictions. As explained in the last section, there is a better possibility for the iterative algorithms:

1. Perform all iterations with the original distributions.
2. Extract the final unfolding matrix (i.e. the current pseudo-inverted response matrix) in the last step.
3. Apply the final unfolding matrix to a histogram with one single event as defined in the unfolding algorithm.

This approach is based on the fact that the measured spectrum is multiplied with the unfolding matrix. This is a linear operation, hence it should not make a difference whether the measured events are summed up to the measured distribution before or after this multiplication. Nevertheless, the unfolding matrix still needs to be derived using the full histogram.
For IBU this method is implemented straight-forward by extracting the final unfolding matrix and extracting one column. The IDS algorithm has a more involved way of deriving the unfolded distribution as discussed in Section 3. This has to be represented in the derivation of the single event unfolded distribution.
To get an event-by-event representation of the IDS algorithm, it is necessary to adjust the final formula for the unfolded distribution derived in Section 3.1 as

$$\begin{aligned} u_j = \quad & \frac{N_D}{N_{\mathrm{MC}}} \cdot \tilde{t}_j \\ & + \sum_i \Big[f(|\Delta r_i|, \hat{\sigma}(r_i), \lambda_U) \cdot \Delta r_i \cdot \tilde{R}_{ji} \\ & + (1 - f(|\Delta r_i|, \hat{\sigma}(r_i), \lambda_U)) \Delta r_i \delta_{ij} \Big], \end{aligned} \tag{6.5}$$

with the unfolded distribution u_j, the Monte Carlo normalization factor N_D/N_{MC}, the truth-level Monte Carlo $\tilde{t}_j$, the unfolding matrix $\tilde{R}_{ji}$, the regularization function f, the regularization parameter λ_U, the norm-corrected difference between data and Monte Carlo on detector level Δr_i and its uncertainty $\hat{\sigma}(r_i)$. Using the fact that unfolding the detector-level Monte Carlo $\tilde{r}_i$ with the unfolding matrix $\tilde{R}_{ji}$ yields the truth Monte Carlo $\tilde{t}_j$, Equation (6.5) can be re-written as

$$\begin{aligned} u_j = \quad & \sum_i \Big[\left(\frac{N_D}{N_{MC}} \cdot \tilde{r}_i + f(|\Delta r_i|, \hat{\sigma}(r_i), \lambda_U) \cdot \Delta r_i \right) \tilde{R}_{ji} \\ & + (1 - f(|\Delta r_i|, \hat{\sigma}(r_i), \lambda_U)) \Delta r_i \delta_{ij} \Big], \end{aligned} \tag{6.6}$$

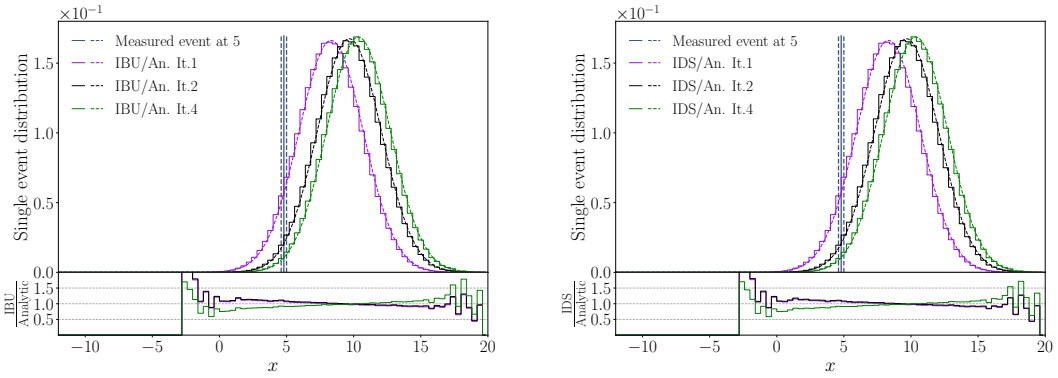


Figure 6.2: True single event unfolding for the iterative matrix-based algorithms. The bin of the single event is drawn in blue. The left plot shows the single event unfolded distributions for IBU, the right one for IDS (violet, black, green; solid lines). The analytic expectations (respective dashed lines) are always matched.

with $r_i = \Delta r_i + \frac{N_D}{N_{MC}} \tilde{r}_i$. The summation $\sum_i$ includes all bins in the detector-level data. To get a proper single event distribution of a detector-level event the sum over bins i is dropped and the fixed bin value i_s is used instead. In addition, the number of entries in the detector-level bin i_s has to be accounted for.
First, the transformation of $\frac{N_D}{N_{MC}} \cdot \tilde{r}_{i_s}$ in Equation (6.6) should be normalized to one by replacing it with $\frac{N_D}{N_{MC}} \cdot \tilde{r}_{i_s}/r_{i_s}$. Similarly, for the contribution of Δr_{i_s} the per-event unfolded distributions should be normalized to $\frac{\Delta r_{i_s}}{r_{i_s}}$. An event in bin i_s of the detector-level data is consequently unfolded to a single event distribution $e_j(i_s)$ calculated as

$$\begin{aligned} e_j(i_s) = \quad & \frac{N_D}{N_{MC}} \cdot \frac{\tilde{r}_{i_s}}{r_{i_s}} \cdot \tilde{R}_{ji_s} \\ & + \frac{\Delta r_{i_s}}{r_{i_s}} \cdot \Big[f(|\Delta r_{i_s}|, \hat{\sigma}(r_{i_s}), \lambda_U) \cdot \tilde{R}_{ji_s} \\ & + (1 - f(|\Delta r_{i_s}|, \hat{\sigma}(r_{i_s}), \lambda_U)) \cdot \delta_{ji_s} \Big]. \end{aligned} \tag{6.7}$$

It can be shown that this distribution is normalized, i.e. $\sum_j e_j(i_s) = 1$. In conclusion, it is possible to obtain the single event unfolded distributions by performing a usual IDS unfolding and extracting all necessary variables of Equation (6.7) to construct the single event distribution for each event of the detector-level data.

The toy model results for the single event unfolded distributions for IBU and IDS after each iteration are shown together with their analytic prediction in Figure 6.2. The distributions show that there are no more negative entries. This is expected, because the pseudo-inverse of the response matrix cannot contain negative entries, even for multiple iterations. In addition, the distributions match the analytic predictions in the regions

of high statistics, both for IBU and IDS. With this method, the individual single event distributions sum up to the full unfolded distribution exactly.
It can be concluded that this way to calculate the single event unfolded distributions is valid and does not suffer the inconsistencies of the weighting approach in the previous section. It is therefore possible to compare the single event distributions with an iterative machine learning unfolding algorithm which predicts single event unfolded distributions as well. Since OmniFold is a data-driven reweighting, it cannot be compared to this single event unfolding. In the next chapter a cINN-based, iterative unfolding algorithm is introduced which allows comparisons with the above derived single event distributions of matrix-based unfolding.

6.4. Single Event Uncertainties

The next step is to calculate the covariance matrices of the single event distributions. For validation it needs to be possible to calculate the covariance matrix of the full unfolded distribution by using the covariances of the single event unfolded distributions. *Bootstrapping*, i.e. the Poissonian fluctuation of the response matrix or the detector-level data is used to estimate the covariance matrix as explained in Section 3.3:

$$\mathrm{cov}_{kl} = \frac{1}{N_{\mathrm{toys}}} \sum_{n=1}^{N_{\mathrm{toys}}} (u_k^{(n)} - \overline{u}_k)(u_l^{(n)} - \overline{u}_l), \tag{6.8}$$

with the bins of the unfolded distribution u_j and the indices (k, l) in the range of the truth-level bins. The index (n) shows that the current unfolded distribution depends on the fluctuations of the current toy model, $\overline{u}_j$ is the mean value of this bin entry for all toys.
To derive the (rather technical) connection between the single event covariances and the full distribution covariance the following notation is used:

- $e(i_s)$ = truth-level distribution resulting from the unfolding of a single event in bin i_s of the detector-level data,
- $e^{(n)}(i_s)$ = single event unfolded truth-level distribution, including Poissonian fluctuations, indicated with the number of the toy (n),
- $e_k^{(n)}(i_s)$ = k'th bin of the $e^{(n)}(i_s)$ distribution,
- $\overline{e_k(i_s)}$ = mean value of $e_k^{(n)}(i_s)$ over all toys,
- $B(i_s)$ = number of events in bin i_s of the experimental detector-level data distribution which is unfolded,
- $\mathrm{cov}_{kl}(e(i_1), e(i_2))$ = covariance between two single event unfolded distributions. It is neither necessary nor forbidden that $i_1 = i_2$.

It is possible to calculate the covariance between two single event unfolded distributions of the measurement:

$$\mathrm{cov}_{kl}(e(i_1), e(i_2)) = \frac{1}{N_{\mathrm{Toys}}} \sum_{n=1}^{N_{\mathrm{Toys}}} \left(e_k^{(n)}(i_1) - \overline{e_k(i_1)}\right) \cdot \left(e_l^{(n)}(i_2) - \overline{e_l(i_2)}\right). \tag{6.9}$$

The uncertainties of a bin $e_j(i_s)$ of a single event unfolded distribution $e(i_s)$ can be calculated as

$$\sigma_j = \sqrt{\mathrm{cov}_{jj}(e(i_s), e(i_s))}, \tag{6.10}$$

while the correlations of the single event unfolded distribution $e(i_s)$ are given as

$$\mathrm{cor}_{kl}(e(i_s), e(i_s)) = \frac{\mathrm{cov}_{kl}(e(i_s), e(i_s))}{\sigma_k \cdot \sigma_l}. \tag{6.11}$$

The full unfolded distribution $u_j^{(n)}$ of one toy can be expressed in terms of the corresponding bins of the single event unfolded distributions $e_j^{(n)}(i_s)$ as

$$u_j^{(n)} = \sum_{i_s=1}^{N_{\mathrm{bins}}} B(i_s) \cdot e_j^{(n)}(i_s), \tag{6.12}$$

with N_{bins} being the number of bins on detector-level. This equation originates in the fact that every single event unfolded distribution can contribute to every bin of the full unfolded distribution. Equation (6.12) can be plugged into Equation (6.8) to calculate the covariance of the full distribution

$$\begin{aligned}\mathrm{cov}_{kl} &= \frac{1}{N_{\mathrm{toys}}} \sum_{n=1}^{N_{\mathrm{toys}}} \left(\left(\sum_{i_1=1}^{N_{\mathrm{bins}}} B(i_1) \cdot \left(e_k^{(n)}(i_1) - \overline{e_k(i_1)}\right) \right) \left(\sum_{i_2=1}^{N_{\mathrm{bins}}} B(i_2) \cdot \left(e_l^{(n)}(i_2) - \overline{e_l(i_2)}\right) \right) \right) \\ &= \sum_{i_1=1}^{N_{\mathrm{bins}}} \sum_{i_2=1}^{N_{\mathrm{bins}}} \frac{1}{N_{\mathrm{toys}}} \sum_{n=1}^{N_{\mathrm{toys}}} B(i_1) B(i_2) \left(e_k^{(n)}(i_1) - \overline{e_k(i_1)}\right) \left(e_l^{(n)}(i_2) - \overline{e_l(i_2)}\right). \end{aligned} \tag{6.13}$$

The two different ways to apply the Poissonian fluctuations need to be treated differently because of their impact on the number of entries in the detector-level measured distribution $B(i_s)$.
If a fluctuation of the response matrix is applied, the entries of the detector-level data r_i and in consequence $B(i_s)$ are constant. This makes it possible to move $B(i_1)$ and $B(i_2)$ out of the sum over all toys in Equation (6.13) to obtain

$$\mathrm{cov}_{kl} = \sum_{i_1=1}^{N_{\mathrm{bins}}} \sum_{i_2=1}^{N_{\mathrm{bins}}} B(i_1) B(i_2) \mathrm{cov}_{kl}(e(i_1), e(i_2)). \tag{6.14}$$

This relation allows for a closure test. The interpretation of Equation (6.14) is straightforward: the sums $\sum_{i_s=1}^{N_{\mathrm{bins}}} B(i_s)$ give one contribution for each single event, while multiple

events in the same bin deliver the same contribution. Equation (6.14) is thus only a summation of all possible combinations of covariances between two single event distributions. This result is based on the fact that $B(i_s)$ is not impacted by the Poissonian fluctuations of the response matrix.
The second possibility is that the Poissonian fluctuation is applied to the detector-level experimental data which needs to be unfolded. Per definition this leads to a variation of $B(i_s)$ and makes it dependent on the toy model index: $B(i_s) = B(i_s)^{(n)}$. In consequence it is impossible to move the factors $B(i_1)^{(n)}$ and $B(i_2)^{(n)}$ out of the summation over the toys, thus the covariance of a single event $\mathrm{cov}_{kl}(e(i_1), e(i_2))$ cannot be plugged in. Nevertheless, it is possible to rewrite Equation (6.13) in analogy to Equation (6.14) as

$$\begin{aligned}\mathrm{cov}_{kl} = &\sum_{i_1=1}^{N_\mathrm{bins}} \sum_{i_2=1}^{N_\mathrm{bins}} B(i_1)B(i_2) \\ &\cdot \frac{1}{N_\mathrm{toys}} \sum_{n=1}^{N_\mathrm{toys}} \frac{B(i_1)^{(n)}}{B(i_1)} \frac{B(i_2)^{(n)}}{B(i_2)} \left(e_k^{(n)}(i_1) - \overline{e_k(i_1)}\right)\left(e_l^{(n)}(i_2) - \overline{e_l(i_2)}\right),\end{aligned} \tag{6.15}$$

where $B(i_s)$ without an index n is the unfluctuated number of entries. It is therefore an option to rewrite this equation like Equation (6.14) by redefining the single event covariance as the second part of the previous equation:

$$\begin{aligned}\mathrm{cov}_{kl}(e(i_1), e(i_2)) = &\frac{1}{N_\mathrm{toys}} \sum_{n=1}^{N_\mathrm{toys}} \frac{B(i_1)^{(n)}}{B(i_1)} \frac{B(i_2)^{(n)}}{B(i_2)} \\ &\cdot \left(e_k^{(n)}(i_1) - \overline{e_k(i_1)}\right)\left(e_l^{(n)}(i_2) - \overline{e_l(i_2)}\right).\end{aligned} \tag{6.16}$$

This formula introduces the correction factor $B(i_s)^{(n)}/B(i_s)$ in the single event covariance. Since $B(i_s)^{(n)}$ is drawn from a Poissonian with expectation value $B(i_s)$, the correction factor will have nearly no impact on the majority of the single event distributions, as long as their corresponding detector-level bin contains enough events. The correction factor effectively implements that toys with extremely low number of events in the respective bin contain less information about the single event unfolded distributions of this bin and are therefore considered less important. In the same way toys with fluctuations towards a very high number of events in the respective bin are considered to contain more information. An implementation of both approaches for the detector-level data fluctuation (with and without the correction factor) in the considered examples showed, that the result does not change significantly. Nevertheless, to obtain numerical closure the correction factors are needed and are therefore used in the following.

