Alma Mater Studiorum · Università di Bologna

Scuola di Scienze
Dipartimento di Fisica e Astronomia
Corso di Laurea in Fisica

QUnfold a Quantum Annealing-based Unfolding_tool for HEP: a case study on entangled $t\bar{t}$ pair production at the ATLAS experiment

Relatore:
Prof. Matteo Franchini

Correlatori:

PhD. Simone Gasperini
PhD. Gianluca Bianco

Presentata da:
Valerio Brugnami

Anno Accademico 2023/2024

Abstract

In questi ultimi decenni la computazione quantistica è stato uno dei settori maggiormente in via di sviluppo, nonostante ad oggi pochi siano stati i risultati ottenuti per quanto riguarda la realizzazione di grandi computer quantistici in grado di risolvere problemi complessi. Ad oggi l'azienda D-Wave è la prima ed unica che permette agli utenti di interagire direttamente con un computer quantistico, così da rendere accessibile la computazione quantistica a chiunque voglia risolvere determinati problemi. É da questa possibilità che nasce l'idea di *QUnfold*, un software che si prefigge l'obiettivo di implementare una nuova tecnica di Unfolding (un problema standard della fisica delle particelle) utilizzando i vantaggi quantistici ottenuti grazie a D-Wave. Questa tesi si prefigge l'obiettivo di testare ed analizzare i limiti ed i punti di forza di questo approccio rispetto a quelli classici comunemente usati. Per fare ciò, sono stati utilizzati dati ottenuti dal decadimento di quark t, e su questi sono stati girati i vari algoritmi di Unfolding, così da poter comparare i risultati ottenuti.

Abstract

In recent years, quantum computing has emerged as a rapidly advancing field, though practical achievements remain limited due to significant technical challenges in constructing large-scale quantum computers. D-Wave currently stands as the only company offering a commercially accessible quantum annealer, enabling users to perform specific computations more efficiently. Leveraging this technology, we introduce *QUnfold*, a software designed to implement a novel unfolding technique (a key problem in particle physics) by harnessing the advantages of quantum computing. This work aims to evaluate the performance and limitations of this quantum-based approach compared to traditional methods. The evaluation is based on data from t quark decays, with various unfolding algorithms applied and the results compared.

Contents

Introduction

Quantum Mechanics, together with Einstein's Theory of Relativity, is the most revolutionary theory of the XX century. Despite it was developed over a century ago, at the moment we have just started to explore all of its possibilities and consequences.

Among all of the new fields born following quantum mechanics, Quantum Computation is the one that in recent years has evolved rapidly, offering computational advantages that are still being uncovered. The concept of a quantum computer was first proposed by Richard Feynman in 1982, when he observed that there seemed to be essential difficulties in simulating quantum mechanical systems on classical computers. He suggested that building computers based on the principles of quantum mechanics would allow us to avoid those problems. However, building such computers up to now has been a difficult challenge and today we have not still managed to fully exceed such problems. In fact at the moment the only few quantum computers can be used freely by the user is the one offered by D-Wave's company. Despite this clearly is a very important achievement, it has to be understood that this does not represent Universal Computation, which means that not every kind of problem can be solved by the quantum computer current available. In particular this computer is called a Quantum Annealer and it is able to solve different kind of optimization problems. We will see that the only form of mathematical problems that can be implemented is called the QUBO form. On the other hand Universal Computers are being developed, as they represent the real goal of quantum computation, but nowadays they are apparently not as powerful as D-Wave's quantum annealer.

In this thesis we will analyse the advantages given by this quantum computer in the High Energy Physics field, in particular on the solution of the Unfolding problem. The Unfolding is a very common procedure in particle physics, as it deals with the distortions that arises when carrying out a measurement due to all the detector's effects. Therefore what we measure can be distorted with respect to the the real phenomena, implying we cannot compare the observed data with the theoretical ones. The unfolding technique enables us to correct this distortions and to obtain the "real" data. The starting point of this thesis is realizing that it is possible to see the unfolding as a quantum annealing problem. This means that once having done this we could theoretically solve the unfolding problem through D-Wave's quantum annealer. Unfortunately, this is not as easy as it seams. There are a lot of problems and challenges in trying to implement an algorithm able to solve this kind of problem.

QUnfold is a Python package that implements this new method of solving the unfolding. This package enables the user to directly send his unfolding problem to D-Wave and to solve it via quantum annealing. However, this is still a work in progress project, and the aim of this thesis is to test it analyzing its strengths and its critical points. To do so we have taken data from an high energy physics experiment about quantum properties of particles. In this work they present a great quantity of data and they need to unfold them. We have therefore taken this data and have tested *QUnfold* methods comparing them to the classical ones.

To sum up, this work is divided in four main chapters:

- **Chapter 1**: here Quantum Annealing is presented throughout its mathematical description and how it has been implemented in D-Wave's quantum computers.
- **Chapter 2**: the Unfolding is described and some classical algorithms are presented. In order to better understand this procedure, at the end of the chapter an example is provided.
- **Chapter 3**: in order to be aware of the analysis used to test the unfolding via quantum annealing, here is presented an introduction to the HEP detector (in particular ATLAS) and to the Standard Model of particle physics, with a Section concerning the process on which we are studying the problem.
- **Chapter 4**: this is the core of this work. It explains how the analysis has been carried out presenting its results.

Chapter 1

Quantum Annealing

The aim of this first chapter is to present the central problem of this thesis: Quantum Annealing (QA). This technique has been studied throughout these last years, as it is one of the first quantum computational problems we are able today to solve very efficiently. This chapter is divided in two main Section. In the first we present a mathematical description of the problem, analyzing the theorems that gave birth to quantum annealing and all the mathematics behind it. In Section 1.2 we present the first quantum annealer (i.e. a computer able to perform quantum annealing) of the world, that is the one provided by D-Wave system. We give a brief description of the Annealer and then we focus our attention in understanding the three main annealing method provided by D-Wave.

1.1 Mathematical description

In the last decades, and in particular these last years, quantum computation has spread across the industry with a lot of possibility of technological applications. In fact, following the idea of R. Feynman proposed back in the 1980s [1], quantum computers are believed to being able to solve certain problems more efficiently than classical ones.

Today there are two different kind of quantum computers: gate model and quantum annealing. The first one implements the algorithm with quantum gates, and are studied and implemented by IBM and PASQUAL. It represents the so called universal computation, which means that these kind of quantum computers can ideally solve any n-qubit operation. At the end of the process the system is measured and collapses in a classical state. Quantum annealers work in a total different way by taking advantage of the Schrodinger equation evolution. The idea was firstly introduced by Kadowasi and Nishimori in 1998 [2]. They are based on the *quantum adiabatic theorem* (better explained in Sec. 1.1.1) which states that if the equation parameters evolve slowly enough over time, the system will remain in the ground state. In contrast of the quantum gate based computers, quantum annealers are very efficient in solving optimization problems, but they are not universal, which means that there are some problems they cannot solve.

1.1.1 Quantum Adiabatic Theorem

The quantum adiabatic theorem, proposed by Max Born and Vladimir Fock in 1928, states that:
A physical system remains in its instantaneous eigenstate if a given perturbation is acting on it slowly enough and if there is a gap between the eigenvalue and the rest of the Hamiltonian's spectrum[3].

Its implication in quantum mechanics are quite evident. Suppose we have a quantum mechanical system H starting at time t_0, prepared in its ground state denoted by $|\psi_0(t_0)\rangle$. Suppose then that the spectrum is discrete, so that there is always a gap between the eigenvalues. If we evolve the system under a slow perturbation according to the theorem it will remain in the ground state. In particular, at $t_1 >> t_0$, the system will be in the ground state of the modified Hamiltonian, denoted by $\langle\psi_f(t_1)|$. Basically Quantum Annealing uses this property to find the final state of the system which, as we will see in Section 1.1.2, describes the optimal solution of the problem we want to solve.

To have an idea of what it means "slowly enough", the condition our evolution must respect is [4]

$$\frac{max[\langle\psi_f(t)| \frac{dH(t)}{dt} |\psi_0(t)\rangle]}{min[E_f(t) - E_0(t)]} \ll 1 \tag{1.1}$$

In this condition, E_f and E_0 are the eigenstates corresponding respectively to ψ_f and ψ_0, and the denominator is the so called minimum gap, which is the minimum difference throughout time of the two eigenstates. Is it clear now why we need to change the Hamiltonian as slowly as we can, so that we can decrease its derivative over time and respect the condition.

By these, we can finally introduce a brief description of adiabatic quantum computation, which is the technique quantum annealing uses. In optimization problems, one has an Hamiltonian where its ground state represents the solution. More specifically, this is often referred to as the *final* Hamiltonian. A simpler Hamiltonian, known as the *initial* Hamiltonian, is prepared on a quantum system with its initial configuration being the ground state. The quantum system adiabatically evolves this Hamiltonian towards the configuration of the final Hamiltonian. Following the theorem, at the end the system will be in the modified ground state, corresponding to the solution to the initial problem.