As an example the covariances for the analytic toy model can be calculated. Since the results of IBU and IDS are very similar, it would be repetitive to use both approaches. Nevertheless, it is of course possible to implement it the same way for both of these iterative algorithms. In the following, the IBU algorithm is employed.
Since the covariance matrices are not easily human-readable, it is instructive to plot the

correlations and the square-root of the diagonal of the covariance, which are the uncertainties. Figure 6.3 shows the correlations of the full distribution after one and after four iterations, once for Poissonian fluctuations of the detector response and once for Poissonian fluctuations of the detector-level data. In addition, the evolution of the relative uncertainties over several iterations is shown. The number of events in the simulation is set to 500 000, the number of toys is $N_{\text{toys}} = 1000$.
The plots show a different behavior for the two kinds of fluctuations. The fluctuations of the response matrix, i.e. the fluctuation of the Monte Carlo simulation, lead to a correlation matrix which is nearly diagonal, showing that the bins are only weakly correlated. In contrast to this, the fluctuation of the detector-level data component introduces correlations with neighboring bins. This difference is due to the smearing of the detector response, which was determined in Section 3.4 by $\sigma_s = 3$. Fluctuating the entries of the response matrix only changes single bins in the final unfolding result, not multiple bins in a correlated way. Contrary to this, a bin of the detector-level data is smeared out to neighboring bins by the unfolding matrix. In consequence, a fluctuation of a detector-level bin is smeared out in a correlated way as well, hence introducing strong correlations to neighboring bins.
At this point it is interesting to look at the differences between the correlation matrix in the first iteration and the correlation matrix of the fourth iteration: while there are anti-correlations (=negative entries) after four iterations, the first iteration leads to purely positive correlations, which has been verified numerically. This can be easily understood remembering the fact that the first pseudo-inversion of the response matrix is unaffected by the detector-level data fluctuations. If a bin is fluctuated up or down statistically, this will induce the same upward or downward fluctuations in all the bins of the unfolded distribution. During the second iteration, the fluctuations of the detector-level data also impact the pseudo-inverted response matrix. This introduces anti-correlations, because the number of events needs to be conserved during the unfolding. In consequence the pseudo-inverted response matrix needs to be normalized along the truth-level axis. This normalization causes every upward change of a bin to be connected to a downward change of the other bins of the column, thus an anti-correlation is implemented. This behavior is clearly visible in the correlation matrices in Figure 6.3.
The uncertainties can be calculated using the square-root of the diagonal of the covariance matrix. As can be seen in the bottom panels of Figure 6.3, the expected behavior is apparent: the more iterations, the higher the uncertainty. As explained previously, the balance between bias and uncertainty is important while choosing the number of iterations.

The covariance matrices of single events are calculated as stated earlier in this section. The results are shown in Figure 6.4, again displayed as a combination of correlations and uncertainties. The covariance matrix of the full distribution is numerically equal to the sum of every single event covariance with every event (including itself) according to Equations (6.14) and (6.16). The fluctuations of the response matrix are again expected: as for the full distribution the correlation matrix of a single event is completely diagonal, the argumentation is analogous.

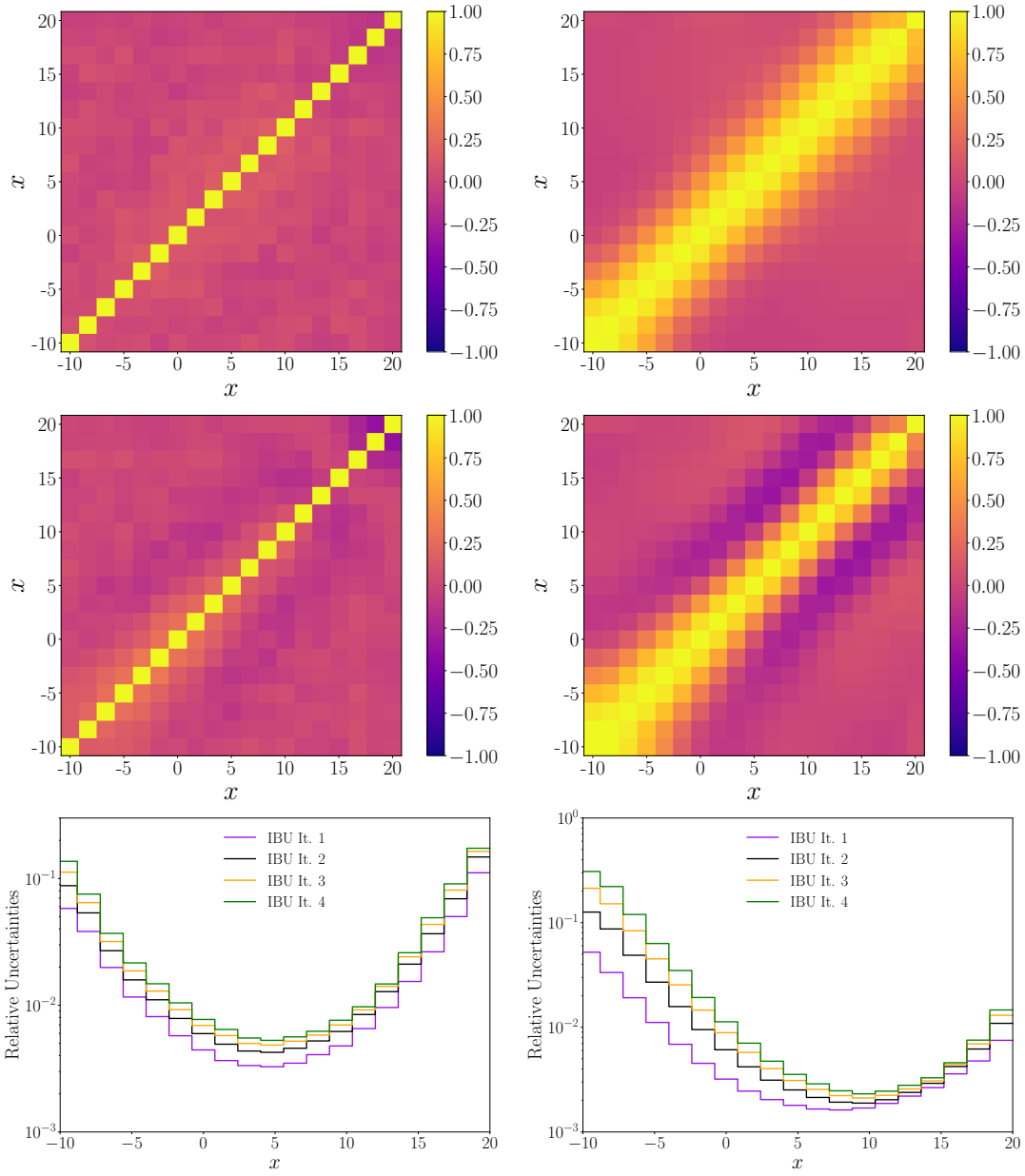


Figure 6.3: Representative graphs for the uncertainties of the full unfolded distribution obtained for the analytic example using IBU. On the left hand side fluctuations of the response matrix are implemented, on the right fluctuations of the detector-level data. The top row shows the correlation matrices after one iteration, the middle row after four iterations. The bottom row shows the evolution of the relative uncertainties over several iterations.

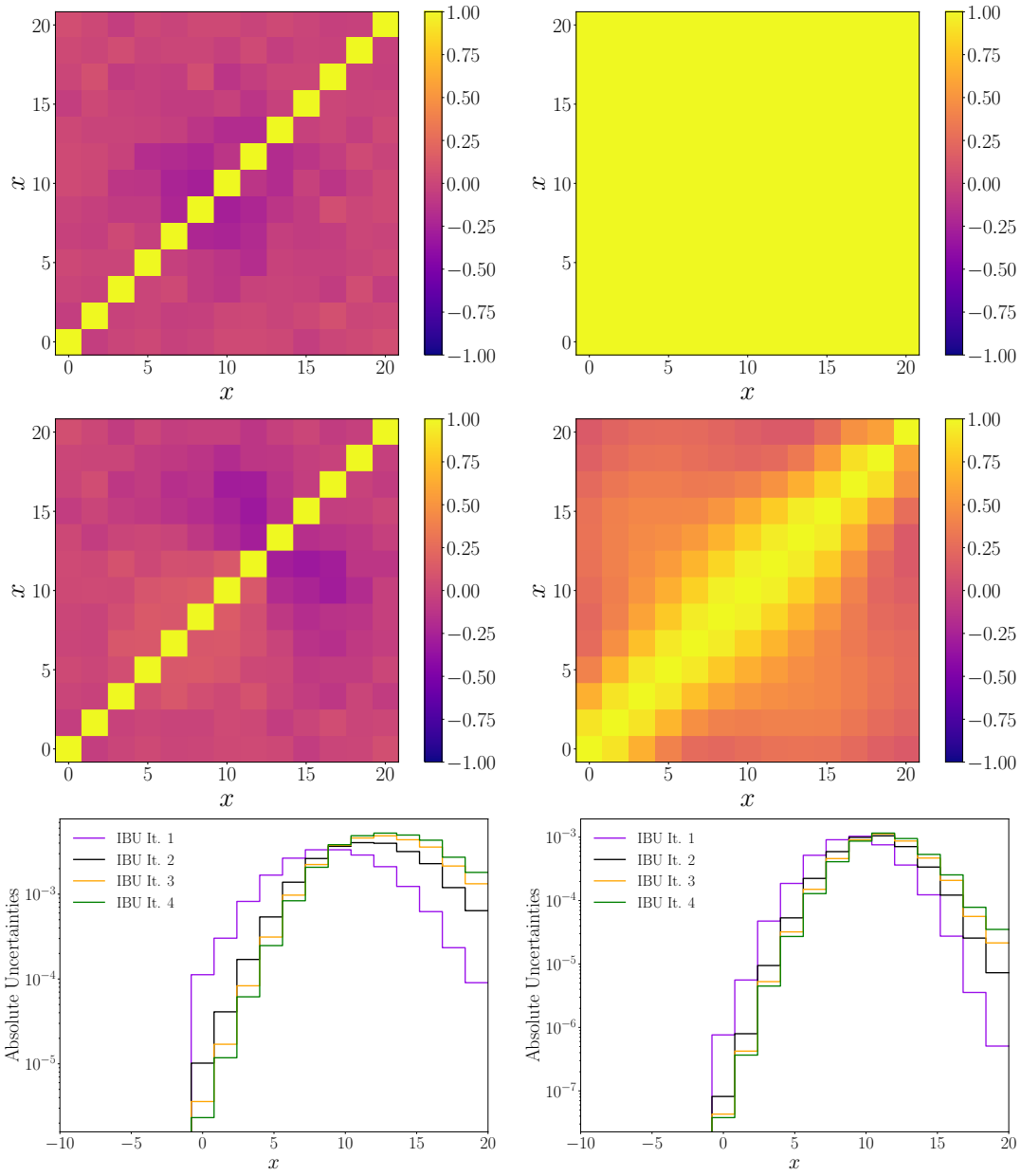


Figure 6.4: Representative graphs for the uncertainties of a single event unfolded distribution obtained for the analytic example using IBU. The single event is set at $r_m = 5$. On the left hand side fluctuations of the response matrix are implemented, on the right fluctuations of the detector-level data. The top row shows the correlation matrices after one iteration, the middle row after four iterations. The bottom row shows the evolution of the absolute uncertainties over several iterations.

The fluctuations of the detector-level data leads to much more interesting results: the correlation matrix contains only values which are very close to one, especially in the first iteration. This implies that there are very strong correlations between every bin of the single event unfolded distribution. This is due to the fact that the unfolding matrix is not impacted by the fluctuation of the detector-level data in the first iteration. Unfolding a single fluctuated bin will therefore yield the output of the unfluctuated bin, multiplied with a factor. Since this factor is equal for each bin of the single event unfolded distribution, there is a perfect correlation of one. With further iterations this effect is weakened because the fluctuations affect the pseudo-inverted response matrix, exactly the same as for the full distribution. The correlation matrix of the full distribution after the fluctuation of the detector-level data is not as close to one as the single event correlations. This is due to the fact that the full correlation matrix additionally contains contributions of correlations between two events which are not in the same bin. It was made sure that closure is achieved in each iteration.

The absolute uncertainties of the single event distribution are visualised in the bottom panel of Figure 6.4. The uncertainties of the single event distribution are not increasing significantly, instead they are shifted towards the right. This is due to the design of the toy model, where each iteration introduces such a shift. It can clearly be seen, that each event is impacting multiple bins on truth-level. In addition, to reconstruct the uncertainty of the full distribution, the covariances of two events from different bins again need to be added. The fact that there is no iterative increase of the uncertainty in the single event distribution therefore does not pose a problem.

7. Iterative cINN Unfolding

In this section the cINN unfolding is applied to the toy example introduced in Section 3.4. The main design choice for this toy example is to determine the cINN behavior under the "extreme" conditions of a total mismodeling by the Monte Carlo simulation as well as a large detector response. As it will turn out, an iterative approach for the cINN unfolding is necessary to reduce the bias towards the Monte Carlo continuously. This method will be called *Iterative cINN unfolding* (IcINN).

7.1. Limits of cINN Unfolding

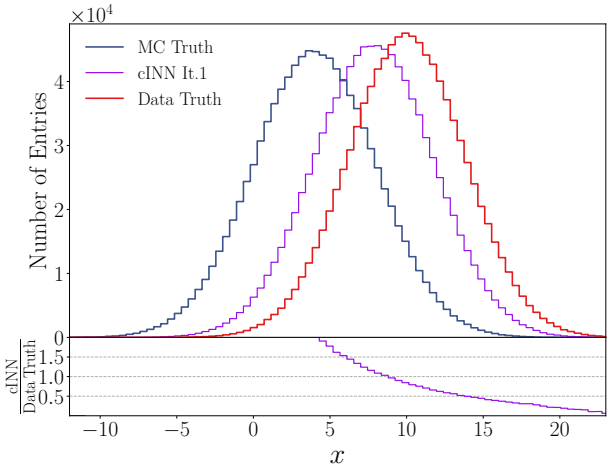
To demonstrate the limitations of cINN unfolding, it is applied to the analytic toy model introduced in Section 3.4. The detector-level information is used as a condition, while the truth-level (pseudo-)data serves as an output. The input is once again given by a Gaussian latent space, whose dimension matches the dimension of the truth-level data. Since the cINN needs to split the input and output as described in Equation (4.21), a second dimension has to be added which is trivial to the problem. This can be achieved by using Gaussian noise, i.e. in the second dimension the cINN only has to learn an identity mapping.
To further improve the performance of the cINN, a *preprocessing* is applied to the full dataset. For each input, the mean $\mu_{\text{Dist.}}$ and the standard deviation $\sigma_{\text{Dist.}}$ are calculated. With these values each value x of each distribution is normalized to

$$x' = \frac{x - \mu_{\text{Dist.}}}{\sigma_{\text{Dist.}}}. \tag{7.1}$$

After the transformation all values of the dataset have the same order of magnitude which matches the order of magnitude of the initialised network parameters.
In addition to this preprocessing, a one-cyclic learning rate is implemented as well as an ADAM optimizer, which is initialised with the standard values described in Section 4.2. Weight decay is already implemented naturally in the cINN loss in Equation (4.28). The result of the cINN unfolding and a table of the cINN parameters are given in Figure 7.1. The cINN does not achieve a good unfolding result, there is a strong bias towards the Monte Carlo simulation on which the cINN was trained. This problem can be understood in the context of Bayes' theorem, which can be used to calculate the pseudo-inverted response function

$$p(t|r) = \frac{p(r|t) \cdot p(t)}{p(r)}, \tag{7.2}$$

where $p(r|t)$ is the detector response, $p(t)$ and $p(r)$ are the prior of the truth- and detector-level distributions. The prior used in the cINN approach is the Monte Carlo simulation on which the cINN is trained. If this prior can be improved to be closer to the data while still keeping the same detector response, this will improve the final unfolding result. In analogy to the matrix-based Iterative Bayesian Unfolding (see Section 3.1), an improved Monte Carlo simulation is obtained by adapting an iterative setup.