1.1.2 Ising and QUBO formulation

Despite in adiabatic quantum computation there are a lot of possible Hamiltonians that can be used, the Ising Hamiltonian is the most common Hamiltonian supported and studied in this field. The **Ising model** is a mathematical representation commonly used in statistical mechanics. It is based on variables that describe moments of magnetic dipoles called "spin up"($|\uparrow\rangle$) and "spin down"($|\downarrow\rangle$), corresponding respectively to $+1$ and -1 values. In the spin basis we therefore have this computational basis:

$$|\uparrow\rangle = \begin{pmatrix}1\\0\end{pmatrix} \equiv |0\rangle \qquad |\downarrow\rangle = \begin{pmatrix}0\\1\end{pmatrix} \equiv |1\rangle \tag{1.2}$$

According to the Ising model, the possible relationship between the spins, which are represented by *couplings*, are clearly only correlation or anti-correlation. We can then write the function expressed by the Ising model as:

$$H_{ising} = \sum_{i=1}^{N} h_i s_i + \sum_{i<j} J_{ij} s_i s_j \tag{1.3}$$

where s_i are the spin of of the i-th qubit (having therefore $s_i = \pm 1$), h_i are the *biases* of each qubit (i.e. the magnetic field applied on the qubit) and J_{ij} represent the couplings. We now need to understand why this kind of Hamiltonian is very useful in Quantum Annealing.

In order to translate the input problem we want to solve in a form that the quantum computer is able to read, we need the so-called **QUBO form** (Quadratic Unconstrained Binary Optimization form). That is the only type of problem a quantum annealer can solve. In this QUBO formulation of the problem, the purpose is to minimize a quadratic function of binary variables (i.e. only 0 for False and 1 for True), that can be expressed as:

$$f(\mathbf{x}) = \mathbf{x}^T Q \mathbf{x} \tag{1.4}$$

where Q is a $N \times N$ square matrix which defines the function, and $\mathbf{x}$ is a vector of binary variables. We can therefore express:

$$f(\mathbf{x}) = \sum_{i=1}^{N} Q_{ii} x_i x_i + \sum_{i<j} Q_{ij} x_i x_j \tag{1.5}$$

We have in this way separated the diagonal term with the off-diagonal terms. It is important to underline that QUBO problems are not constrained, meaning that there are no constraints on the variables other than those expressed in Q.

It is slightly evident that there is a similarity between these two models, in particular between eq. 1.3 and 1.5. It can be in fact shown that these two formulations are equivalent, meaning we can translate one into the other [5].

To sum up what we have discussed so far, we can say that if we are able to formulate our problem through an Hamiltonian which follows the Ising model, we are sure that we can translate it into the QUBO form, meaning we can solve the starting problem via

Quantum Annealing. Despite that, as we will see later in Section 4.1.1, this is one of the main difficulties faced when trying to solve optimization problems. In fact, as it seems quite immediate the correlation between the Ising model and the QUBO one, doing it in practice may create some issues.

1.1.3 Quantum Annealing process

Now that we have seen the main features of Quantum Annealing, it is time to understand how it effectively operates.

Following what we have said before, we consider an Ising model with its Hamiltonian H_P, where P denotes the "problem Hamiltonian". The aim of annealing is to find the ground state of H_P, which expresses the solution of the starting problem. Following the Ising model presented in Section 1.1.2, each element can be in two different states called *spin up* and *spin down*.

We assume that H_P is a classical many-body Ising Hamiltonian, that can be described in terms of the z components of the Pauli operator σ_j^z. In particular, in quantum annealing we have

$$H_P = \sum_i h_i \sigma_i^z + \sum_{i<j} J_{ij} \sigma_i^z \sigma_j^z \tag{1.6}$$

where σ_i^z are Pauli-z matrices associated with the i-th spin, h_i represents the interaction between the local magnetic field and the i-th spin and J_{ij} defines the strength of the interactions between the two spins. This Hamiltonian clearly follows the one in eq. 1.3.

We now introduce a driver Hamiltonian H_D which does not commute with H_P and has the trivial ground state. This represents the starting hamiltionian of the process. Usually a common choice for H_D is the transverse field $H_D = -\sum_j \sigma_j^x$, so that the non commutativity is satisfied. We can define the total Hamiltonian as:

$$H(t) = A(t)H_D + B(t)H_P \tag{1.7}$$

$A(t)$ and $B(t)$ are some time-dependent functions satisfying at the initial time $A(t_i) \gg B(t_i)$ and at the final time $A(t_f) \ll B(t_f)$. By substituting H_P and H_D we can obtain:

$$H(t) = -\frac{A(t)}{2}(\sum_i \sigma_i^x) + \frac{B(t)}{2}(\sum_i h_i \sigma_i^z + \sum_{i<j} J_{ij} \sigma_i^z \sigma_j^z) \tag{1.8}$$

At t_i we only have the driver Hamiltonian H_D.The initial state is set at the ground state of H_D, which is known a priori. In fact, the lowest energy state of this term is when all the qubits are in a superposition of the states $|0\rangle$ and $|1\rangle$; in particular as we will see in Section 1.2 in a D-Wave quantum computer we will always have that the starting situation is with all the qubits in the $|+\rangle$ state, which corresponds to:

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

Instead, at t_f we will have just the problem Hamiltonian H_P. This term is the Hamiltonian that contains our problem, encoded through the values of the biases h_i and the coupling strengths J_{ij}. The lowest energy state of this term is the answer to the problem we are trying to solve. Note that the final state of our system is always a classical state and depending in which classical state the system ends we have a different solution for our problem.

The idea of QA is that according to the quantum adiabatic theorem if $H(t)$ changes "sufficiently" slowly with t, the spin state evolves adiabatically and it remains in the ground state of the instantaneous Hamiltonian, passing form the ground state of H_D to the one of H_P. In this way at the end of the process it will arrive at the ground state of H_P, giving us the result we were seeking.

Unfortunately in practice it is impossible to have a perfect adiabatic evolution, which means that the final result may be slightly different from the ground state. This implies that it is very important in QA to repeat the process several times.

1.2 D-Wave samplers

A Quantum Annealer is a quantum computer designed to solve optimization problems through QA. As explained before, it operates through the slow evolution of the Schrodinger equation of the quantum system.

The D-Wave system, founded in 1999, is a pioneering company in quantum computing, focusing on developing efficient techniques to realize Quantum Annealing [6]. D-Wave system implements this technique using a single quantum algorithm, being able to realize QPU (quantum processing units) with more than 5000 physical qubits (quantum bits), much more than the ones that gate-based quantum computers have reached so far.

As said before, in order to satisfy the conditions of the adiabatic theorem, one of the key points in D-Wave's QPU is that it must be kept in an environment isolated from the outside, with a temperature below 20°mK. This is because it is the only way to ensure that the system behaves quantum mechanically. In Fig. 1.1 we can see the D-Wave quantum computer.

Another important aspect of D-Wave is that users interact directly with the D-Wave quantum computer through a web user interface (UI), and through open-source tools that communicate with the Solver API (SAPI). In this way users are able to submit their problems to D-Wave, having three different solving methods:

- **Simulated Annealing** : that is the classsical counterpart of Quantum Annealing, and so does not uese D-Wave's QPU.
- **Hybrid solver** : it is an annealing methods that divides the problem in two subproblem. One will be solved by the QPU and the other one by a classical CPU.

Figure 1.1: This picture shows the D-Wave Quantum Computer

- **Quantum Annealing** : that is the method we have discussed so far. The main problem with this approach is that at this moment total quantum machines are not able to solve very complex problems.

1.2.1 Simulated Annealing

Simulated Annealing (SA) is the classical counterpart of Quantum Annealing. It was firstly introduced by Kirkpatrick et al [7] to solve optimization problems and now it has become a very used and studied technique. In mathematical optimization, simulated annealing is used to find an approximate global minimum of a function E(x). It is therefore evident the first main difference with quantum annealing. In the latter, indeed, the optimal solution represented by the ground state is precise, as it is an eigenstate of the Hamiltonian. In SA we can obtain only an approximation of that, which can be more or less accurate. It becomes more efficient if it is used in a solution space that has discrete solutions, so that these intrinsic uncertainties are reduced.

Therefore, as for QA, our goal is to find the minimum of a given function expressing the problem, which plays the role of the energy of the system. SA algorithm starts with an initial solution x_0 and a starting temperature of T_0. At each iteration, the simulated annealing considers some neighboring state x' of the current state x, and evaluates the $\Delta E = E(x') - E(x)$. If $\Delta E \leq 0$ the new solution is accepted, because it means that the new state has a minor energy. Otherwise, if $\Delta E > 0$, x' is accepted only with the probability of

$$P(x, x', T) = \exp\left\{-\frac{\Delta E}{T}\right\} \tag{1.9}$$

where T is the current temperature. This probability is necessary to escape from a local minimum by accepting worst solutions. In fact otherwise the algorithm would get stuck in a local minimum being enable to get out. Obviously sometimes this problem persists, as for high energy barrier it is still difficult to escape from a non global minimum.

Another important aspect of SA, is that at each iteration the temperature is reduced. According to this probability starting from high temperatures means accepting all the solutions, as the exponential will be big. In this way at the beginning of the process, the system has the same probability to stay in any possible solution state. At each step, T is therefore reduced following the *annealing schedule*. The choice of the cooling schedule is critical for the performance of the algorithm. As it turns out, under certain conditions it can be shown that the algorithm converges to the global minimum with probability 1 as the number of iterations approaches infinity, provided the cooling schedule is sufficiently slow [8].

To compare this method to the quantum annealing, we can identify from eq. 1.9 the escape rate of SA being $e^{-\frac{h}{k_B T}}$, with k_B and T respectively the Boltzmann constant and the temperature, while h is the height of the barrier. We assume that h is proportional to the system size N, which means that to reach the global minimum we need an exponentially long time in N. $(h \sim O(N))$

The advantage given by QA is that according to Quantum Theory the tunneling probability is $e^{-\frac{\sqrt{h}w}{g}}$, [9] whit w the width of the barrier and g the strength of quantum fluctuations.