Parameter	cINN
Conditional blocks	5
Layers (per block)	2
Units (per layer)	32
Epochs	100
Learning rate	10^{-4}
Max. learning rate	$3 \cdot 10^{-4}$
Weight decay	0.01
Batch size	4096
Training events	500 000

Figure 7.1: Result of the cINN unfolding applied to the toy example. The left image shows the unfolded distribution (purple) which is biased towards the Monte Carlo simulation (blue) and does not match the data truth (red). The table on the right shows the hyperparameters of the cINN. It was made sure that the performance of the cINN is stable for small changes of these hyperparameters.

7.2. Iterative cINN Unfolding

To iterate the cINN unfolding, a classifier is introduced to reweight the Monte Carlo simulation. This update leads to an iterative reduction of the bias while simultaneously increasing the uncertainties. The full *Iterative cINN unfolding* (IcINN) algorithm iterates three distinct steps. They are visualised in Figure 7.2:

1. **Train cINN:** First the cINN is trained on the plain Monte Carlo simulation to learn a pseudo-inverted detector response which has a bias towards the Monte Carlo.

2. **Predict:** In this step the cINN is applied to the measured data to obtain an unfolded distribution. This unfolding result will carry a bias towards the Monte Carlo.

3. **Reweight:** The truth-level component of the Monte Carlo simulation is reweighted to behave like the result of the unfolded distribution of the cINN unfolding. This reweighting is done on an event-by-event basis using a classification neural network. The calculated event weights on truth-level can be pulled to the detector-level Monte Carlo simulation to obtain a full reweighting of the Monte Carlo simulation.

The first two steps are exactly the same cINN unfolding used before, the new idea appears in step three: by reweighting the Monte Carlo simulation, a better starting point to retrain the cINN is obtained. It has proven to be computationally more efficient, if the cINN of the previous iteration is used as a new starting point. To train the cINN on

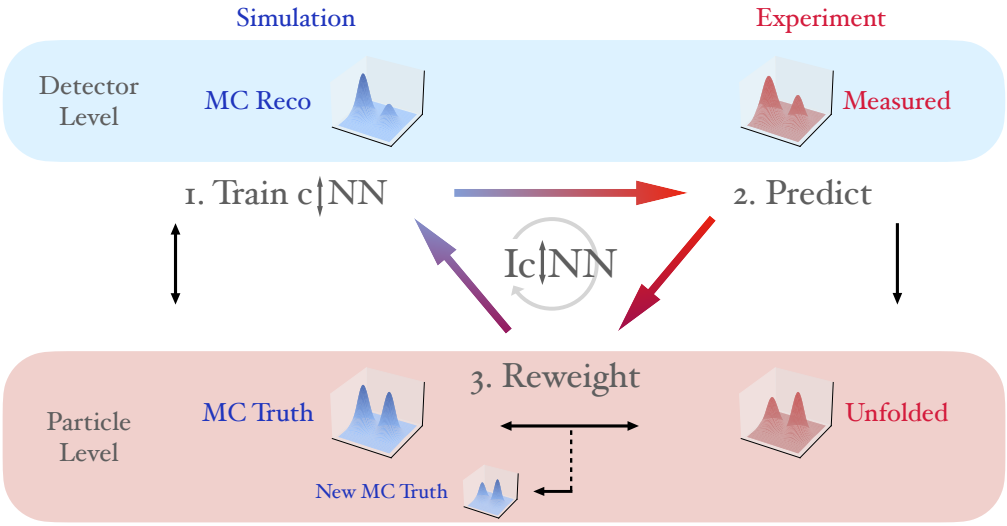


Figure 7.2: Illustration of the Iterative cINN unfolding algorithm. In a first step, the regular training of the cINN on the current Monte Carlo Data is performed. As a second step, the cINN unfolds the experimentally measured distribution. In a third step, the Monte Carlo simulation is reweighted to match the unfolded distribution *on truth-level*. This procedure is iterated, always with a modified Monte Carlo Simulation.

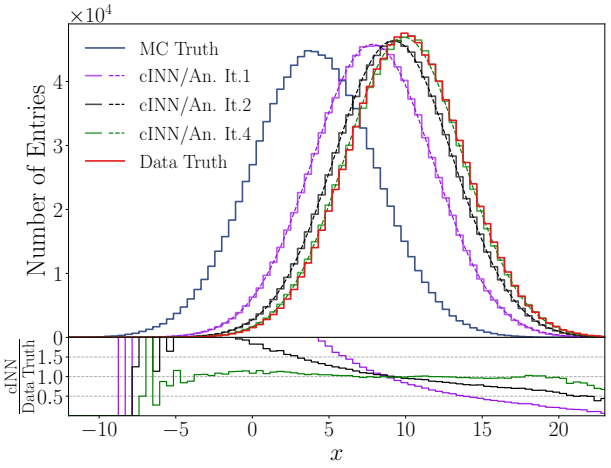
the weighted Monte Carlo, the loss function of the cINN (Equation (4.28)) is modified to account for the weights w. It can be constructed in analogy to Equation (5.2) as [116]

$$L_{\text{cINN}} = -\frac{1}{\langle w \rangle} \left\langle w(x) \cdot \log p_{\text{model}}(\theta|x,y) \right\rangle_{x\sim p_X, y\sim p_Y} . \tag{7.3}$$

The number of iterations is set to obtain a good balance between the bias towards the Monte Carlo and the uncertainties of the unfolded distribution.

7.3. Unfolding the Toy Example

In a next step the IcINN is applied to the analytic toy example introduced in Section 3.4. Since it is an iterative algorithm, it is possible to cross-check the unfolding result after each iteration with the analytic prediction.
The classifier in this example is build as a fully connected neural network with a single output neuron, which has a sigmoid activation function to restrict the classifier output between zero and one. As explained in Section 4.3 it is possible to reweight distributions towards another with such a classifier. To improve the classifier training, ADAM, a one-cyclic learning rate scheduler as well as the preprocessing of Equation (7.1) are



Parameter	Classifier
Layers	3
Units (per layer)	8
Epochs	100
Learning rate	10^{-3}
Max. learning rate	$3 \cdot 10^{-3}$
Weight decay	-
Batch size	4096
Training events	400 000

Figure 7.3: Iterative unfolding results for the $1D$ toy model. The left image shows the iterative unfolding results for iterations 1, 2 and 4 (violet, black, green; solid lines) together with the analytic prediction (respective dashed line). The MC reweighting leads to an improvement towards the data truth. In the lower part of the left plot the ratio of the unfolding results with the data truth is shown. It is clearly visible that the bias towards the Monte Carlo simulation is reduced with each iteration. On the right the hyperparametes of the classifier are shown.

implemented. The unfolding result and the hyperparameters of the classifier are given in Figure 7.3. The unfolded distribution of each iteration is very close to its analytic prediction. It is clearly visible that the bias towards the Monte Carlo simulation is iteratively reduced.
From this result it can be concluded that the algorithm is working properly. Nevertheless there are several possibilities for closure-checks:

1. Comparing the reweighted truth-level Monte Carlo simulation to the unfolded distribution of the previous iteration. They should be close to each other in the region of high statistics, because the classifier reweights the truth-level Monte Carlo to match the unfolding result. This validates the performance of the classifier.

2. Applying the cINN to the (potentially weighted) detector-level Monte Carlo simulation and comparing the result to the (potentially weighted) truth-level Monte Carlo. Since the cINN was trained on this Monte Carlo simulation it should be able to predict the truth-level Monte Carlo very well. This validates the performance of the cINN.

These closure checks have been performed. The results are given in the Appendix A.4.

In addition to the full distribution unfolding, it is possible to display the unfolded distribution of a single event. This is achieved by keeping the condition fixed and using a

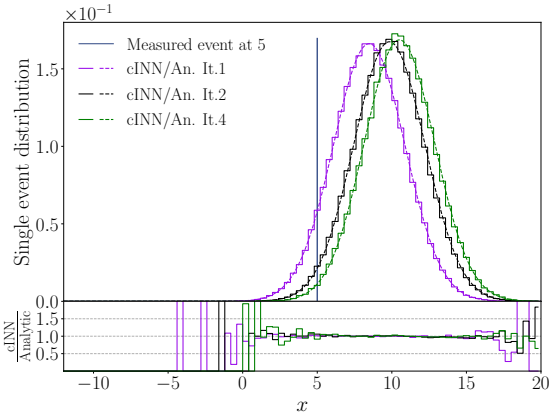


Figure 7.4: Unfolded distributions of a single event at $x = 5$ (blue). The single event unfolded distributions after each iteration (violet, black, green; solid) are very close to the analytic predictions (respective dashed lines).

batch of random numbers sampled from the latent space distribution. It is again possible to calculate an analytic prediction for the single event unfolded distribution, as for the Iterative Bayesian Unfolding in Section 6.1. The result is shown in Figure 7.4. It is very close to the analytic prediction. In conclusion, the IcINN behaves in this toy example very similar to the matrix-based Iterative Bayesian Unfolding.

7.4. Uncertainties

To estimate uncertainties of the unfolded distribution, the same techniques as for the matrix-based unfolding algorithms are applied (see Section 3.3). The cINN unfolding has a probabilistic nature by design. This is due to the fact that in order to generate the events, the cINN needs to sample from a Gaussian latent space. In addition, there is a dependency on the random initialisation of the network. These aspects already introduce a systematic uncertainty which is purely intrinsic to the machine learning algorithm.
Additionally, there are again uncertainties originating from the Monte Carlo simulation as well as the detector-level pseudo-data. Instead of fluctuating bins as before, each event is assigned a weight according to a Poissonian of one. This is applied either to the detector-level data or the Monte Carlo simulation. The final unfolding result changes for each of these *fluctuated toys*.
The covariances and correlations can only be visualised by introducing a binning to display them as a matrix. It is again instructive to look at the correlation matrices as well as the standard deviations instead of the covariance. Figure 7.5 and Figure 7.6 contain the correlation matrices after one and three iterations for the intrinsic correlations, the detector-level data fluctuations, the fluctuations of the Monte Carlo simulation as well as the total correlation. In addition, Figure 7.6 shows in the last row the relative uncertainties after one and three iterations.
The correlation matrices after one iteration show structures, which can be explained in

a similar way as for the binned algorithms (see Section 6.4). Nevertheless, the previous strict positivity for the detector-level data fluctuation correlations are affected by the intrinsic fluctuations of the cINN. This is clearly visible in Figure 7.5, the data fluctuation correlations are not purely positive anymore. In a similar manner the Monte Carlo fluctuations do not lead to a purely diagonal correlation matrix. The correlation matrices after three iterations show the expected behavior: more anti-correlations are introduced, for the data fluctuations as well as for the Monte Carlo fluctuations.
The relative uncertainties are displayed in the third row of Figure 7.6. The uncertainties are dominated by the Monte Carlo fluctuations in the middle of the distribution, the edges are dominated by the data fluctuations. It is visible that the relative uncertainties increase with more iterations, this is expected as well.

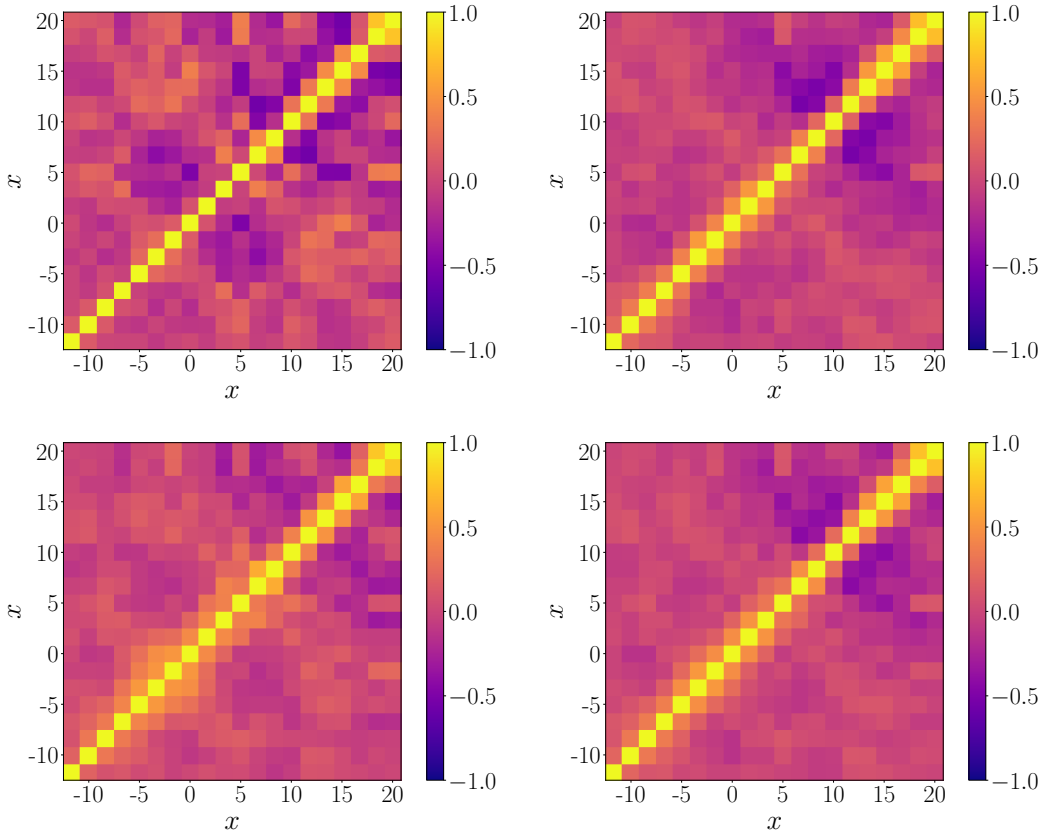


Figure 7.5: Correlation matrices of the IcINN results with one iteration, i.e. a pure cINN unfolding. There are the correlation matrices of the intrinsic fluctuation of the IcINN (top left), of the fluctuation of the Monte Carlo simulation (top right), of the fluctuation of the detector-level data (bottom left) and the total correlation, i.e. the sum of the Monte Carlo and the detector-level data fluctuation (bottom right). These matrices are calculated with $N_{\text{toys}} \approx 400$.