Consequently, as we know that $h \sim O(N)$ and $w \sim O(N^{\frac{1}{2}})$, the time necessary to escape from a local minimum is subexponential in N. This is schematically shown in Fig. 1.2. Several numerical and experimental studies have provided evidences for such an advantage of QA over SA in some specific models [4].

It is important to underline the fact that despite QA advantages, SA nowadays is much more useful. That is because though D-Wave quantum annealers are improving both in the number of qubits and in topology [1], they are still not able to solve complex problems. SA is then a very useful tool to solve them. That is why D-Wave gives the user the possibility to also run Simulated Annealing.

[1]The topology of a QPU is the scheme by which the qubits are realted to each other. Currently D-Wave is working at the Zephyr topology, but we are still very far to being able to have all qubits connected to each other. [6]

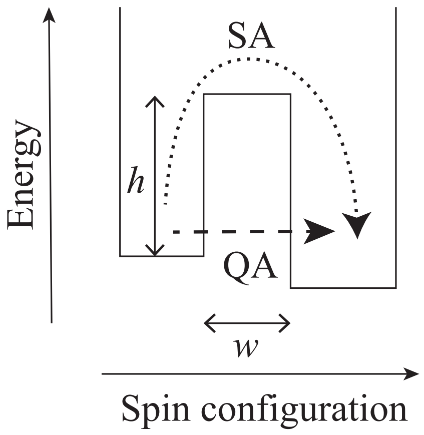


Figure 1.2: Schematic picture of the thermal fluctuation and quantum tunneling in a system with local energy minima separated by an energy barrier with the height h and the width w.

1.2.2 Hybrid sampler

Due to the problems expressed in the last part of Section 1.2.1, an immediate way to overtake them is to use both QA and SA simultaneously. The hybrid solver of D-Wave Systems integrates classical and quantum annealing techniques to solve very complex problems. In this way the advantage of quantum annealing, provided by D-Wave's quantum processing unit (QPU), and of classical algorithms are combined to find very precise solutions. The hybrid solver can therefore studies problems that purely quantum or simulated annealing fail to solve [10].

D-Wave's Hybrid Solver divides therefore the problem in many parts. The QPU analyze the parts of the problem where quantum annealing can give a great advantage, while the CPU studies the rest. This dynamic allocation is managed by a complex system that decides which parts of the problem are best solved by QA and which ones are best for SA. In this way, the solver can resolve larger and more complex problems than a purely quantum or classical approach could solve. It also improves the quality of the solution, as the classical annealing can refine the results obtained from the quantum annealer.

To sum up, D-Wave's hybrid solver uses the advantages given by quantum computing and combined them with the possibilty of solve big problems given by classical algorithms. In this way it is able to give superior results in solving real-world optimization problems.

1.2.3 Quantum Annealer

We have already discussed about quantum annealing techniques and how they works. But how it is implemented in D-Wave system?
The lowest energy states of the superconducting loops that form the D-Wave QPU are the qubits. They have a circulating current with its corresponding magnetic field, that can be $|\uparrow\rangle = |1\rangle$ or $|\downarrow\rangle = |0\rangle$ or any superposition of these two. As explained previously, at the end of the quantum annealing process each qubit collapses from a superposition state into a classical state, that can be either $|0\rangle$ or $|1\rangle$.

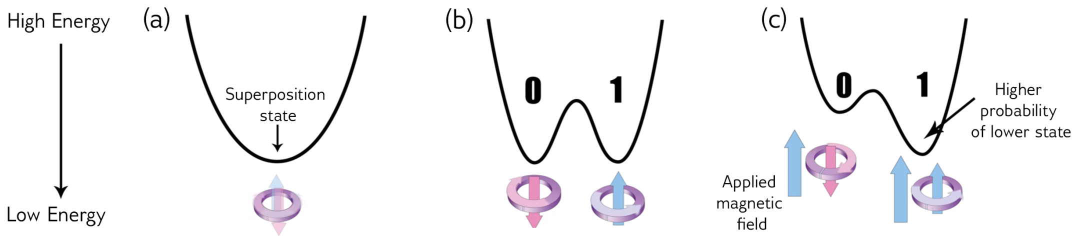


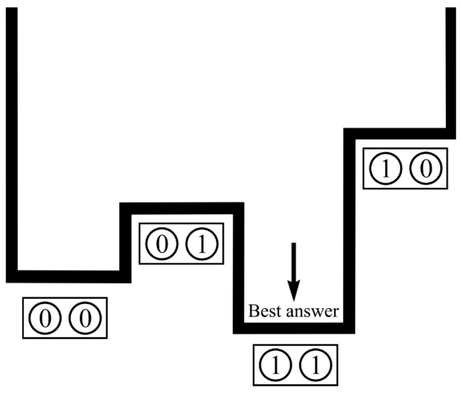
Figure 1.3: This image shows the three main phases of a quantum annealing process. Firstly, the qubit is in superposition, then the two states are separated by an energetic barrier. In the end, this barrier is raised and shaped using biases.

The physics of the process described formally in Section 1.1, is shown better in Fig. 1.3. In this example at the beginning the system is prepared in a superposition state (a), so that the probability of observing 0 or 1 during a measurement is equal. Then the Hamiltonian slightly changes and forms two identical valleys (b), separating the two states. A change in the magnetic field corresponds to an increase or decrease of the barrier. The quantity that controls the external magnetic field is called a *bias* (we have already encountered this term in the Hamiltonian 1.8). If there is a bias, the qubit minimizes its energy going in the new ground state (c). Together with this quantity, there is also another term that is typical of quantum computation. The qubits can in fact influence each other, through the phenomenon of the *entanglement*. It can make two qubits tend to end up in the same state or in the opposite state. We then have, for example,

$$|q_1\rangle + |q_2\rangle \longrightarrow \alpha\,|00\rangle + \beta\,|11\rangle \tag{1.10}$$

Note the coefficient can be set trough this coupling term, with a device called *coupler*. During the anneal, the qubit states are delocalized in the energy diagram. When a problem is formulated, i.e. the Hamiltonian H_P is defined, the strength of the couplers and the biases are set by the machine, so that the problem's energy landscape is shaped.

In Fig. 1.4(a) we can see an example of a landscape associated with a certain problem of two qubits. Following the algorithm of QA, the quantum annealer then finds the energy minimum, giving back the classical state that represents the minimum of all the possible



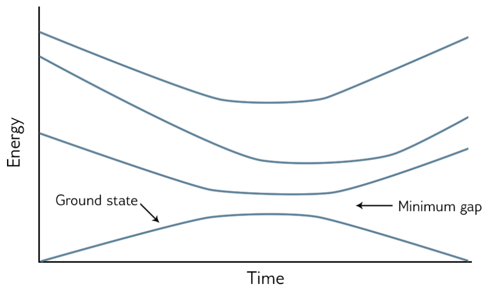
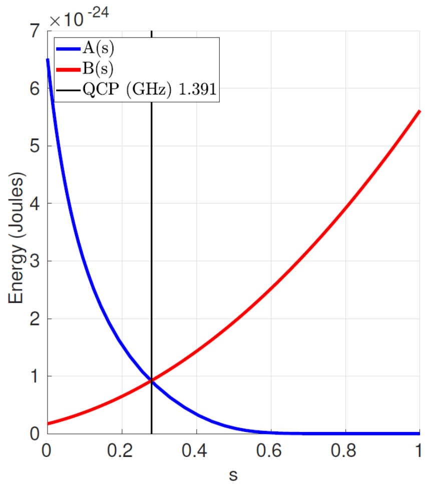


Figure 1.4: (a) shows an example of an energy landscape, pointing out the best answer, while (b) shows how over time the energy levels change. In particular it underlines the point in which the first excited state approaches the ground state, giving the system a possibility to escape from the g.s.



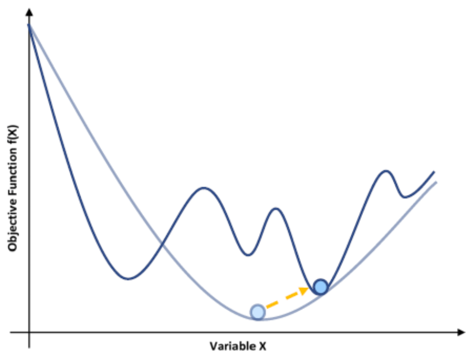


Figure 1.5: Fig. (a) shows the changing over time of A(s) and B(s), where $s = \frac{t}{t_f}$. Note that they are both expressed in units of Joules. Fig.(b) shows how H changes over time, with the system starting from the g.s of H_D and finishing at the ground state of H_P

ones. A way to better visualize what happens during quantum annealing in D-Wave system is to visualize the energy landscape and how it changes over time. The process starts whit the system in the ground state of the Hamiltonian H_D, well separated from the other energy levels. As the problem Hamiltonian H_P is slowly introduced, some eigenvalues could get closer to the ground state. As they get closer, the probability that the system jumps into an higher energy state increases. As shown in Fig. 1.4(b), there is a point in which the first exited state comes closer to the ground state and then goes away. That point is called the *minimum gap*.

Because of this approach, the system could jump into a different state from the lower one. To limit this fact the thermal fluctuations must be controlled, and that is achieved by bringing the system to very low temperatures.

We now draw the graph of the total Hamiltonian 1.7 in function of the parameters $A(t)$ and $B(t)$. Usually an anneal has $A(t) \propto \frac{t}{t_f}$ and $B(t) \propto (1 - \frac{t}{t_f})$ with t_f the total annealing time. In particular, in *D-Wave* they follow the graph in Fig. 1.5(a)

This implies the changing in the total Hamiltonian over time, obtaining for example the situation described in Fig. 1.5(b) If the D-Wave's quantum annealing works correctly, the system should stay in the ground state throughout all the process and in particular at the end of it.