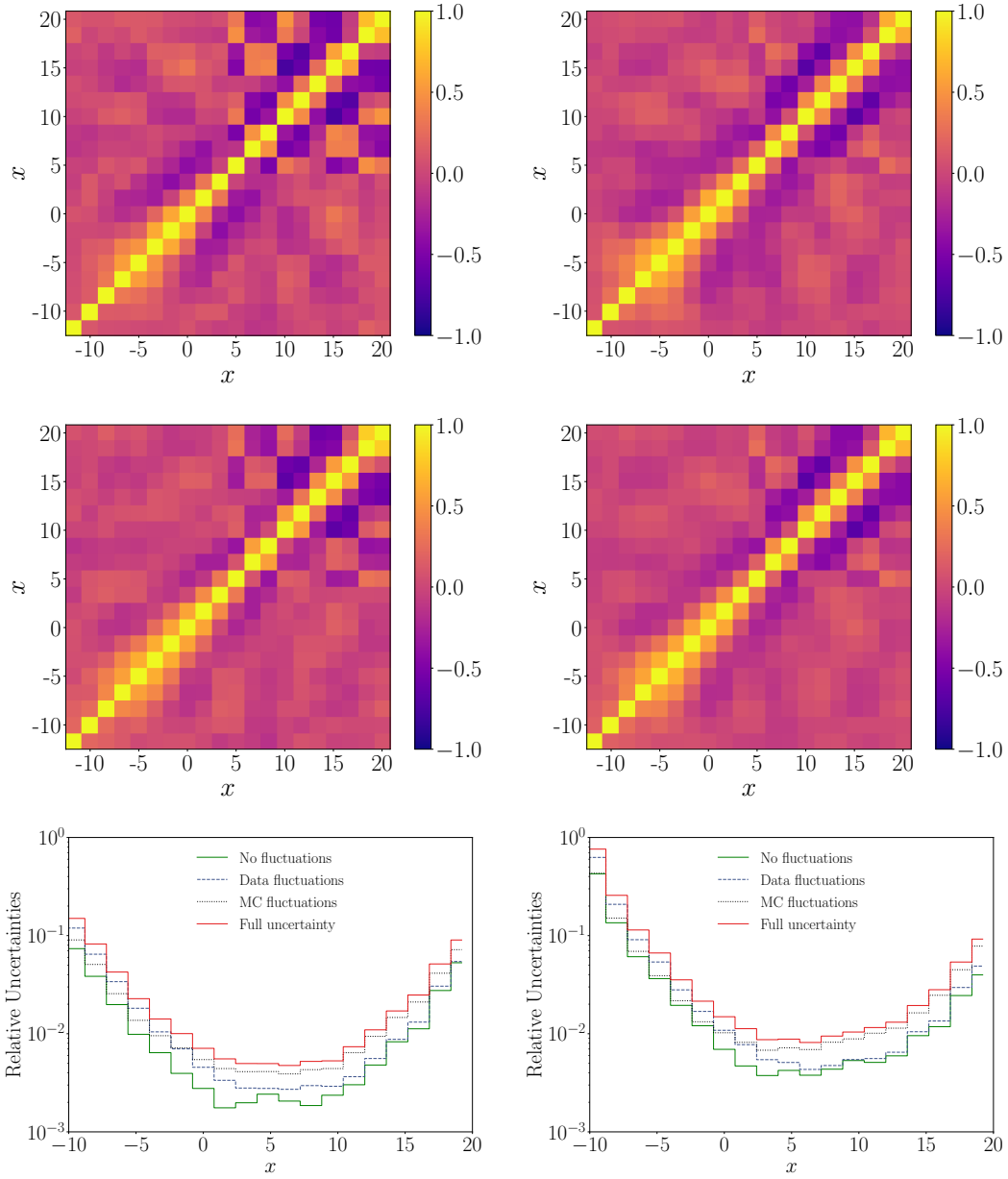


Figure 7.6: Correlation matrices of the IcINN results with three iterations as well as the uncertainties after one and three iterations. There are the correlation matrices of the intrinsic fluctuations of the IcINN (top left), of the fluctuation of the Monte Carlo simulation (top right), of the fluctuation of the detector-level data (center left) and the total correlation, i.e. the sum of the Monte Carlo and the detector-level data fluctuation (center right). In addition, there are the relative uncertainties after one (bottom left) and three (bottom right) iterations for each of these fluctuations. The expected increase of these uncertainties for more than one iteration is clearly visible. These matrices and uncertainties are calculated with $N_{\text{toys}} \approx 400$.

8. Unfolding of EFT Pseudo-Data

The IcINN approach to unfolding was demonstrated in the last chapter. The next logical step is to construct a more advanced example which involves a Standard Model cross-section. Furthermore, it is necessary to demonstrate that multi-dimensional distributions can be unfolded as well. The easiest way to do this is to replace the trivial Gaussian dimension of the last section with a second observable. The reference process used in this chapter is the production of $Z\gamma\gamma$ at a pp-collider measured with the ATLAS detector and simulated with and without an EFT contribution. The "data" used in this section is again simulated and not measured. In the following, the term "data" hence refers to distributions containing an EFT-contribution.

8.1. Generation of Pseudo-Data

The process $Z\gamma\gamma$ was introduced in Section 2.2. As explained, it is possible to construct an EFT contribution for this process which is significant. The necessary data for the unfolding process is created by simulating the process

$$pp \to Z\gamma\gamma, \qquad Z \to \mu^+\mu^-. \tag{8.1}$$

To generate the Monte Carlo simulation the pure Standard Model parametrisation is used, while the experimental (pseudo-)data contains an additional EFT contribution. The observables to unfold are the p_T distributions of the muons and the antimuon. These p_T distributions are very sensitive to EFT contributions introduced by $\mathcal{L}_{T,8}$ [48]. The truth-level event generation is performed using *MadGraph5* [8]. *Pythia8.308* [9] is used for the parton shower simulation. The necessary Wilson coefficient $C_{T,8}^{(8)}$ as well as the scale for new physics Λ are collectively set to

$$\frac{C_{T,8}^{(8)}}{\Lambda^4} = \frac{2}{\text{TeV}^4}. \tag{8.2}$$

This value has the same order of magnitude as current exclusion limits [6]. It is therefore a realistic setting for an analysis. The detector simulation is performed using *DELPHES 3.5.0* [10]. A standard ATLAS parameter-card is used. For simplicity and without loss of generality, the rapidity-dependence of the muon momentum smearing is removed. This modification is done in order to avoid hidden observables that have an impact on the detector smearing, while not being explicitly used in the IcINN. In an application of this algorithm to real data, the ultimate goal is to simultaneously unfold all observables measured by the detector, which will avoid this problem. The explicit momentum smearing is given as

$$\Delta p_T = p_T \cdot \sqrt{0.025^2 + 3.5 \cdot 10^{-8} \cdot \left(\frac{p_T}{\text{GeV}}\right)^2}. \tag{8.3}$$

A plot of the momentum smearing is given in the Appendix A.5. A low-p_T cut of the full dataset is applied at 10 GeV. In order to avoid covering too many orders of magnitude

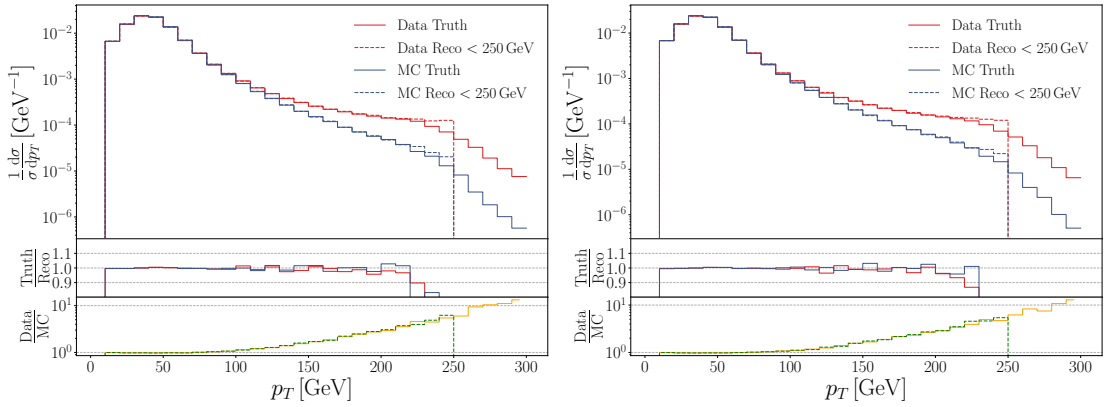


Figure 8.1: p_T distributions of the muon (left) and the antimuon (right). The plots show the experimental pseudo data (red) containing the EFT contibution as well as the Monte Carlo simulation (blue) based on the pure Standard Model. The detector-level distributions (dotted) were cut at 250 GeV.

in the training data, an additional high-p_T cut at 250 GeV is implemented for the reconstructed muons. Alternatively, it is possible to train several cINN's for various different p_T-ranges. The full dataset containing the Monte Carlo and the pseudo-data are shown in Figure 8.1, both on detector-level and on truth-level.
In addition to the distributions themselves, it is insightful to consider the response matrix R, i.e. the migration matrix normalized along the detector-level axis. Since the detector response is the same for the experimental data and the Monte Carlo simulation, they should look the same up to statistical fluctuations. This can be seen in the upper part of Figure 8.2. If the migration matrix is instead normalized along the truth-level axis (according to Equation (3.15)), the result is the posterior pseudo-inverted detector response matrix $\tilde{R}$. This matrix is based on a prior and carries not only information about the detector response, but also about the shape of the represented distribution. It is therefore not surprising that the differences between the two pseudo-inverses $\tilde{R}$ of the experimental pseudo-data and the Monte Carlo simulation are more significant than the differences between their response matrices. This can especially be seen in the high-p_T-region of $\tilde{R}$ shown in the lower part of Figure 8.2.

8.2. Unfolding $Z\gamma\gamma$

In this section, the IcINN is applied to the $Z\gamma\gamma$ data in order to unfold the experimental detector-level pseudo-data containing the EFT contribution. As always the truth-level EFT data is used to validate the unfolding result. It is possible to implement another preprocessing which is applied previous to the resizing of Equation (7.1): the truth-level Monte Carlo simulation $\tilde{t}$ is re-calculated according to

$$\tilde{t}' = \log\left(\frac{\tilde{t} - 10\,\text{GeV}}{\text{GeV}}\right). \tag{8.4}$$

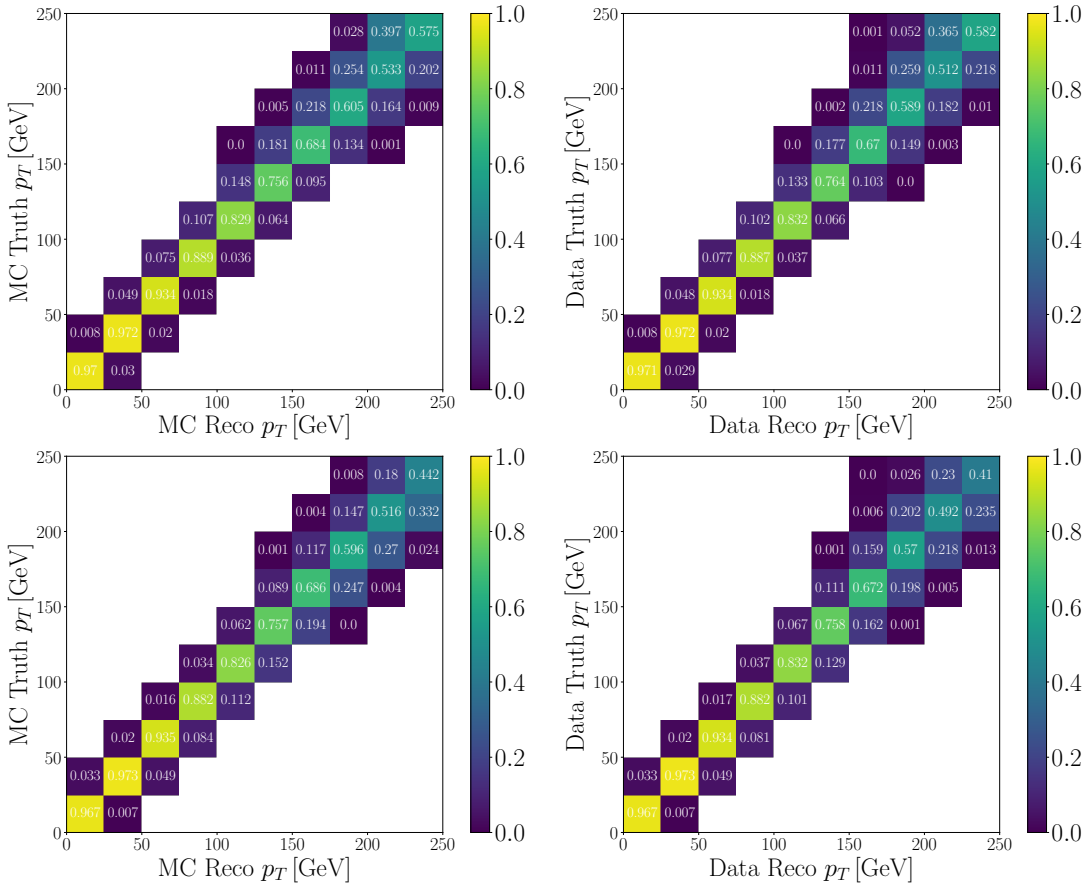


Figure 8.2: Migration matrices of the muon-p_T with different normalization. The upper left plot shows the response matrix of the Monte Carlo simulation, the upper right plot the response matrix of the experimental data. They are equal up to statistical fluctuations. In the lower half, the pseudo-inverted detector response matrices are shown for various distributions: the Monte Carlo simulation on the left, the experimental data on the right. Since the pseudo-inverted response matrices also depend on the shape of the represented distributions, the differences between them are more significant than for the response matrix.

This has the effect that it maps the distribution to look more like a Gaussian, which simplifies the training procedure. The value 10 GeV is chosen to match the low-p_T cut of the data. For different low-p_T cuts it needs to be adjusted. In order to obtain proper p_T-values after the unfolding procedure, the predicted truth-level data t'_{unf} is corrected by using the inverse mapping

$$t_{\text{unf}} = \left(10 + \exp(t'_{\text{unf}})\right) \text{GeV}. \tag{8.5}$$

The parameters of the IcINN unfolding are displayed in Table 2. Before displaying the final unfolding result it is necessary to validate the cINN and the classifier separately. As explained in Section 7.3, it is possible to check the classifier performance by comparing the reweighted truth-level Monte Carlo simulation to the previous unfolding result of the cINN. The distributions for various iterations are shown in Figure 8.3. A good agreement is observed, validating the performance of the classifier.
In addition, it is possible to perform a similar check for the cINN: applying the cINN to the (potentially weighted) detector-level Monte Carlo should result in a distribution which is very close to the (potentially reweighted) truth-level Monte Carlo. The results for this comparison are shown in Figure 8.4: the distributions agree in each iteration. Furthermore, these plots contain the truth-level data distribution as well as the unfolded detector-level data distribution. This comparison indicates an improvement with the increasing number of iterations. The closure check for the Monte Carlo simulation shows much smaller deviations than the one performed for the data distribution since only the latter is impacted by the differences between data and Monte Carlo.
The first row of Figure 8.5 shows the unfolding result using a one-dimensional representation. It is visible that already after a single cINN training most of the distribution is already unfolded correctly. Nevertheless, in the high-p_T region starting at 220 GeV the iterative corrections are needed.
It is not sufficient that the one-dimensional projections are unfolded properly since this projection to one axis loses information about the true distribution in a multi-dimensional phase space. In this case it is therefore necessary to confirm that the two-dimensional representation of the truth-level data is reobtained during the unfolding. The two-dimensional unfolding result as well as the corresponding truth-level data are shown in the second row of Figure 8.5. In order to obtain a more quantitative result, the distributions

$$\frac{u}{r} - 1 \qquad \text{and} \qquad \frac{u}{t} - 1, \tag{8.6}$$

are calculated, with the unfolded distribution u, the detector-level data r and the truth-level data t all given as two-dimensional matrices. The results of this equations are visualised in the last row of Figure 8.5. These figures prove that the unfolded distribution is clearly closer to the truth-level data than the detector-level data, i.e. the algorithm is validated as a proper unfolding. This is especially visible in the high-p_T bins. To quantify the improvement, the explicit values are given in the figures.