However, at the moment D-Wave's QPU is not fully evolved. In fact, despite its 5000 physical qubits, these are not fully connected, meaning that too complex problems cannot be embedded. That is why D-Wave is currently developing a new topology for its QPU.

Chapter 2

Unfolding in High Energy Physics

This second chapter concerns the Unfolding problem, which occurs very often in high energy physics. In fact it happens frequently that due to the limited resolutions of the measuring device the distributions are subjected to distortions and random fluctuations. The *unfolding* is the procedure of correcting these distortions. In Section 2.1 we define the problem, giving a mathematical description of it. Then, in Section 2.2 we present three main classical method of unfolding, in particular the ones we will use in our analysis. In the end an unfolding example is presented, in order to better visualize what we have previously discussed.

2.1 Problem definition

In particle physics, as well as in medical physics and other disciplines, a frequent issue involves the distortion of physical observable distributions. This distortion arises during the measurement process due to the limited resolution of the instruments. Therefore, physicists cannot compare the observed data with the theoretical ones.

The **Unfolding** technique (also called deconvolution or restoration) is the process of correcting these alterations. As we are going to see, this process can be complex and scientists when it is possible try to avoid it. For example, if the goal is to compare the result with the prediction of an existing theory, one can simply modify the prediction to include the distortions of the detector, and this can be directly compared with the measurement. This is way simpler than unfolding. However, without unfolding the measurement cannot be compared with the results of other experiments, in which the effects of resolution will in general be different. In that case it is necessary to unfold, and that is why we are now focusing on it.

Consider a variable x of which we want to determine the distribution f(x). During the experimental measurements, one came inevitably across some distortions that changes the shape of the measured f(x). This is mainly due to the finite resolution in the measurement and the detection efficiency, $\epsilon(y)$, being less than 100%.

These phenomenons, together with the background noise, leads to *migration* and *distortion* effects in the measured distribution.

Each observed event is therefore characterized by two quantities: a **true** value x, which is unknown, and an **observed** value y, which is the one we measured. Consequently we can define $g(y) \equiv g_M(y)$ as the *measured* distribution and $f(x) \equiv f_T(x)$ as the *truth.* The goal of the unfolding is hence to find $f_T(x)$ knowing $g_M(y)$.

The mathematical foundations of unfolding are related to the descriptions of the inverse problem given by Fredholm integral equation of the first type [11]:

$$g_M(\mathbf{y}) = \int K(\mathbf{y}, \mathbf{x}) f_T(\mathbf{x}) dx \tag{2.1}$$

where as explained before $f_T(\mathbf{x})$ is the true distribution of the variable $\mathbf{x} = (x_1, x_2, ..., x_J)$ and $g_M(\mathbf{y})$ is the measured distribution for the variable L-dimensional $\mathbf{y} = (y_1, ..., x_L)$. These two distributions are related to each other by the convolution with the *Kernel* function $K(\mathbf{y}, \mathbf{x})$, which is known in the unfolding case as the *smearing function* as it describes all the detector effects on the measurement.

In particle physics usually the distributions are one-dimensional, which means that they can be described by histograms representing the expected number of counts of the variable for a given interval. So now the goal is to find a more accurate relationship of this kind specifically for histograms.

Suppose there is an experiment in which occurs a certain number of events m_{tot} described by the vector $\mathbf{m}$, and assume they follow a certain $f_T(x)$. It is clear that the probability to find the true value x in bin j is

$$p_j = \int_{binj} f_T(x) dx \tag{2.2}$$

We will consider now the expectation value of the total number of *real* events μ_{tot}, which is different from m_{tot} as the first represents the physics we expect the system to follow, while the latter is the real physics of the events. Usually μ_{tot} is obtained through Monte Carlo simulations. In the unfolding problem, our goal is therefore to estimate m_i with μ_i, as the *real* distribution is impossible to know. Because of that, m_i does not even enter the present formulation of the problem. Instead, we will construct our relations directly using μ_i.

By using 2.2, we can then express the expected number of events in the j-th bin as:

$$\mu_j = \mu_{tot} \ p_j \tag{2.3}$$

It can be written as $\boldsymbol{\mu} = (\mu_1, ..., \mu_M)$ and it is called the '*true histogram*'. In analogy to what we have said now we can describe how we expect the *measured* value to be due to the detector effects. We can then define the probability to observe a certain value y in bin i to be

$$p'_i = \int_{bini} g_M(y)dy \tag{2.4}$$

As said before, it is important to notice the difference between the *measured* value n_i and the one we *expect* to be measured ν_i. It is then useful to define also the '*measured histogram*' $\mathbf{n} = (n_1, ..., n_N)$. This is the N-bin histogram filled with the observed values of the variable x. Note that in general, N $\neq$ M. Following the same reasoning established previously, in this formulation we will start by using the expected value ν_i, which are the ones we can know given the $g_M(y)$, i.e. the expected distribution of the measured data. Note that the relationship between the real values and the expected ones will be much more clear in the example presented in Section 2.3.

By using the law of total probability[12] the expectation values ν_i can be also expressed as

$$\begin{aligned}
\nu_i &= \mu_{tot} \text{ P(event observed in bin i)} \\
&= \mu_{tot} \int dy \text{ P(observed in i | true value y and detected)} \, \epsilon(y) f_T(y) \\
&= \mu_{tot} \int_{bini} dx \int dy \;\, s(x|y)\, \epsilon(y)\, f_T(y)
\end{aligned} \tag{2.5}$$

Here $s(x|y)$ is the conditional p.d.f. for the measured value x given that the true value was y. We will call

$$r(x|y) = m(x|y)\, \epsilon(y) \tag{2.6}$$

the **response function**, and $m(x|y)$ the **migration function**. Tthe difference between these two is that the response contains also the information about the efficiency, while the migration tells only how the true value y changes in x (i.e. "migrates" from y to x). The **efficiency** is a quantity representing how the detector is good at detecting events. As we will see in Chapter 4, it is the ratio between the number of true events that were successfully detected and reconstructed in each bin ρ_i and the number of events that truly occurred in each bin μ_i. This means:

$$\epsilon_i = \frac{\rho_i}{\mu_i} \tag{2.7}$$

If we write 2.5 by breaking the integral over y into a sum over bins, we can then write

$$\begin{aligned}\nu_i &= \sum_{j=1}^{M} \frac{\int_{bini} dx \, \int_{binj} dy \, m(x|y) \, \epsilon(y) \, f_T(y)}{\mu_j/\mu_{tot}} \mu_j \\ &= \sum_{j=1}^{M} R_{ij}\mu_j \end{aligned} \tag{2.8}$$

where we have defined the **response matrix** R by

$$\begin{aligned}R_{ij} &= \frac{\int_{bini} dx \int_{binj} dy \, m(x|y) \, \epsilon(y) \, f_T(y)}{\int_{binj} dy \, f_T(y)} \\ &= \frac{\text{P(observed in bin i and true value in j)}}{\text{P(true value in bin j)}}\end{aligned} \tag{2.9}$$

obtained by substituting the relation 2.3 into 2.8. In addition to the effects studied above, i.e. limited resolution and efficiency of the detector, it is necessary to include the noise caused by some background process. If we consider β_i as the expectation value for the number of events originated from background process, the relation 2.8 can be modified as

$$\nu_i = \sum_{j=1}^{M} R_{ij}\mu_j + \beta_i \tag{2.10}$$

It is important to understand that $\boldsymbol{\beta}$ includes not only the background process, but also all the fakes measurements, which means all the detected signals created by the detector which do not correspond to real events. We will see better this in Chapter 4.

To sum up what we have said above, we have the following vectors:

- $\mathbf{m} = (m_1, ..., m_M)$, which describes the *real* distribution, i.e. the one we want to find with the unfolding procedure.
- $\boldsymbol{\mu} = (\mu_1, ..., \mu_M)$, that represents the *expected* value of the *real* distribution. It is known also as the **truth**.
- $\mathbf{n} = (n_1, ..., n_N)$, that is the real number of the entries observed, i.e. the measured values. It is also called the **reco** distribution.
- $\boldsymbol{\nu} = (\nu_1, ..., \nu_N)$, the expectation values for the observed number of entries.
- R, the **response matrix**. As seen from eq. 2.9, it contains all the information about the conditional probability that an event will be found with measured value x in bin i given that the true value y was in bin j. To sum up, it tells how the distribution is altered during the measuring phase.

The goal is to find μ assuming that the other quantities in eq. 2.10, that is the response matrix and the background effects, are known.

2.2 Unfolding classical algorithms

Throughout the years a lot of unfolding algorithms have been developed, especially with the establishment of machine learning methods. In this chapter we will focus on three classical methods commonly used in high energy physics. The unfolding methods that will be discussed here are the ones that we will use for the analysis. In particular, we are interested in two main methods: **matrix inversion (MI)**, **iterative bayesan unfolding (IBU)**. In the end of the Section, another method is presented, the **regularisation** method, which will not be directly used in our analysis but it will be the starting point in order to express the unfolding in terms of a quantum annealing problem.