Parameter	cINN	Classifier
Conditional coupling blocks	5	-
Layers (per block)	4	6
Units (per layer)	64	32
Epochs	100	100
Learning rate	10^{-4}	10^{-3}
Maximum learning rate	$3 \cdot 10^{-4}$	$3 \cdot 10^{-3}$
Weight decay	0.01	-
Batch size	4096	4096
Number of training events	1 500 000	1 500 000

Table 2: Hyperparameters of the cINN and the classifier used in the IcINN algorithm. Both use the ADAM optimizer as well as a one-cyclic learning rate. The conditional coupling blocks are only used in the cINN. It was made sure that the performance is invariant under small changes of these parameters.

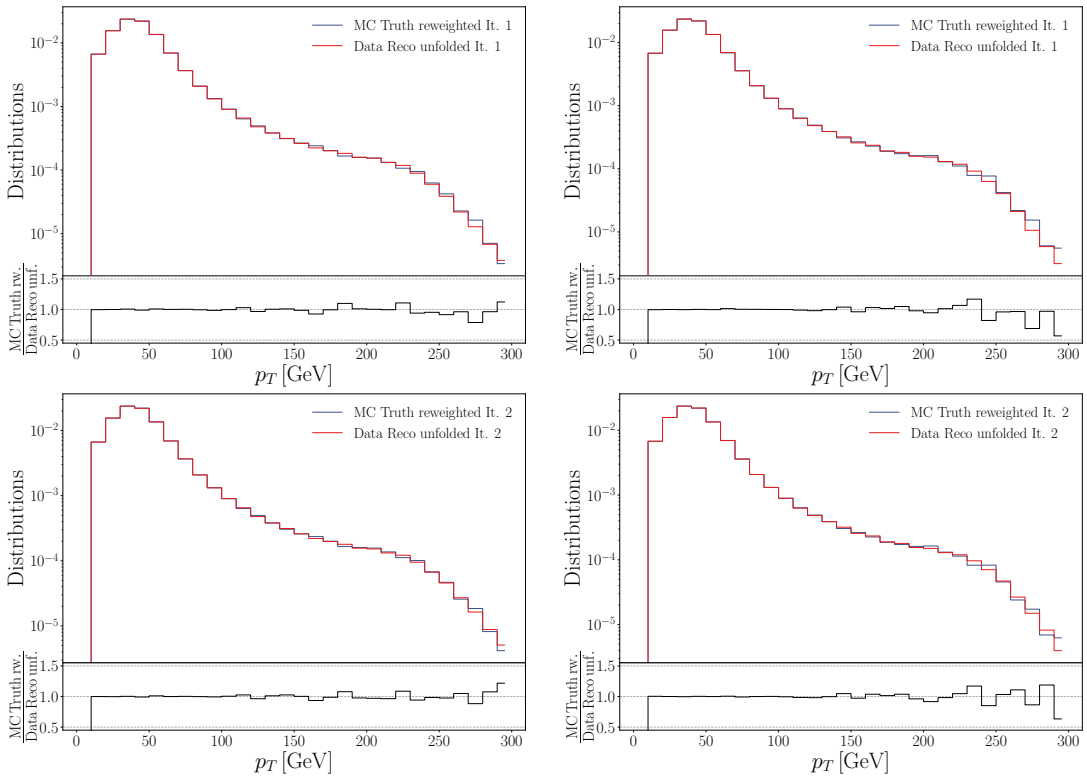


Figure 8.3: Closure check for the classifier performance. The reweighted truth-level Monte Carlo (blue) is compared to the unfolded distribution (red). The results are displayed for iteration 1/2 in the top/bottom row. In the left/right column the p_T distributions of the muon/anti-muon are shown. The distributions agree, hence demonstrating technical closure for the classifier.

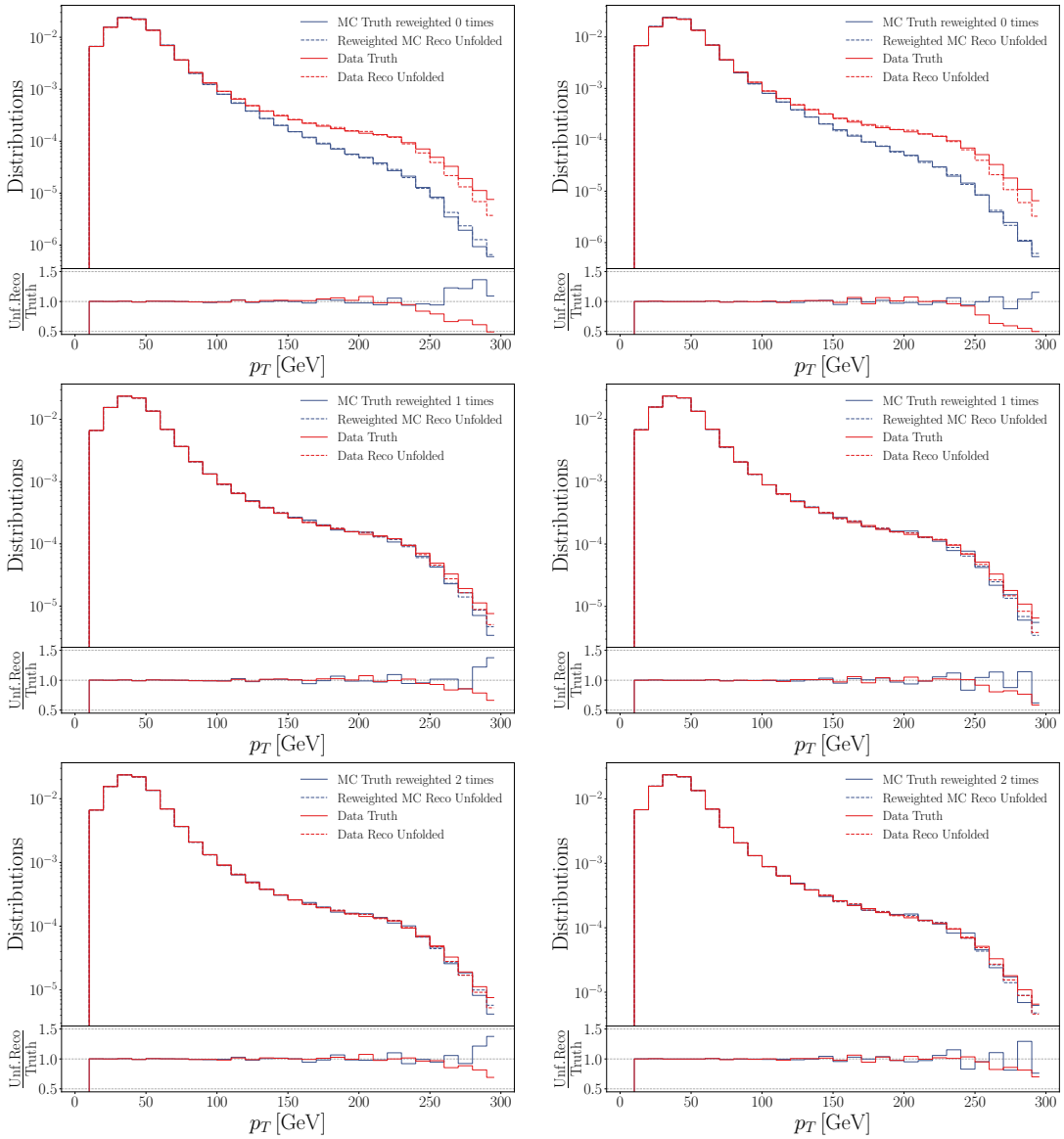


Figure 8.4: Closure test for the cINN in the IcINN algorithm. The cINN is applied to the (potentially weighted) detector-level Monte Carlo distribution and the result (dashed blue) is compared to the (potentially weighted) truth-level Monte Carlo distribution (solid blue). Since the cINN is trained on the Monte Carlo-simulation, these distributions should agree very well. The top/middle/bottom row show the first/second/third iteration. The left/right column shows p_T distribution of the muon/anti-muon. In addition, the truth-level data (solid red) as well as the unfolded detector-level data (dashed red) are shown. The continuous reduction of the bias towards the Monte Carlo simulation in the unfolded distribution is clearly visible in the high-p_T bins.

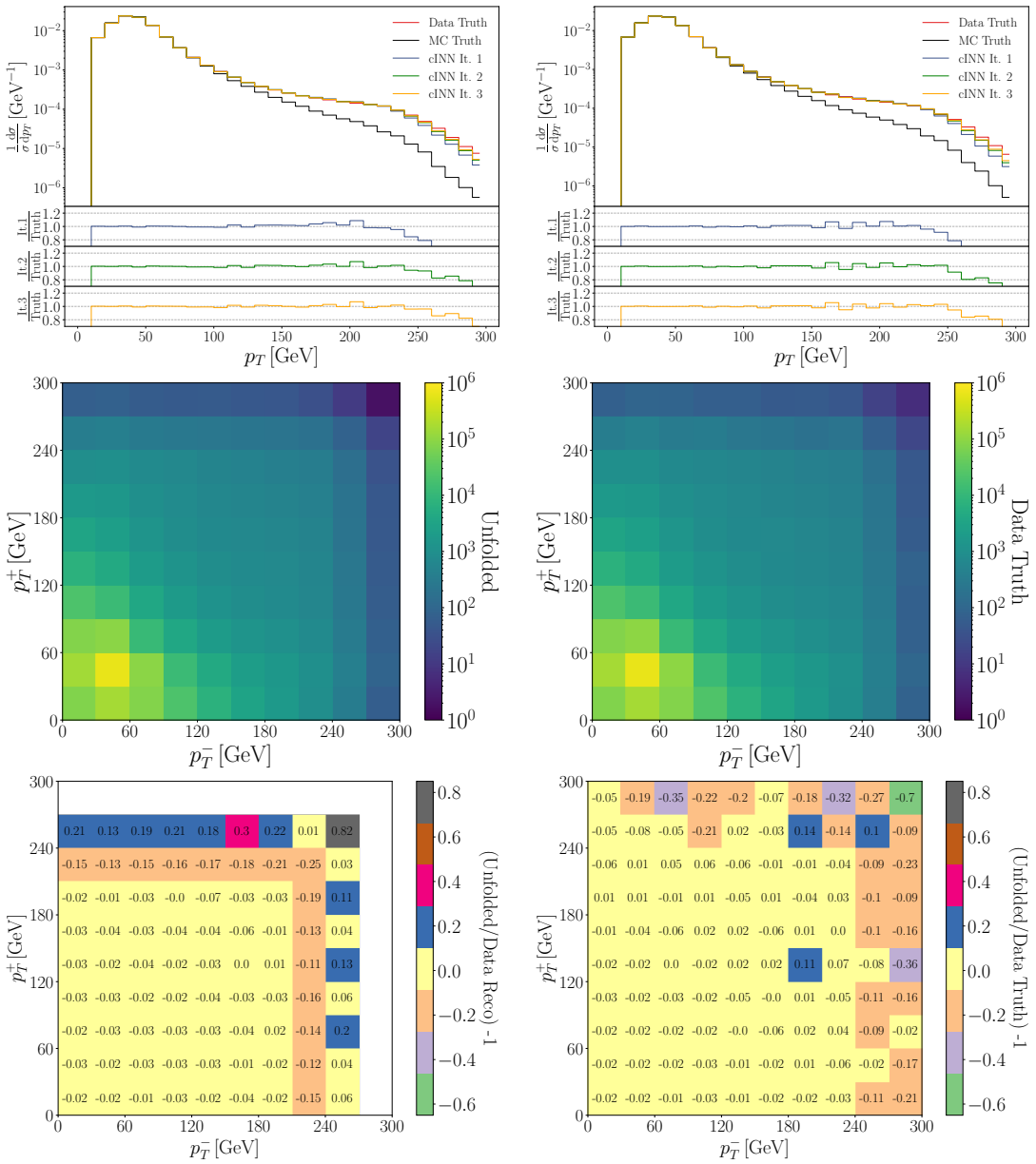


Figure 8.5: Unfolding result of the EFT-dataset. The first row shows the one-dimensional projection of the unfolded distributions after several iterations (blue, green, orange) as well as the truth-level data (red) and the truth-level Monte Carlo (black). A clear improvement in the high-p_T bins is observed. In the second row, the two-dimensional representation is shown for the unfolding result (left), as well as the truth-level data (right). For a quantitative comparison, the last row shows the matrix (unfolded distribution/detector-level data)-1 on the left and (unfolded distribution/truth-level data)-1 on the right. It is clearly visible that the unfolding result is closer to the truth-level data, i.e. there has been a real improvement during the unfolding process.

At this point it can be concluded that the *Iterative cINN unfolding* has been a success since it is clearly visible that a better unfolding result can be achieved if the classifier reweighting is used at least once. A more extensive discussion on how and when to us the IcINN unfolding is highly analysis dependent. This topic is covered in the conclusion of this thesis. Nevertheless, the proof of principle has been accomplished.

8.3. Comparison to IBU

One possibility to further validate the performance of the IcINN is a direct comparison to Iterative Bayesian Unfolding (IBU), which was introduced in Section 3. The application of IBU to the $Z\gamma\gamma$ data is straight-forward. The unfolding result is shown in Figure 8.6. For simplicity, only one of the two muon momenta is unfolded. Since the muon- and anti-muon-momentum are very similar, this will not be a problem. The unfolding result shows some structures which are very similar to the IcINN performance: the first iteration already predicts most of the truth-level data correctly. Nevertheless, further iterations can improve the result in the high-p_T region. It is again questionable if three iterations are needed or if two are sufficient.
At this point it is possible to compare the unfolded distributions of a single measured event. IBU derives the (one-dimensional) single event unfolded distributions as explained in Section 6. For illustrative purposes, a low-p_T event at 45 GeV as well as a high-p_t event at 185 GeV are chosen. Due to the binning, the IBU algorithm is restricted to unfold the full bin of a single event.
The IcINN unfolding is still two-dimensional, i.e. to obtain a single event unfolded distribution the (two-dimensional) event is fixed as the condition. The result after sampling

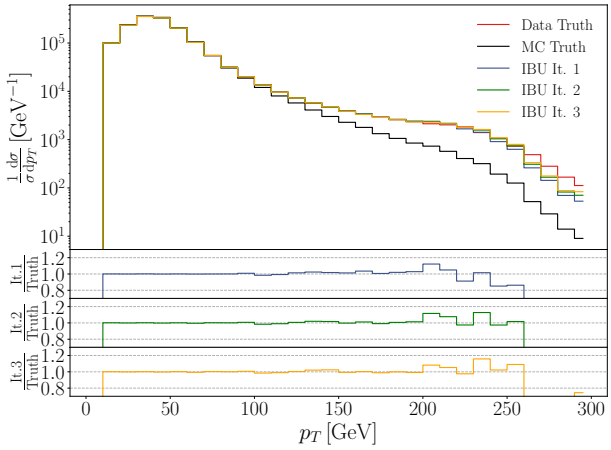


Figure 8.6: Unfolding result for Iterative Bayesian Unfolding. Most of the distribution is unfolded correctly after a single iteration (blue). The high-p_T bins show a similar behavior as for the IcINN unfolding, i.e. they require more than one iteration (green, orange). In addition, the figure shows the truth-level data (red) and the truth-level Monte Carlo (black).