2.2.1 Matrix inversion method

We begin by examining the first one. The *MI* method is the first one that comes to mind when trying to solve equation 2.10. This, however, often leads to an unacceptable solution, as we will see. Consider the case where the response matrix can be inverted and where the number of bins in the true and in the observed histograms are equal, $M = N$. We obtain

$$\mu_i = \sum_{j=1}^{M} R_{ij}^{-1}(\nu_j - \beta_j) \tag{2.11}$$

As explain above we cannot know exactly $\boldsymbol{\nu}$, so an obvious choice for his estimators can be given by the corresponding data values $\mathbf{n}$. We can then rewrite 2.11 as

$$\mu_i = \sum_{j=1}^{M} R_{ij}^{-1}(n_j - \beta_j) \tag{2.12}$$

This relation can also be derived formally from the principle of maximum likelihood. In this thesis we deal with counting experiments, so it is fairly general to assume that the data in each bin are independent Poisson observations. We can define then the likelihood function as

$$\mathcal{L} = \nu_i^{n_i} \frac{e^{-\nu_i}}{n_i!} \tag{2.13}$$

Consequently the maximum likelihood estimator for $\boldsymbol{\nu}$ can be obtained by maximizing $\mathcal{L}$ or, more conveniently, the $log(\mathcal{L})$:

$$\frac{\partial log\mathcal{L}(\mu_i)}{\partial \mu_i} = 0 \qquad \forall i \tag{2.14}$$

that can be easily solved obtaining exactly $\boldsymbol{\nu} = \mathbf{n}$, and consequently equation 2.12. Unfortunately, this simple solution is not always working.

What is happening? The main reason for the failure arises from the fact that we do not have the *expectation values* $\boldsymbol{\nu}$, we only have the data **n**, which are subject to statistical fluctuations. As a consequence, R views these fluctuations as effects of smearing by the detector (and not of statistical origins) and uses this input to reconstruct $\boldsymbol{\mu}$. This process magnifies the fluctuations back into the result, and this leads to an unacceptable solution. This is strictly related to the fact that MI is a ill-conditioned problem , which means that small changes in the input parameters could lead to big changes in the output. This is evident in 2.3. Despite of its problems, *MI* gives a general idea of how unfolding works and provides a good starting point for other methods.

2.2.2 Iterative Bayesian method

It is in fact evident from the discussion made in 2.1 that there is the need to incorporate prior knowledge of the distribution. This suggests that a possible solution could be given by a Bayesian approach [13].

That is the main idea behind the **Iterative Bayesan Unfolding** method. This procedure can be described as a "cause and effect" model following the Bayes theorem. We can indeed identify the cause C_i as the number of events in bin i of the *true* histogram, while the effect E_j as the events in the i-th bin of the *measured* histrogram. It is therefore clear that the effect E_j is known, while the exact cause C_i is impossible to determine. However, thanks to some knowledge about the migration, efficiency and resolution, we can estimate the probability that a certain event belongs to a defined cause $P(E_j|C_i)$. (which means we can determine the number of events observed based on the true values.) This implies that the number of events ν_i in the measured histogram can be expressed as the sum over all the possible causes that give that effect divided by the efficiency that the cause i has an effect. (That efficiency can be expressed as the sum of all the effect's probability caused by a fixed C_i.)

$$\nu_i = \frac{1}{\epsilon_i} \sum_j P(E_j|C_i)\mu_j \tag{2.15}$$

If we equals this relation to 2.8, we have that

$$R_{ij} = \frac{P(E_j|C_i)}{\epsilon_i} \tag{2.16}$$

This means that by knowing the response matrix, we know also this combined probability. Following a similar reasoning, it is clear that the number of events in the real histogram can be seen as

$$\mu_j = \sum R_{ij}^{-1}\nu_i = \frac{1}{\epsilon_i} \sum_j P(C_i|E_j)\nu_i \tag{2.17}$$

This means that $R^{-1}_{ij} = P(C_i|E_j)/\epsilon_i$

We have then find a way to calculate the *inverse* of the response matrix by estimating this probability. Using Bayes theorem, we have that

$$\begin{aligned} R^{-1}_{ij} &= \frac{P(c_i|E_j)}{\epsilon_i} = \frac{P(E_j|C_i)\,P_0(C_i)}{P(E_j)\,\epsilon_i} \\ &= \frac{P(E_j|C_i)P_0(C_i)}{\epsilon_i \sum_{k=1}^{n_{cause}} P(E_j|C_k)\,P_0(C_k)} = \\ &= \frac{P(E_j|C_i)\,P_0(C_i)}{[\sum_{k=1}^{n_{effects}} P(E_k|C_i)]\;[\sum_{k=1}^{n_{cause}} P(E_j|C_k)\,P_0(C_k)]} \end{aligned} \tag{2.18}$$

where $P(E_j|C_i)$ is determined by 2.1, and $P_0(C_i)$ is the *a priori* probability of the cause C_i, i.e. the probability of the *truth* distribution. This is usually taken by Monte Carlo simulations or by using a simple constant distribution.

Once that we have find R^{-1}_{ij}, we can calculate the *a posteriori* probability of the cause C_i,

$$P_1(C_i) = \frac{\mu_i}{\sum_j \mu_j} = \frac{\sum_j R^{-1}_{ij}\nu_j}{\sum_j \mu_j} \tag{2.19}$$

that is a better estimator for the *real* distribution than the Monte Carlo simulation by which we have found P_0 before. It is clear that now we can put 2.19 inside 2.18, and find a new R^{-1}, that can be used to find $P_2(C_i)$.

By using this method iteratively we will obtain improved estimation of the *real* distribution, until the algorithm reaches a certain stability. To have an idea, in most of the commons practical applications 4 iterations are sufficient.

2.2.3 Tikhonov regularisation

Although this methods are very useful and gives quite good results as we shall see in 2.3, it is important for the seek of this thesis to discuss another unfolding method, that will be used later in Section 4.1.1. This alternative approach is to impose a certain value representing the smoothness of the *real* histogram $\boldsymbol{\mu}$. This is also known as **regularisation** of the unfolded distribution. When explaining the *MI* method we have introduced the likelihood function as 2.13. We can then follow the idea that $\boldsymbol{\mu}$ maximize the value of $\mathcal{L}$, as explained before. It is clear that we can obtain a similar result by minimizing the chi-square function. In this way, we ensure that the reconstructed data fits the observed data as closely as possible. We can therefore express the chi square relation as

$$\chi^2 = \sum_i \frac{\nu_i - (R\mu)_i}{\sigma_i^2} \tag{2.20}$$

where the notation is the one defined earlier and σ_i represent the standard deviation. If we write this relation in matrix form, we obtain

$$\chi^2 = (\boldsymbol{\nu} - R\boldsymbol{\mu})^T W(\boldsymbol{\nu} - R\boldsymbol{\mu}) = ||\boldsymbol{\nu} - R\boldsymbol{\mu}||_W \tag{2.21}$$

where we have written W as the covariance matrix. For clarity, we will omit from now the subscript W.

In addition to this term, which represent the acceptability of the solution, we need to define a measure of its smoothness by introducing an arbitrary function $S(\boldsymbol{\mu})$ called the **regularization function**. In *Tikhonov regularisation*, this function is define as

$$S(f_T(y)) = -\int (\frac{d^k f_T(y)}{dy^k})^2 = -||D\boldsymbol{\mu}||^2 \tag{2.22}$$

Here we have used the fact that our f_T are discrete, so that we can go from integration to sum. We have then used D as the Laplacian operator. That is because usually a common choice for the derivative is k = 2, in order that $S(\boldsymbol{\mu})$ is related to the curvature. We can now minimize using the method of the Lagrangian multipliers with a free parameter λ and we obtain:

$$y = ||\boldsymbol{\nu} - R\boldsymbol{\mu}|| + \lambda||D\boldsymbol{\mu}|| \tag{2.23}$$

That is the function that we want to minimize in function of $\boldsymbol{\mu}$. Note that in practice as for the Matrix Inversion method we will use $\mathbf{n}$ instead of $\boldsymbol{\nu}$.

2.3 RooUnfold example

In particle physics, one of the most commonly used framework for performing the unfolding is *RooUnfold*[14]. In this Section we have used it to show an example of the unfolding process in order to give a visual explanation of how it works. This would be very useful to better understand Chapter 4.

We consider a double-picked distribution of 10000 samples, generated from Monte Carlo simulation by two gaussians with expected value respectively of $x = 3.3$ and $x = 6.4$, and with standard deviation of $\sigma_1 = 0.9$ and $\sigma_2 = 1.2$, having 20 bins equally spaced. This distribution corresponds to our *truth* distribution. We now simulate a measurement, by introducing the detectors parameters. For this simulation we choose constant values for the efficiency $\epsilon = 0.6$, while the smearing function is a gaussian with mean smear $b = -0.13$ (bias) and standard deviation $s = 0.21$ (smear). We do not consider background effects which would give us fake data. These enable us to generate the *measured* through a loop over all the events. Fig. 2.2 shows these two distributions.

The *response* matrix is generated separately with a Fill or Miss process, inside an identical loop of the distributions generation. The choice to use an identical loop but not the same to generate the response is necessary to have statistical fluctuations. Otherwise

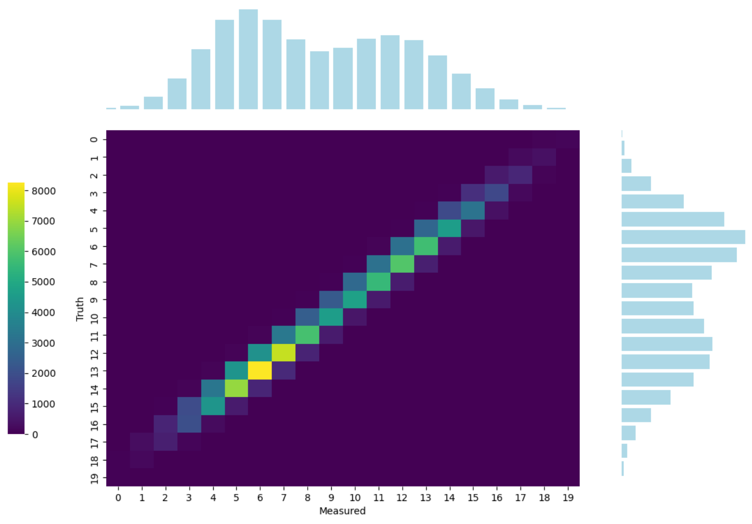


Figure 2.1: This figure shows the response matrix having the truth histogram on the top and the measured on the right. It is important to understand that these are not the truth and the measured that we unfold, but come from another Monte Carlo generation. Following the notation, these are $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$, while we are unfolding $\mathbf{n}$.

the unfolding would be without sense, because we would already know everything. We cannot generate the response with the same distributions that we want to unfold, but they have to be slightly different. The response basically is a model that tells how the distribution changes, but it has to be applied to slightly different distributions, otherwise the unfolding result would be perfect bu with no meaning.

If an event is both generated and detected, it means that it is present both in the truth and in the measured distributions, and so it fills the response. If it is only generated instead it fills just the truth, not the measured. That is why it is called a Miss event. Thanks to this process we obtain the response in Fig.2.1.

Having the truth, the measured and the response enable us to use RooUnfold methods to compute the unfolding and calculate the chi square. In particular, we are interested in Matrix Inversion and IBU methods. Fig. 2.3(a) shows the result.

An interesting aspect is to see what happens if we change some parameters. We have already seen that *Matrix inversion* is quite a good way to unfold, except from the fact that small changes in the initial conditions could lead to big unfolding differences. If for example we take the same distribution as before and we just change the smearing value from $s = 0.21$ to $s' = 0.4$, which does not change significantly the measured, we clearly see that MI collapses while IBU, which is more stable and regular, remains valid. We see that in Fig. 2.3(b)

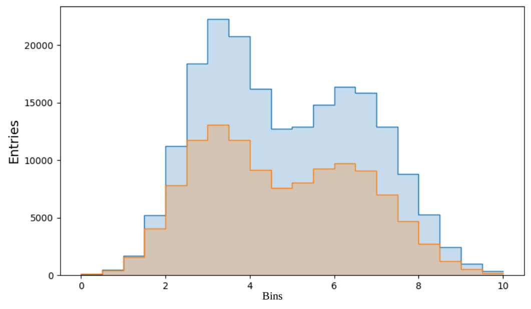
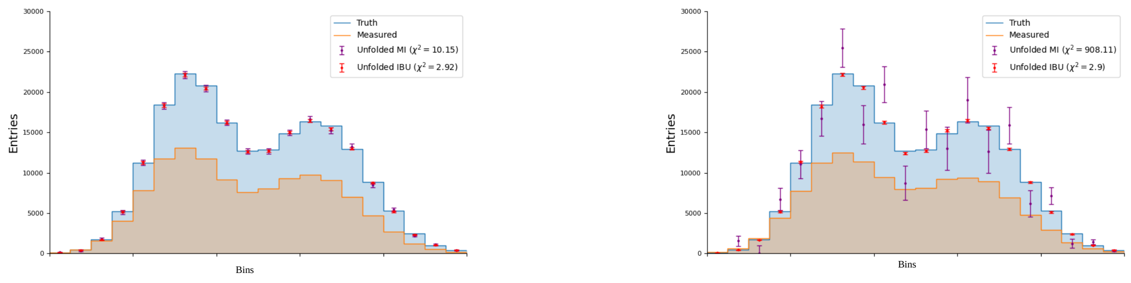


Figure 2.2: These are the truth and the measured distribution of a double picked gaussian distribution. The data are generated through Monte Carlo simulations, and has no physical meaning.



(a) (b)

Figure 2.3: (a) shows the unfolding result comparing the two methods presented, giving optimal final chi square. Figure (b) shows that with a simple changing in the smearing parameter s from 0.21 to 0.4 (resulting in a little distortion on the measured) MI method inevitably fail, while IBU remains stable.

Chapter 3

The $t\bar{t}$ pair production

This chapter is about a brief introduction to particle physics. As the goal of this thesis is to use the quantum annealing unfolding to high energy physics distributions. First the Standard Model of particle physics is presented, describing what we currently know about all the particles and the forces of the universe. We then give a presentation of the particle physics detectors at CERN, in particular ATLAS, which is the one used in the analysis. In the last Section of this chapter we give an overview of the pair of quarks $t\bar{t}$ studied in the analysis of this thesis. It is explained their production and their decay, focusing on the decay channel we will study in Chapter 4.

3.1 Standard model of particle physics

The standard model of particle physics (SM) is a theory that describes the fundamental particles and how they interact. It was developed throughout the latter half of the 20th century, and it integrates how the three interaction forces (electromagnetic, strong and weak forces, not including gravity) influences elementary particles, such as quarks and leptons. Despite recently studies have proved some holes in the theory, it is presently one of the best models to interpret the world.

It is based on three gauge groups, associated respectively to the strong interaction and to the electroweak force:

$$SU(3)_c \otimes SU(2)_L \otimes U(1)_Y \tag{3.1}$$

3.1.1 Particles of SM

The elementary particles of the SM are divided in two main groups: **fermions** and **bosons**. The first ones have semi-integer spin, while the latters have only integer values [15].

The fundamental "matter" particles are all fermions. They are divided in 6 **leptons** and 6 **quarks**. The main difference between them is that leptons don't feel the strong force, while both interact with the electro-weak interaction. For each one of these it exists a corresponding anti-particle with same mass, but opposite quantum numbers, such as charge. Fermions are organized in three different generations, each formed by two leptons and two quarks, listed in Table 3.1.

Quarks	**Charge**	**Spin**	**Leptons**	**Charge**	**Spin**
u	2/3	1/2	e	-1	1/2
d	$-1/3$	1/2	ν_e	0	1/2
c	2/3	1/2	μ	-1	1/2
s	$-1/3$	1/2	ν_μ	0	1/2
t	2/3	1/2	τ	-1	1/2
b	$-1/3$	1/2	ν_τ	0	1/2

Table 3.1: The fundamental fermions in the Standard Model organized by generation. The table shows some fundamental properties of the fermions, such as charge and spin.

The 6 quarks have 3 possible colors (red, green, blue) with their corresponding quantum numbers. Leptons, conversely, have the leptonic number L, which needs to be conserved during interactions. Then, to sum up, we have 6 leptons and 6×3 bosons, with the respective anti-particles, for a total of 48 fermions. On the other hand, particles are subjected to forces. In order to not violate relativity principles, these interactions are necessarily mediate by other particles, which have integer spin and are so bosons. They are presented in Table 3.2, where we we note that we have 12 bosons (13 with the H boson).[16]

Boson	**Spin**	**Charge**	**Force Mediated**
Photon (γ)	1	0	Electromagnetic
W^+	1	1	Weak
W^-	1	-1	Weak
Z^0	1	0	Weak
Gluon (g)	0	1	Strong
Higgs (H)	0	0	-

Table 3.2: The bosons of the Standard Model, their spin, and the forces they mediate. Note that according to QCD, we need 8 different type of gluons, which are all identified by g. Another important aspect not shown in the table is that they are all massless except for W and Z bosons.

While γ is massless, which means that its interaction has an infinite range, W and Z are massive and have a finite range of action. Note that despite also g are massless

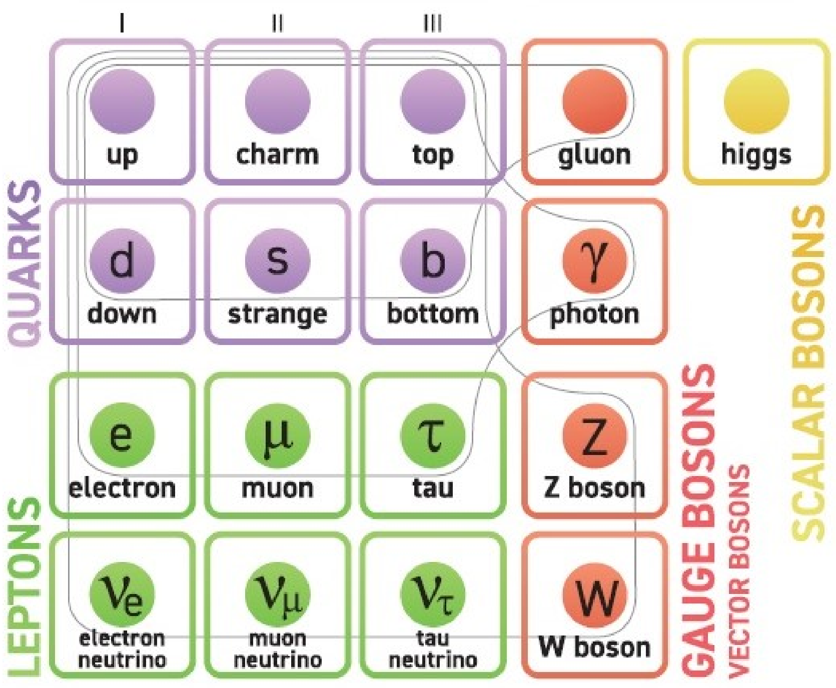


Figure 3.1: Fundamental particles of SM grouped depending on their interactions

they also have a finite range, due to a phenomenon called quark confinement [17]. Another important boson for the SM theory is the *Higgs boson* H, which is the cause of fundamental particles having mass. The total of $48 + 12 + 1 = 61$ particles that we have discussed above, can be sum up in Fig. 3.1.