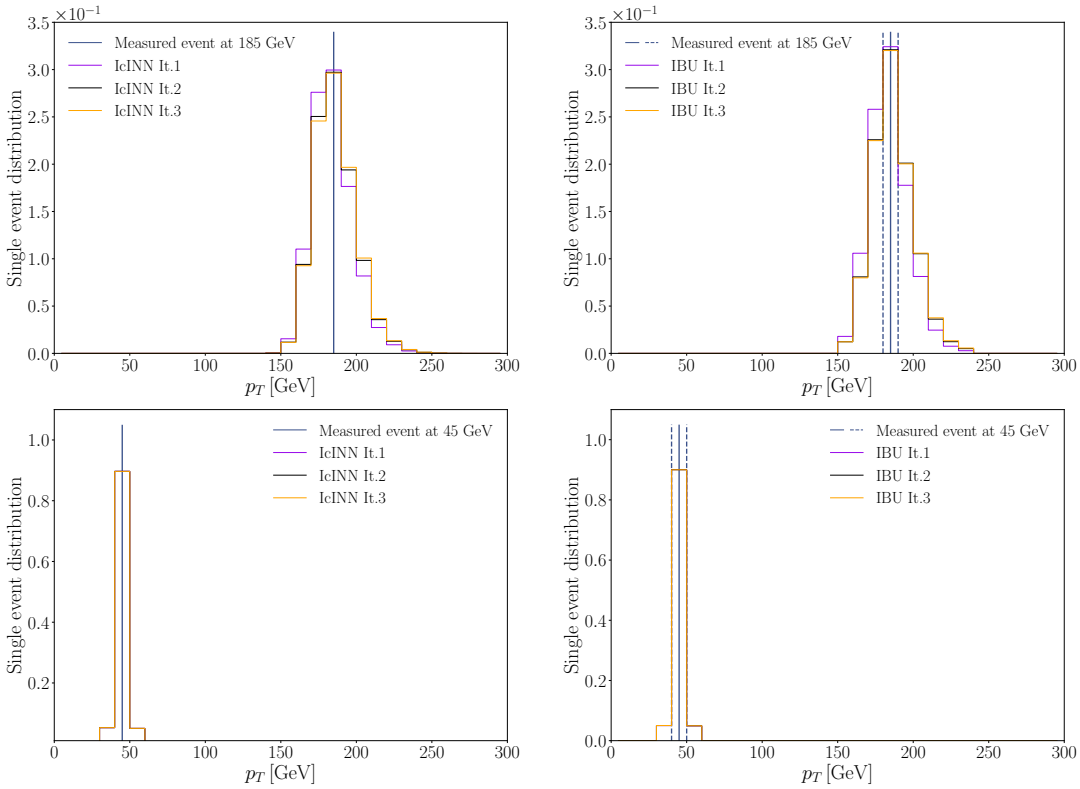


Figure 8.7: Single event unfolded distributions after several iterations (violet, black, orange). The top row shows the 1D-projection of the single event unfolded distribution for an event at 185 GeV. The bottom row shows the same for 45 GeV. The results for the IcINN are shown on the left; the results for IBU are shown on the right.

from the latent space is a two-dimensional unfolded distribution of the single event. In order to compare the single event unfolded distributions for IBU and IcINN, the two-dimensional IcINN distribution needs to be integrated out to a one dimension. In addition, it is also possible to compensate for the fact that the IBU unfolds the full bin of the event: the IcINN unfolds every event of the detector-level data which fulfills in the first dimension $p_T^- \in [40\,\text{GeV}, 50\,\text{GeV}]$ or $p_T^- \in [180\,\text{GeV}, 190\,\text{GeV}]$, with the p_T-value of the muon p_T^-. The two-dimensional unfolded distribution is then projected on the p_T^--axis. This result is the natural reduction to one dimension and can be compared to the single event unfolded distributions of IBU. The single event unfolded distributions are shown in Figure 8.7.

The high-p_T events look similar in the case of the IcINN and IBU. Most importantly, they show the same iterative behavior: bins below 185 GeV are reduced; bins above are increased in size with every further iteration. This similarity in behavior, as well as the overall similarity in the shapes shows a clear indication that the single event distributions predicted by the IcINN are consistent with the matrix-based unfolding algorithms. The

low-p_T single event unfolded distribution is shown in Figure 8.7 as well. It is expected that the single event unfolded distribution does not change significantly with more iterations since this part of the distribution is already properly unfolded after one iteration. The very narrow structure is expected because of the small detector response in this region. Again the IcINN and IBU distributions match.
This comparison between the IBU and IcINN single event unfolded distributions already shows that the single event unfolded distributions predicted by IcINN are in agreement with the matrix-based unfolding algorithms. To further compare the IcINN unfolding to the IBU unfolding, it is advantageous to implement a two-dimensional version of the IBU unfolding.

9. Conclusion and Outlook

The aim of this thesis was to investigate iterative unfolding in matrix-based algorithms as well as machine-learning-based algorithms. Throughout the project a fully Gaussian analytic toy model was used, which turned out to be a valuable benchmark for the performance of iterative unfolding algorithms.
A trained cINN is able to predict unfolded distributions even for a single measured event. In order to be able to validate these single event unfolded distributions of the cINN, the first part of the project was the implementation of a *single event unfolding* for matrix-based unfolding algorithms. This was achieved for Iterative Bayesian Unfolding (IBU) and Iterative Dynamic Stabilising (IDS). The implementation has been validated using the analytic toy model.
In addition to the matrix-based unfolding, a cINN unfolding was implemented and applied to the analytic toy model. The result of the cINN unfolding showed a strong model-dependency. This problem was solved by introducing the novel *Iterative cINN unfolding* (IcINN). With the IcINN is was possible to unfold the analytic toy model properly, reducing the bias to the prior assumptions iteratively. In addition to the unfolding result itself, the covariances of the final unfolded distributions were derived.

In the next step of the project the single event unfolded distributions for the matrix-based methods were compared to the single event predictions of the IcINN. In the case of the toy model, it was possible to calculate an analytic prediction at truth-level, both the IcINN as well as IBU and IDS were able to reproduce this analytic expectation. This validates the iterative approach of the IcINN even further.
Finally, the derived algorithms were applied to pseudo-data with a $pp \to Z\gamma\gamma$ final state. Applying IcINN and IBU to the full dataset yielded similar results for the full unfolded distribution. Additionally, the single event unfolded distributions matched as well. This is the first time that the single event unfolded distributions predicted by the IcINN have been validated using matrix-based unfolding algorithms.

This project provides a powerful unfolding tool: the IcINN. In a next step the IcINN can be applied to real experimental data. In addition, a direct comparison to other machine-learning-based unfolding algorithms like OmniFold is interesting. The main difference between the two algorithms is that the reweighting in the IcINN is applied on truth-level, not on detector-level. This can be advantageous while choosing the phase space of the Monte Carlo simulation. The construction of an example in which OmniFold breaks down is an interesting case for the IcINN to further investigate its performance. It is especially interesting, if an interpolation to regions of the phase space which are not covered by the Monte Carlo simulation is possible.
Finally, the question arises if and how to use the *iterated* version of the cINN unfolding. The application to the $Z\gamma\gamma$ dataset might give the impression that the training of a single cINN is sufficient in all cases, while the reweighting is a small correction in regions of low statistics. This is true at first glance, but the results for IBU showed a similar

behavior. There are analyses in which IBU is applied and multiple iterations are needed. In these cases the iterative improvement of the IcINN algorithm will become apparent as well, since throughout this project the performance of IcINN and IBU was always similar. Overall, the IcINN has strong potential for multiple applications in unfolding.

A. Appendix

A.1. Feynman vertices of the Higgs and the electroweak sector.

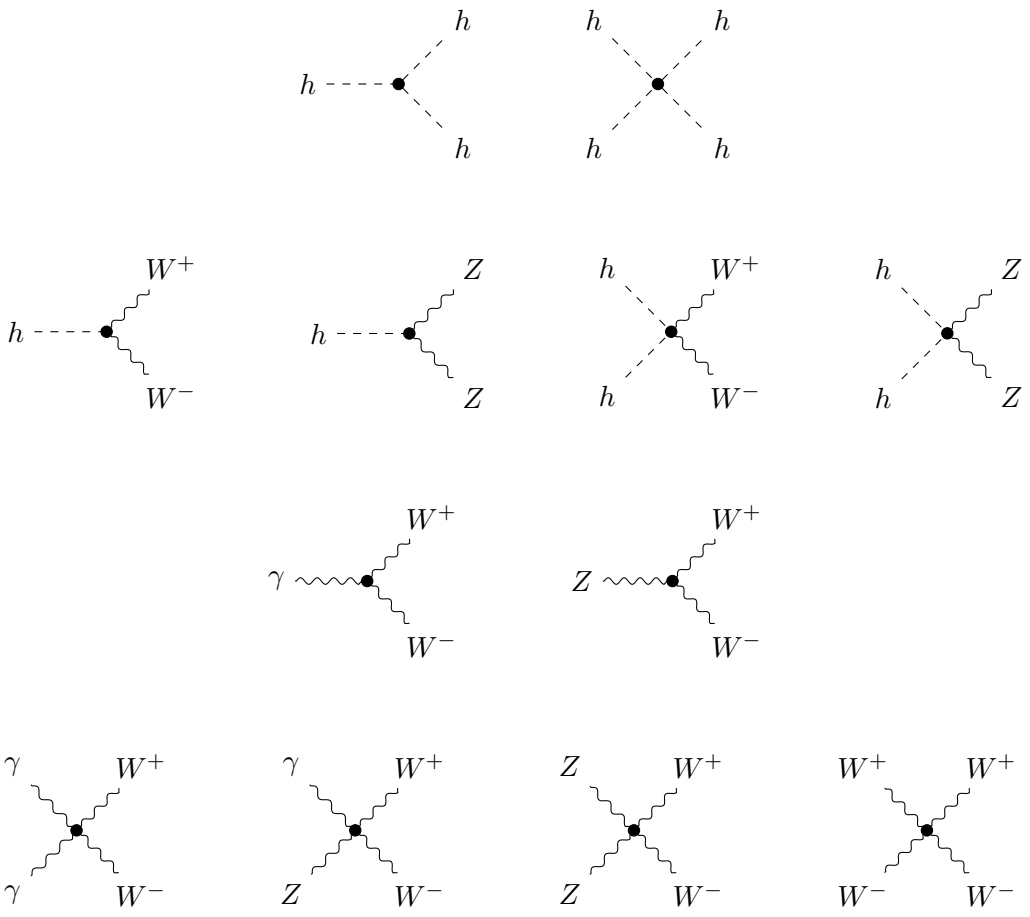


Figure A.1: Several interaction vertices of the Higgs and the electroweak sector of the Standard Model. In the first line there are the triple-Higgs and quartic-Higgs self-interactions. The second line shows the possible interactions between the Higgs and the massive gauge bosons of the electroweak sector. The third and fourth line show the possible interactions among the electroweak gauge bosons, including the massless photon. [20]

A.2. Gauge Boson Interactions with Fermions

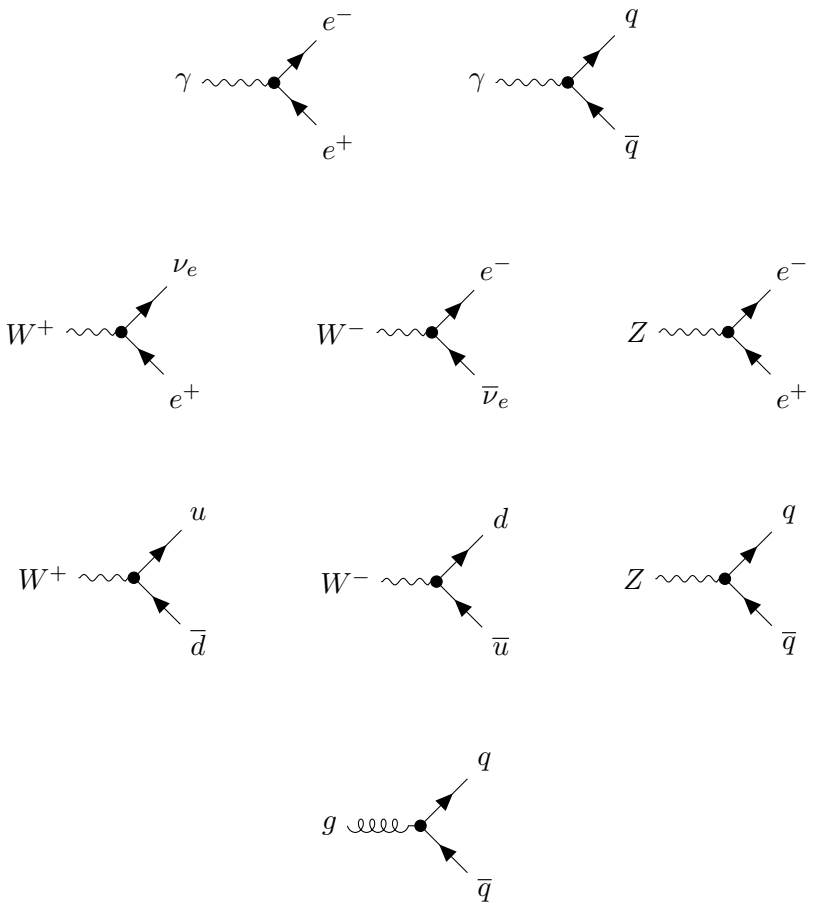


Figure A.2: Interactions of gauge bosons and fermions in the Standard Model of Particle Physics. In the first line there are interactions with the photon γ, in the second and third lines there are interactions with the weak bosons $W^{\pm}$ and Z, in the last line the interaction of the gluon g with quarks. Although only interactions with generation-1-particles are displayed, all these interactions are possible with their corresponding higher-generation particles as well. [20, 21]

A.3. Convergence of the Toy Model

The analytic expectation after one iteration is calculated using Bayes' theorem. In the purely Gaussian toy model the paramters of the resulting Gaussian distribution are

$$\mu_{t,1} = \frac{\mu_M \sigma_t^2 + \mu_t \sigma_s^2 - \mu_s \sigma_t^2}{\sigma_s^2 + \sigma_t^2}, \qquad \sigma_{t,1} = \frac{\sqrt{\sigma_t^2 \sigma_M^2 + \sigma_t^2 \sigma_s^2 + \sigma_s^4}}{\sigma_s^2 + \sigma_t^2} \sigma_t. \tag{A.1}$$

Further iterations can be performed by plugging in the result $\{\mu_{t,1}, \sigma_{t_1}\}$ instead of $\{\mu_t, \sigma_t\}$ in the equation above. The iterative mean values and the iterative standard deviations therefore define *recursive* sequences.
It is assumed that these sequences converge. Close to the limiting values, the recursive iteration should not have an impact anymore:

$$\mu_{t,\infty+1} = \mu_{t,\infty}, \qquad \sigma_{t,\infty+1} = \sigma_{t,\infty}. \tag{A.2}$$

From this, the limiting values can be derived. For the mean the equation

$$\mu_{t,\infty} = \frac{\mu_M \sigma_{t,\infty}^2 + \mu_{t,\infty} \sigma_s^2 - \mu_s \sigma_{t,\infty}^2}{\sigma_s^2 + \sigma_{t,\infty}^2}, \tag{A.3}$$

has to hold, which implies

$$\mu_{t,\infty} = \mu_M - \mu_s. \tag{A.4}$$

This is exactly the expected value since it matches μ_{Truth}. For the standard deviation the equation can be evaluated in the same way

$$\begin{aligned} & \sigma_{t,\infty} = \frac{\sqrt{\sigma_{t,\infty}^2 \sigma_M^2 + \sigma_{t,\infty}^2 \sigma_s^2 + \sigma_s^4}}{\sigma_s^2 + \sigma_{t,\infty}^2} \sigma_{t,\infty}. \\ \Rightarrow \quad & \sigma_{t,\infty}^2 \sigma_M^2 + \sigma_{t,\infty}^2 \sigma_s^2 + \sigma_s^4 = (\sigma_s^2 + \sigma_{t,\infty}^2)^2 \\ \Rightarrow \quad & \sigma_{t,\infty}^2 = \sigma_M^2 - \sigma_s^2. \end{aligned} \tag{A.5}$$

This matches the expectation as it coincides with the standard deviation of the truth distribution σ_{Truth}.