3.1.2 Forces of SM

As said previously in 3.1.1, the SM provides an explanation for three fundamental interactions: **electromagnetic** (EM), **weak** and **strong** [17].

EM interaction is the only one that has also a classical description. Maxwell's equations imply that two bodies electrically charged interact between each other through a photon γ. According to quantum field theory, in particular **Quantum Electrodynamics** (QED), γ is a massless boson, which means as explained before it has an infinite range of interaction. QED, first introduced by Feynman [18], is described by a gauge group $U(1)_{EM}$. This implies that when two fermions electrically charged interact with each other, they simply exchange photons, so that one fermion emits γ while the other absorbs it. An example of this is shown in Fig. 3.2(a).

From the vertex of this interaction we can evaluate the *fine structure constant* α, which is an adimensional constant representing the strength of the EM force. Its value depends on the energy scale used, but usually its common value is:

$$\alpha \simeq \frac{1}{137} \tag{3.2}$$

Secondly, the Standard Model explains the *weak interaction* through the **Quantum Flavourdynamics** (QFD). In this interaction, particles change the quantum numbers associated to their flavours. Therefore, diagrams as the one in Fig. 3.2(b) are permitted. The force is mediated by three bosons: two electrically charged ($W^{\pm}$), which is present in charged currents interaction, and a neutral one (Z), for neutral currents. As for QED, also QFD has its adimensional constant indicating the strength G, which is:[16]

$$G = \sqrt{2}\,\frac{g_W^2}{8m_W^2} \tag{3.3}$$

There are two main difference with EM interactions. Firstly, the weak interaction violates two symmetries: parity (P) and charge (C). In addition, as it is a non Abelian gauge theory, it allows to have vertices in absence of fermions. EM theory does not allow this.

To better understand these two phenomenons, the SM in reality provides a single theory called **Electroweak interaction**. According to this, EM and weak interaction are seen as two different aspects of a single electroweak interaction. In particular, while at the beginning of the universe ($En \simeq 250GeV$) these two interactions were combined, at lower energies their differences emerged. To have just one group able to bring a massless boson γ and three massive bosons $W^{\pm}$, Z, the simplest is defined as:

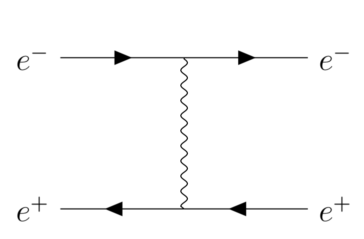
$$SU(2)_L \otimes U(1)_Y \tag{3.4}$$

where Y is the *hypercharge* $Q = I_3 + Y/2$, Q the elecrtic charge and I_3 the third component of the weak isospin, which is one of the always conserved quantum numbers. Note that in order to be verified, this theory must include the Higgs meccanism to give masses to the bosons, otherwise (as it is at high energies) they all four would be massless.

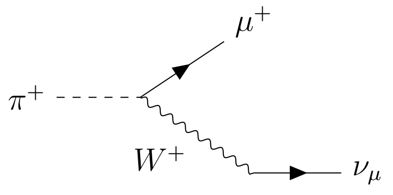
In the end, the third force described by SM is the *strong interaction*. According to **Quantum Chromodinamics** theory (QCD), this last interaction involves only quarks and their **color** quantum numbers. It is therefore related to the $SU(3)_C$ gauge group. It is important to know that in this interaction, the energy scale varies significantly with the numbers of quark flavors, so for convention it is used the value of $250MeV$ (the energy needed to confine two quarks in a pion.) According to QCD, there are 8 different mediators massless called gluons. Obviously then, the 6 quarks presented previously in Sec. 3.1.1 all are triplets of colour. As for the weak interaction, QCD is a non-Abelian gauge group, so 3 or 4 gluons vertices are permitted. An example of an allowed diagram is shown in Fig. 3.2(c)

To sum up all the various charges discussed above, absolutely conserved quantum numbers in the Standard Model are: *electric charge (Q)*, *weak isospin (I_3)* and *leptonic number (L)*.[1]

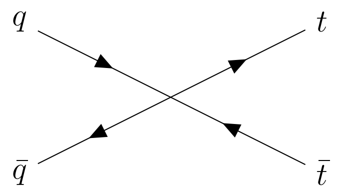
[1]There is also the baryonic number B, which we have not introduced because it concerns system of



(a) Bhabha scattering



(b) π^+ ($u\bar{d}$) decay via weak force



(c) Production of $t\bar{t}$ via $q\bar{q}$ annihilation

Figure 3.2: Feynman diagrams representing the three fundamental forces. (a) shows an EM interaction between an electorn and a positron, (b) shows the decay of a pion, composed by two quarks, into a doublet lepton-neutrino while (c) represents the production of $t\bar{t}$ trough $q\bar{q}$ annihilation via strong force.

3.2 The ATLAS detector

The European Organization for Nuclear Research, commonly known as CERN, [19] is one of the world's largest and most respected centers for scientific research. It was established in 1954 by 12 European nations with the aim of exploring fundamental questions about the universe thanks to the study of particle physics. Throughout the years many particle accelerators have been built with this goal, and the current LHC accelerators is CERN's most significant and ambitious project.

3.2.1 LHC

The **Large Hadron Collider** (LHC) is the world's most powerful particle accelerator, located at *CERN*. It consists of a 27-kilometer ring of superconducting magnets and detectors buried underground, with a medium depth of 100m. The aim of this accelerator is to bring protons to very high energies (with velocity near c) and collide them with each other. It was built with an ideal energy of 14 TeV, which we are very close to reaching.[16]

The collider consists of two parallel beam pipes, separated by 194 mm, through which beams circulate in opposite directions. These pipes intersect at only four interaction points, where four different big experiments have been set up, as shown in Fig. 3.3:

- ATLAS (A Toroidal LHC ApparatuS): a general-purpose detector investigating a wide range of physics
- CMS (Compact Muon Solenoid): designed for purposes similar to ATLAS, but uses a different technical solutions

quarks and anti quarks. Here we work only with single quarks.

- LHCb (Large Hadron Collider beauty): specialized in investigating the slight differences between matter and antimatter by studying the b quark
- ALICE (A Large Ion Collider Experiment): dedicated to heavy-ion physic and designed to study the physics of strongly interacting matter at extreme energy densities

.

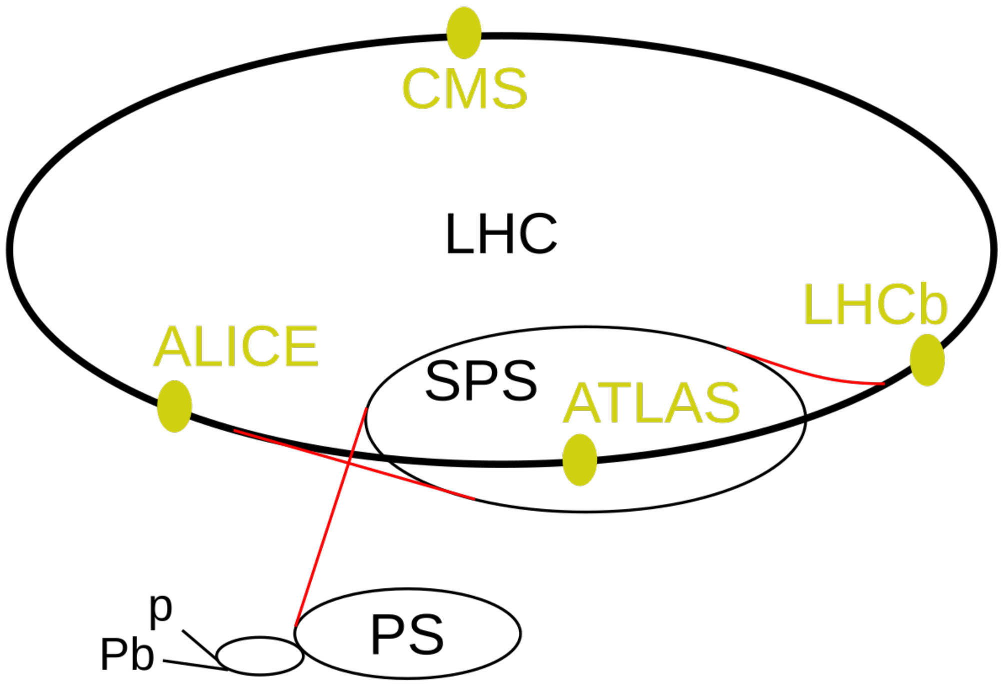


Figure 3.3: A view of the complete structure of the LHC facility at CERN. We can see the 4 main experiments (ATLAS, CMS, LHCb, ALICE) and the small accelerators that inject the protons inside the collider.

LHC is made by superconductive magnets that enable us to control and direct the particle beam and by 8 superconductive cavities that accelerate it. As the magnet are superconductive, they specifically need very low temperatures to work properly, in particular below $2\,K$. This enables the passage of circulating currents of the order of $35\,kA$ producing magnetic field above $8\,T$. It is important to underline the fact that protons, thanks to a system of acceleration outside LHC, are injected at energy up to $450\,GeV$.