A.4. Closure Checks for the Toy IcINN Unfolding

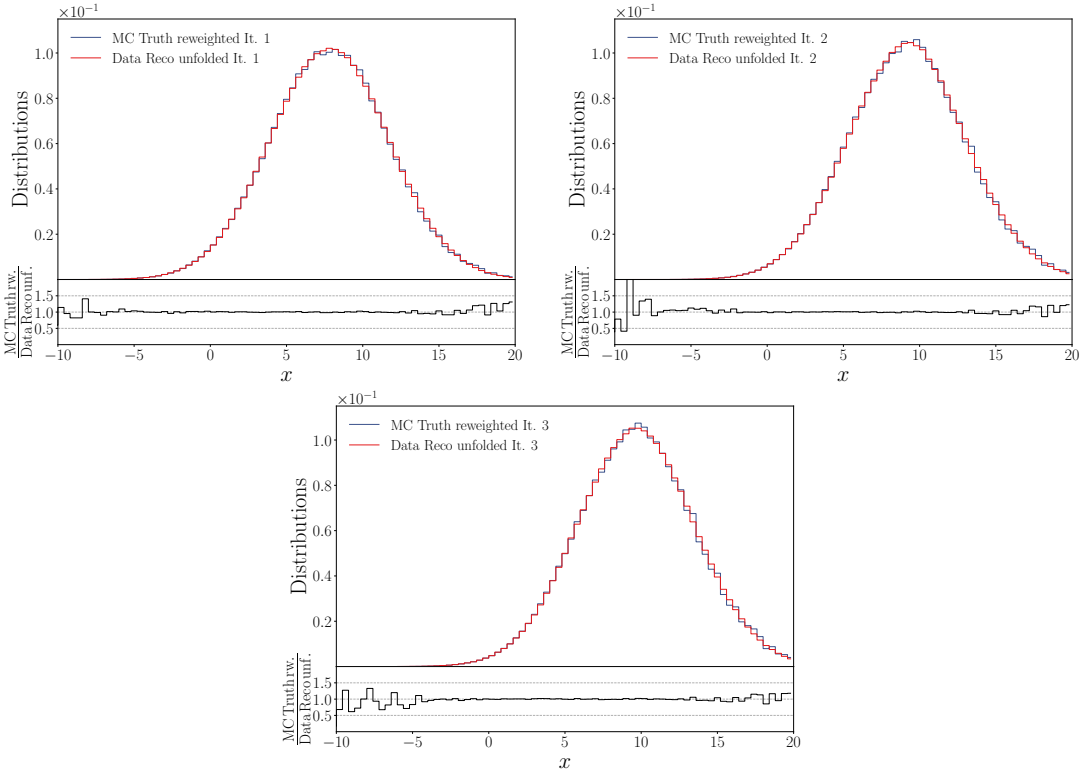


Figure A.3: Closure check for the classifier performance in the IcINN algorithm applied to the analytic toy model. The reweighted truth-level Monte Carlo (blue) is compared to the unfolded distribution of the current iteration (red). Displayed are the results for iteration one in the upper left, iteration two in the upper right and iteration three in the lower plot. The distributions match up and show that the classifier is working properly in the region of high statistics.

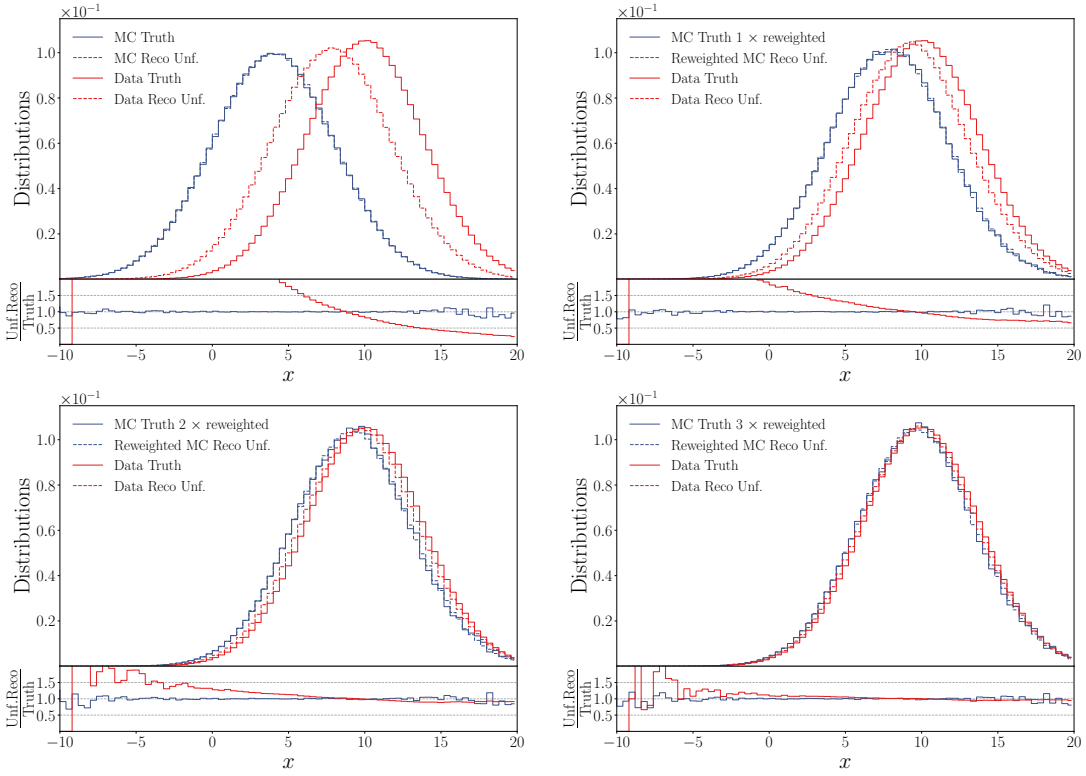


Figure A.4: Closure test for the cINN component of the IcINN algorithm applied to the analytic toy model. The cINN is applied to the (potentially weighted) detector-level Monte Carlo distribution and the result (dashed blue) is compared to the (potentially weighted) truth-level Monte Carlo distribution (solid blue). Since the cINN is trained on the Monte Carlo-simulation these distributions should agree very well. The upper left and upper right show these distributions for iterations one and two, the lower left and lower right for iterations three and four. The distributions always agree with each other in the region of high statistics. In addition, the truth-level data (solid red) as well as the unfolded detector-level data (dashed red) are shown. The closure check for the data component is shows larger deviations than the closure check for the Monte Carlo distributions. This is expected, because the former is impacted by differences between the data and the Monte Carlo simulation. The reduction of the bias towards the Monte Carlo simulation in the unfolded distribution is clearly visible throughout the iterations.

A.5. Applied Momentum Smearing for $Z\gamma\gamma$

The muon momentum smearing function used in the generation of the $Z\gamma\gamma$ data is

$$\Delta p_T = p_T \cdot \sqrt{0.025^2 + 3.5 \cdot 10^{-8} \cdot \left(\frac{p_T}{\text{GeV}}\right)^2}. \tag{A.6}$$

This function is plotted in Figure A.5.

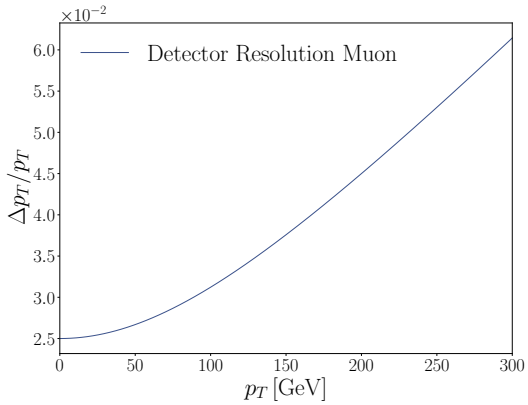


Figure A.5: Visualisation of the momentum smearing function for muons used in the data generation for the $Z\gamma\gamma$ data. The full ATLAS momentum smearing additionally contains a rapidity dependence.

References

[1] Glen Cowan. *Statistical data analysis*. Oxford university press, 1998.

[2] Andreas Hoecker and Vakhtang Kartvelishvili. Svd approach to data unfolding. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 372(3):469–481, 1996.

[3] Giulio D'Agostini. A multidimensional unfolding method based on bayes' theorem. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 362(2-3):487–498, 1995.

[4] Bogdan Malaescu. An iterative, dynamically stabilized method of data unfolding. *arXiv preprint arXiv:0907.3791*, 2009.

[5] Marco Bellagente, Anja Butter, Gregor Kasieczka, Tilman Plehn, Armand Rousselot, Ramon Winterhalder, Lynton Ardizzone, and Ullrich Köthe. Invertible networks or partons to detector and back again. *SciPost Physics*, 9(5):074, 2020.

[6] ATLAS Collaboration et al. Measurement of $z\gamma\gamma$ production in pp collisions at $\sqrt{s} = 13$ tev with the atlas detector. *arXiv preprint arXiv:2211.14171*, 2022.

[7] Tim Adye, R Claridge, K Tackmann, and F Wilson. Roounfold, 2005.

[8] Johan Alwall, R Frederix, S Frixione, V Hirschi, Fabio Maltoni, Olivier Mattelaer, H-S Shao, T Stelzer, P Torrielli, and M Zaro. The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations. *Journal of High Energy Physics*, 2014(7):1–157, 2014.

[9] Christian Bierlich, Smita Chakraborty, Nishita Desai, Leif Gellersen, Ilkka Helenius, Philip Ilten, Leif Lönnblad, Stephen Mrenna, Stefan Prestel, Christian Tobias Preuss, et al. A comprehensive guide to the physics and usage of pythia 8.3. *SciPost Physics Codebases*, page 008, 2022.

[10] J. de Favereau, , C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi. DELPHES 3: a modular framework for fast simulation of a generic collider experiment. *Journal of High Energy Physics*, 2014(2), feb 2014. doi: 10.1007/jhep02(2014)057.

[11] Michael E Peskin. *An introduction to quantum field theory*. CRC press, 2018.

[12] Otto Nachtmann. *Elementary particle physics: concepts and phenomena*. Springer Science & Business Media, 2012.

[13] Tilman Plehn. *Lectures on LHC physics*. Springer, 2012.

[14] H. David Politzer. Reliable perturbative results for strong interactions? *Phys. Rev. Lett.*, 30:1346–1349, Jun 1973. doi: 10.1103/PhysRevLett.30.1346. URL https://link.aps.org/doi/10.1103/PhysRevLett.30.1346.

[15] David J. Gross and Frank Wilczek. Ultraviolet behavior of non-abelian gauge theories. *Phys. Rev. Lett.*, 30:1343–1346, Jun 1973. doi: 10.1103/PhysRevLett.30.1343. URL https://link.aps.org/doi/10.1103/PhysRevLett.30.1343.

[16] Harald Fritzsch and Murray Gell-Mann. Current algebra: Quarks and what else? *arXiv preprint hep-ph/0208010*, 2002.

[17] S Weinberg. Weinberg 1967. *Phys. Rev. Lett*, 19:1264, 1967.

[18] Sheldon L. Glashow. Partial-symmetries of weak interactions. *Nuclear Physics*, 22(4):579–588, 1961. ISSN 0029-5582. doi: https://doi.org/10.1016/0029-5582(61)90469-2. URL https://www.sciencedirect.com/science/article/pii/0029558261904692.

[19] Abdus Salam. Weak and Electromagnetic Interactions. *Conf. Proc. C*, 680519: 367–377, 1968. doi: 10.1142/9789812795915_0034.

[20] Jorge Crispim Romão. Advanced quantum field theory (ist, 2019). *URL http://porthos. ist. utl. pt/ftp/textos/tca. pdf.*

[21] Mark Thomson. *Modern particle physics*. Cambridge University Press, 2013.

[22] Peter Schmüser. *Feynman-Graphen und Eichtheorien für Experimentalphysiker*. Springer, 1988.

[23] Ramon Winterhalder. How to gan lhc events.

[24] Eric Drexler. *Wikimedia: Elementary particle interactions in the Standard Model*. https://commons.wikimedia.org/w/index.php?curid=32230766 [Accessed: 2022-09-15].

[25] Peter W. Higgs. Broken symmetries and the masses of gauge bosons. *Phys. Rev. Lett.*, 13:508–509, Oct 1964. doi: 10.1103/PhysRevLett.13.508. URL https://link.aps.org/doi/10.1103/PhysRevLett.13.508.

[26] F. Englert and R. Brout. Broken symmetry and the mass of gauge vector mesons. *Phys. Rev. Lett.*, 13:321–323, Aug 1964. doi: 10.1103/PhysRevLett.13.321. URL https://link.aps.org/doi/10.1103/PhysRevLett.13.321.

[27] G. S. Guralnik, C. R. Hagen, and T. W. B. Kibble. Global conservation laws and massless particles. *Phys. Rev. Lett.*, 13:585–587, Nov 1964. doi: 10.1103/PhysRevLett.13.585. URL https://link.aps.org/doi/10.1103/PhysRevLett.13.585.

[28] Yuval Grossman. Introduction to flavour physics. In *LHC Phenomenology*, pages 35–80. Springer, 2015.

[29] Raymond Davis Jr, Don S Harmer, and Kenneth C Hoffman. Search for neutrinos from the sun. *Physical Review Letters*, 20(21):1205, 1968.

[30] Yoshiyuki Fukuda, T Hayakawa, E Ichihara, K Inoue, K Ishihara, Hirokazu Ishino, Y Itow, T Kajita, J Kameda, S Kasuga, et al. Evidence for oscillation of atmospheric neutrinos. *Physical review letters*, 81(8):1562, 1998.

[31] Ziro Maki, Masami Nakagawa, and Shoichi Sakata. Remarks on the unified model of elementary particles. *Progress of Theoretical Physics*, 28(5):870–880, 1962.

[32] BM Pontecorvo. Inverse ß-processes and non-conservation of lepton charge. Technical report, Joint Inst. for Nuclear Research, Moscow (USSR). Lab. of Nuclear Problems, 1957.

[33] Bruno Pontecorvo. Neutrino experiments and the problem of conservation of leptonic charge. *Sov. Phys. JETP*, 26(984-988):165, 1968.

[34] Jacobus Cornelius Kapteyn. 80. first attempt at a theory of the arrangement and motion of the sidereal system. In *A Source Book in Astronomy and Astrophysics, 1900–1975*, pages 542–549. Harvard University Press, 2013.

[35] Roberto D Peccei. The strong cp problem. In *CP violation*, pages 501–551. World Scientific, 1989.

[36] Georges Aad, Tatevik Abajyan, B Abbott, J Abdallah, S Abdel Khalek, Ahmed Ali Abdelalim, R Aben, B Abi, M Abolins, OS AbouZeid, et al. Observation of a new particle in the search for the standard model higgs boson with the atlas detector at the lhc. *Physics Letters B*, 716(1):1–29, 2012.

[37] Serguei Chatrchyan, Vardan Khachatryan, Albert M Sirunyan, Armen Tumasyan, Wolfgang Adam, Ernest Aguilo, Thomas Bergauer, M Dragicevic, J Erö, C Fabjan, et al. Observation of a new boson at a mass of 125 gev with the cms experiment at the lhc. *Physics Letters B*, 716(1):30–61, 2012.

[38] Julius Wess and Bruno Zumino. Supergauge transformations in four dimensions. *Nuclear Physics B*, 70(1):39–50, 1974.

[39] T Yanagida. Proc. workshop on unified theory and the baryon number in the universe. *KEK Report No. 79-18*, 95, 1979.

[40] Ilaria Brivio and Michael Trott. The standard model as an effective field theory. *Physics Reports*, 793:1–98, 2019.

[41] Aneesh V Manohar. Introduction to effective field theories. 2018.

[42] Enrico Fermi. Versuch einer theorie der β-strahlen. i. *Zeitschrift für Physik*, 88(3): 161–177, 1934.

[43] Andrew Kobach. Baryon number, lepton number, and operator dimension in the standard model. *Physics Letters B*, 758:455–457, 2016.

[44] Rodrigo Alonso, Elizabeth E Jenkins, Aneesh V Manohar, and Michael Trott. Renormalization group evolution of the standard model dimension six operators iii: gauge coupling dependence and phenomenology. *Journal of High Energy Physics*, 2014(4):1–47, 2014.

[45] Rodrigo Alonso, Hsi-Ming Chang, Elizabeth E Jenkins, Aneesh V Manohar, and Brian Shotwell. Renormalization group evolution of dimension-six baryon number violating operators. *Physics Letters B*, 734:302–307, 2014.

[46] Bohdan Grzadkowski, M Iskrzyński, Mikolaj Misiak, and Janusz Rosiek. Dimension-six terms in the standard model lagrangian. *Journal of High Energy Physics*, 2010(10):1–18, 2010.

[47] Christopher W Murphy. Dimension-8 operators in the standard model effective field theory. *Journal of High Energy Physics*, 2020(10):1–48, 2020.

[48] Philipp Ott. Private conversation.

[49] Oscar JP Éboli, MC Gonzalez-Garcia, and JK Mizukoshi. p p→ j j e±μ±ν ν and j j e±μ ∓ ν ν at o (α em 6) and o (α em 4 α s 2) for the study of the quartic electroweak gauge boson vertex at cern lhc. *Physical Review D*, 74(7):073005, 2006.

[50] Oscar JP Eboli, Ma Concepción Gonzalez-García, SM Lietti, and Sérgio Ferraz Novaes. Anomalous quartic gauge boson couplings at hadron colliders. *Physical Review D*, 63(7):075008, 2001.

[51] ATLAS collaboration et al. Observation of an excess of di-charmonium events in the four-muon final state with the atlas detector. Tech. rep., CERN, Geneva, 2022.

[52] A Augusto Alves Jr, LM Andrade Filho, AF Barbosa, I Bediaga, G Cernicchiaro, G Guerrer, HP Lima Jr, AA Machado, J Magnin, F Marujo, et al. The lhcb detector at the lhc. *Journal of instrumentation*, 3(08):S08005, 2008.

[53] Cian O'Luanaigh. Lhc progresses towards higher intensities. 2015.

[54] Esma Mobs. The cern accelerator complex-august 2018. Technical report, 2018.

[55] Lyndon Evans and Philip Bryant. Lhc machine. *Journal of instrumentation*, 3(08): S08001, 2008.

[56] Kenneth Aamodt, A Abrahantes Quintana, R Achenbach, S Acounis, D Adamová, C Adler, M Aggarwal, F Agnese, G Aglieri Rinella, Z Ahammed, et al. The alice experiment at the cern lhc. *Journal of Instrumentation*, 3(08):S08002, 2008.

[57] Roman Adolphi et al. The cms experiment at the cern lhc. *Jinst*, 803:S08004, 2008.

[58] Georges Aad, E Abat, Jasmin Abdallah, AA Abdelalim, Abdelmalek Abdesselam, BA Abi, M Abolins, H Abramowicz, E Acerbi, BS Acharya, et al. The atlas experiment at the cern large hadron collider. *Journal of instrumentation*, 3(S08003), 2008.

[59] HHJ Ten Kate. The atlas superconducting magnet system at the large hadron collider. *Physica C: Superconductivity*, 468(15-20):2137–2142, 2008.

[60] Akira Yamamoto, Y Makida, R Ruber, Y Doi, T Haruyama, F Haug, H Ten Kate, M Kawai, T Kondo, Y Kondo, et al. The atlas central solenoid. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 584(1):53–74, 2008.

[61] Eduardo Ros. Atlas inner detector. *Nuclear Physics B-Proceedings Supplements*, 120:235–238, 2003.

[62] John Neil Jackson, Atlas Sct Collaboration, et al. The atlas semiconductor tracker (sct). *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 541(1-2):89–95, 2005.

[63] E Abat, TN Addy, TPA Åkesson, J Alison, F Anghinolfi, E Arik, M Arik, G Atoian, B Auerbach, OK Baker, et al. The atlas trt barrel detector. *Journal of Instrumentation*, 3(02):P02014, 2008.

[64] Henric Wilkens, on behalf of the ATLAS LArg Collaboration, et al. The atlas liquid argon calorimeter: An overview. In *Journal of Physics: Conference Series*, volume 160, page 012043. IOP Publishing, 2009.

[65] Hermann Kolanoski and Norbert Wermes. *Teilchendetektoren*. Springer, 2016.

[66] W Bonivento. Overview of the atlas electromagnetic calorimeter. *Nuclear Physics B-Proceedings Supplements*, 78(1-3):176–181, 1999.

[67] A Henriques. The atlas tile calorimeter. In *2015 4th International Conference on Advancements in Nuclear Instrumentation Measurement Methods and their Applications (ANIMMA)*, pages 1–7. IEEE, 2015.

[68] L Pontecorvo. The atlas muon spectrometer. *The European Physical Journal C-Particles and Fields*, 34(1):s117–s128, 2004.

[69] Anke Ackermann. Measurement of the zzγ final state with the atlas detector at the lhc. Masterarbeit, Universität Heidelberg, 2022.

[70] W. Tung. Bjorken scaling. *Scholarpedia*, 2009. doi: 10.4249/scholarpedia.7412.

[71] Halina Abramowicz and Allen C Caldwell. Hera collider physics. *Reviews of Modern Physics*, 71(5):1275, 1999.

[72] Yuri L Dokshitzer. Calculation of the structure functions for deep inelastic scattering and e+ e-annihilation by perturbation theory in quantum chromodynamics. *Zh. Eksp. Teor. Fiz*, 73:1216, 1977.

[73] VN Gribov and LN Lipatov. Deep inelastic electron scattering in perturbation theory. *Physics Letters B*, 37(1):78–80, 1971.

[74] Guido Altarelli and Giorgio Parisi. Asymptotic freedom in parton language. *Nuclear Physics B*, 126(2):298–318, 1977.

[75] Alan D Martin. Proton structure, partons, qcd, dglap and beyond. *arXiv preprint arXiv:0802.0161*, 2008.

[76] Joao Pequenao. Computer generated image of the whole ATLAS detector. 2008. URL https://cds.cern.ch/record/1095924.

[77] Anders Andreassen, Patrick T Komiske, Eric M Metodiev, Benjamin Nachman, and Jesse Thaler. Omnifold: A method to simultaneously unfold all observables. *Physical review letters*, 124(18):182001, 2020.

[78] Marco Bellagente, Anja Butter, Gregor Kasieczka, Tilman Plehn, and Ramon Winterhalder. How to gan away detector effects. *SciPost Physics*, 8(4):070, 2020.

[79] Glen Cowan. A survey of unfolding methods for particle physics. In *Conf. Proc. C*, volume 203181, pages 248–257, 2002.

[80] Andrey Nikolayevich Tikhonov. On the stability of inverse problems. In *Dokl. Akad. Nauk SSSR*, volume 39, 1943.

[81] V Kartvelishvili. Unfolding with singular value decomposition. 2011.

[82] Giulio D'Agostini. Improved iterative bayesian unfolding. *arXiv preprint arXiv:1010.0632*, 2010.

[83] Bogdan Malaescu. An iterative, dynamically stabilized (ids) method of data unfolding. *arXiv preprint arXiv:1106.3107*, 2011.

[84] Bogdan Malaescu. Private conversation.

[85] Falk Bartels. Improving the sensitivity of dark matter searches by combining unfolded experimental signatures. Masterarbeit, Universität Heidelberg, December 2017.

[86] Thomas Malte Spieker. *Detector Corrected Search for Dark Matter in Monojet and Vector Boson Fusion Topologies with the ATLAS Detector*. PhD thesis, Universität Heidelberg, 2019.

[87] Bradley Efron. Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics*, pages 569–593. Springer, 1992.

[88] Niloy Purkait. *Hands-On Neural Networks with Keras: Design and create neural networks using deep learning and artificial intelligence principles*. Packt Publishing Ltd, 2019.

[89] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[90] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

[91] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378*, 2018.

[92] Sotiris B Kotsiantis, Ioannis Zaharakis, P Pintelas, et al. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160(1):3–24, 2007.

[93] Tilman Plehn, Anja Butter, Barry Dillon, and Claudius Krause. Modern machine learning for lhc physicists. *arXiv preprint arXiv:2211.01421*, 2022.

[94] Leslie N Smith. Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE, 2017.

[95] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[96] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

[97] Tijmen Tieleman, Geoffrey Hinton, et al. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[98] Steve Lawrence and C Lee Giles. Overfitting and neural networks: conjugate gradient and backpropagation. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, volume 1, pages 114–119. IEEE, 2000.

[99] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[100] Hwanjun Song, Minseok Kim, Dongmin Park, and Jae-Gil Lee. How does early stopping help generalization against label noise? *arXiv preprint arXiv:1911.08059*, 2019.

[101] Anders Andreassen and Benjamin Nachman. Neural networks for full phase-space reweighting and parameter tuning. *Physical Review D*, 101(9):091901, 2020.

[102] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979, 2020.

[103] Lynton Ardizzone, Jakob Kruse, Sebastian Wirkert, Daniel Rahner, Eric W Pellegrini, Ralf S Klessen, Lena Maier-Hein, Carsten Rother, and Ullrich Köthe. Analyzing inverse problems with invertible neural networks. *arXiv preprint arXiv:1808.04730*, 2018.

[104] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[105] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[106] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

[107] Sebastian Bieringer. Using invertible networks to measure qcd splittings from parton showers. Masterarbeit, Universität Heidelberg, 2021.

[108] Sebastian Bieringer, Anja Butter, Theo Heimel, Stefan Höche, Ullrich Köthe, Tilman Plehn, and Stefan T Radev. Measuring qcd splittings with invertible networks. *SciPost Physics*, 10(6):126, 2021.

[109] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[110] Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Guided image generation with conditional invertible neural networks. *arXiv preprint arXiv:1907.02392*, 2019.

[111] Ian Moult. Ml4theory. ML4Jets, 2022. URL https://indico.cern.ch/event/1159913/contributions/5065071/attachments/2538840/4370049/IanMoult_ML4Jets_2022.pdf.

[112] Diego M Hofman and Juan Maldacena. Conformal collider physics: Energy and charge correlations. *Journal of High Energy Physics*, 2008(05):012, 2008.

[113] Mathias Backes. How to unweight with gans, 2020.

[114] Mathias Backes, Anja Butter, Tilman Plehn, and Ramon Winterhalder. How to gan event unweighting. *SciPost Physics*, 10(4):089, 2021.

[115] Vladimir Andreev, M Arratia, A Baghdasaryan, A Baty, K Begzsuren, A Belousov, Arthur Bolz, V Boudry, G Brandt, D Britzger, et al. Measurement of lepton-jet correlation in deep-inelastic scattering with the h1 detector using machine learning for unfolding. *Physical review letters*, 128(13):132002, 2022.

[116] Bob Stienen and Rob Verheyen. Phase space sampling and inference from weighted events with autoregressive flows. *SciPost Physics*, 10(2):038, 2021.

Acknowledgements

At this point I want to thank all the people who made this project possible and made the last year one of the best throughout my studies.

First and foremost I want to thank the people who have been working with me on this project: Monica Dunford, Anja Butter and Bogdan Malaescu. Monica, thank you for giving me the opportunity to join KIP and be part of the ATLAS group. It was (and still is) great to get your input on the project, I am really looking forward to my PhD! Anja, it was a pleasure to work with you again after my bachelor thesis, thank you for getting this whole project started by recommending me to Monica. Apart from countless helpful discussions, you also initiated my trip to ML4Jets and believed in me when I needed to produce the final results a week before the conference. Bogdan, thank you for hundreds of discussions and explanations, I really learned a lot from you. I am really looking forward to future joint projects.

In addition, I want to thank the whole KIP F8/F11 group for the welcoming atmosphere and the countless helpful discussions. Your input was very valuable to this project. I especially want to thank Falk Bartels for helping me a lot in the beginning of this thesis, when I was struggling a lot with Root and RooUnfold. Furthermore, I want to thank Philipp Ott for his input on the $Z\gamma\gamma$ data and how to construct an EFT simulation with MadGraph. I really couldn't have done it without you. Finally I want to thank the group in general for their support over the year; you really made this a nice experience.
I want to thank Hans-Christian Schultz-Coulon in general, especially for sending me out to ATLAS-D in Siegen. Also I want to express my gratitude to Tilman Plehn, who gave me the opportunity to go to ML4Jets. Thank you, also to his whole group (and Ramon) for the nice time in the US.

Furthermore, I want to thank Philipp Ott, Anke Ackermann, Thomas Junkermann, Julia Kruse, Marie Mühlnickel, Jacqueline Lesch, Anna Backes, Falk Bartels, Lisa Baltes, Varsiha Sothilingam, Eric Jacob, Ramon Winterhalder, Martin Klassen, Markus Schmidt, Theo Heimel, Theresa Schock, Tigran Mkrtchyan, Constantin Nicolai and Sebastian Bieringer for proof-reading this thesis.

Finally, I want to thank my friends and family for their continuous support throughout my studies.

Erklärung

Ich versichere, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den 09.12.2022,

M. Backes

Mathias Backes