In an accelerator, a very important parameter is the *luminosity*. The *instantaneous luminosity* $\mathcal{L}$ is defined as the ratio between R, the rate of produced events, and the cross-section of the process σ

$$\mathcal{L} = \frac{R}{\sigma} \tag{3.5}$$

This parameter depends on beam properties, such as the number of protons N, the revolution frequency f, the relativistic factor γ, the normalized beam emittance η, the beam beta function β and the luminosity reduction factor F. It depends also on the number of bunches per beam k_B, that are the groups of particles that are packed together inside the accelerator and travel as a single unit. Usually the bunches are of about 10^{11} protons. Therefore the total equation is:

$$\mathcal{L} = \frac{\gamma f k_B N^2}{4\pi\sigma} F \tag{3.6}$$

LHC at ATLAS is able to reach peaks of $\mathcal{L} = 10^{34}\ cm^{-2}s^{-1}$, which corresponds to to about 1 billion proton-proton collisions per second at a rate of 40 MHz.

3.2.2 ATLAS experiment

In Section 3.2.1 we have seen how *LHC* hosts 4 main big experiments, including *ATLAS*. ATLAS is a cylindrical detector that spans 44 meters in length and 25 meters in diameter, weighing around 7,000 tons. Its size and sophisticated technology enable it to capture and analyze a wide range of particle interactions resulting from LHC collisions. As explained before, it is able to collect events coming from proton-proton collision at an energy up to 13 TeV and about $10^{23}\ cm^{-2}s{-}1$ of luminosity, with bunches of up to 10^{11} protons colliding 40 million times per second.

Key features of the ATLAS detector include the *inner traking system*, the *electromagnetic* (EM) and *hadronic calorimeters* (HAD) and the *muon spectometer*. Fig. 3.4 presents a schematic view of ATLAS detector.

The *inner detector* has a crucial role on tracking particles and measuring their momentum. It is formed by three sub-detectors, all contained within a cylindrical envelope and surrounded by a 2 T magnetic field. In the center there is the Pixel Detector, which is the closest to the collision point and provides very high spatial resolution. There is then the SemiConductor Tracker SCT, made of silicon strip detectors and able to track precisely over a larger area. Outside these two, there is the Transition Radiation Tracker TRT, which uses gas-filled straw tubes to provide additional tracking and electron identification.

The main purpose of the two *calorimeters* is to absorb particles (respectively photons and electron for EM and hadrons for HAD) and converting their energy to detectable signals, both of liquid Argon (LAr) type. In particular, the hadronic calorimeter is provided by a scintillator-tile calorimeter, which is separated into a large barrel and two smaller extended barrel cylinders, one on either side of the central barrel.

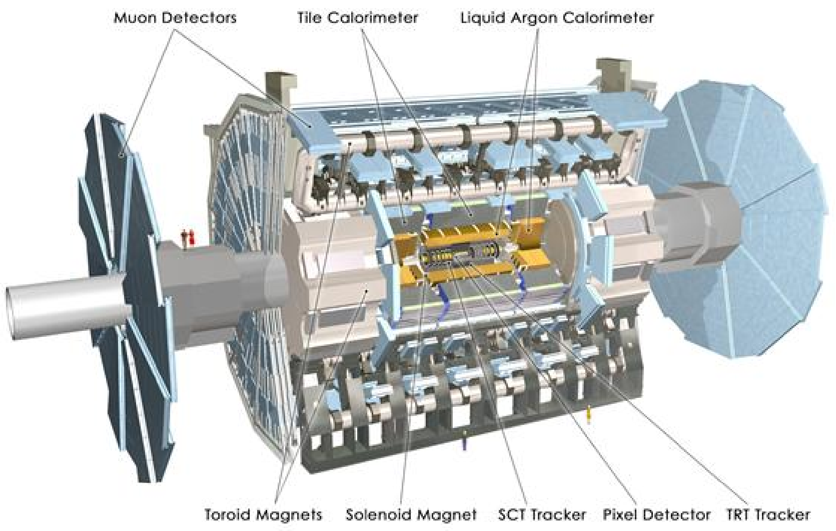


Figure 3.4: A schematic view of the ATLAS detector and its main components: Inner tracking systems (Pixel detector, SCT and TRT trackers), calorimeters and muon spectometers.

They are then surrounded by the *muon spectrometer*, which gives good muon identification and their momentum. It is also able to determine unambiguously the charge of high transverse momentum muons.

In the end, all the collected data are passed to a very high efficient trigger system, which reduces the data rate to a manageable level for storage and analysis. Schematically, it is made up by two levels. The first one is the Level-one Trigger, which uses a subset of total detector information to select potentially interesting events reducing the data rate from 40 MHz to about 100 kHz. There is then the High Level Trigger, a software-based system that further reduces the rate to about 1 kHz for storage.

3.3 $t\bar{t}$ distribution

Up until now, we have generally discussed the Standard Model and high-energy physics accelerators. Now we must study more specifically the quarks and the distribution we are going to analyse in Chapter 4. We are dealing with top quarks, the which we have already introduced. In this Section we will firstly present the particle and its properties, and then we will see how the distribution of $t\bar{t}$ can be produced and how it can decays.

3.3.1 The top quark particle

According to the SM, as seen in Section 3.1.1, the top quark is an elementary particle belonging to the third generation of quarks. Its existence was postulated in 1973 by Makoto Kobayashi [20] and Toshihide Maskawa to explain the observed CP violations in kaon decay, and was discovered in 1995 by the Collider Detector and D0 experiments at Fermilab. It is one of the most particular elementary particle observed so far having very singular properties that let him to behave in a completely different way in respect to the lightest elementary components. In the Standard Model of particle physics, it is the strongest coupling at the scale of the weak interactions and above.

Like all other quarks, the top quark is a fermion with spin spin 1/2 and participates in all four fundamental interactions:

- electromagnetism, having electric charge of $Q/e = +2/3$
- weak interactions, as it is part of the weak isospin doublet, with $I_3 = 1/2$
- strong interactions, having colour charge
- gravitation, which however is not explained in SM

Among all the elementary particles, the top quark is the most massive with a mass of $\simeq$ 173 GeV. According to the SM, due to this it has a very short lifetime of $\tau \simeq 0.5 \times 10^{-24} s$, which is smaller than the typical tome scale of hadronization.[21] Differently from all the others quarks, due to this rapidity t decays semi-weakly into a real W boson and a b quark before it can hadronize, without forming mesons or baryons with other quarks:

$$t \longrightarrow Wb \tag{3.7}$$

In fact, unlike the other quarks, it has never been observed a bounded state involving this quark. This gives physicists a unique opportunity to study a bare quark alone, i.e. not combined to form hadrons, and can only be observed as such. Following 3.7, the W boson decays as well, giving three different decay channels:

$$\begin{aligned} &W \longrightarrow l\nu_l \\ &W \longrightarrow qq' \longrightarrow jj \end{aligned} \tag{3.8}$$

where l is a lepton and ν_l its corresponding neutrino. q and q' are then quarks and j their corresponding jets. [2] This will be better understood later in Section 3.3.3.

[2]A jet of quarks is a phenomenon which occurs when a quark or gluon is ejected at high speeds. Due to the property of color confinement in QCD, quarks and gluons cannot exist freely and independently. As a result, they quickly hadronize, forming a stream or "jet" of particles that includes hadrons

3.3.2 $t\bar{t}$ production process

There are multiple processes that can lead to the production of top quarks. we can divide them into two categories: top quark-antiquark ($t\bar{t}$) pair production, which happens through strong interactions, and single top quark production (t), through weak interactions. At the LHC, the top quark is produced mainly through gluon-gluon fusion, giving $t\bar{t}$ pair. Sometimes it can happen that it is produced singly, with an associated W boson and a b quark. However, this second case is very rare than the other one. At the moment, the top quark is studied in ATLAS and CMS experiments at CERN.

As said before, there are two main way of production of t. For what concerns $t\bar{t}$ production, we can link it to two main processes. At LHC the dominant mechanism is the gluon-gluon fusion, where two gluons collide with each other and decay into the $t\bar{t}$ pair. The second process, which prevailed at Tevatron (the accelerator where the top quark was discovered) but is less present at today accelerators, is the annihilation of a quark with its anti-quark, producing a gluon that originates the pair. For having an idea of the relative contributions for the top production at the LHC, we have:

$$\begin{aligned} &B(q\bar{q} \longrightarrow t\bar{t}) < 20\% \\ &B(gg \longrightarrow t\bar{t}) > 80\% \end{aligned} \tag{3.9}$$

In addition to the $t\bar{t}$ production, which as discussed above uses the strong force, the top quark can be also produced via the weak interactions. In this process, however, only one top quark is produced, and this method is called "single top quark production". To this reason we will not go into detail for this production, as it is not the goal of this thesis. Indeed in this work, as we will see in Chapter 4, we are dealing with correlation between a top and an anti-top quark.

3.3.3 $t\bar{t}$ decay process

As discussed previously in Section 3.3.1, the top quark decays before hadronization and it does not form bound state with other quarks. This implies that is the only quark we can "observe" alone.

Top quark decays almost 100% of the time into a W boson and a b-quark, having the decay fraction as

$$R_{Wb} = \frac{BR(t \longrightarrow Wb)}{BR(t \longrightarrow Wq)} = \frac{|V_{tb}|^2}{|V_{tb}|^2 + |V_{ts}|^2 + |V_{td}|^2} \tag{3.10}$$

Here, $|V_{tb}|^2$, $|V_{ts}|^2$, $|V_{td}|^2$ are the coefficient of the Cabibbo-Kobayashi-Maskawa (CKM) matrix responsible for the $t \longrightarrow q$ transitions. They contains the information of the strength of the decays. Thus, this implies that $|V_{tb}|^2$ is much grater in magnitude than the others. As it is for the laws of the conservation of the charge, if it is a top quark we will obtain W^+ and a b quark while if we have $\bar{t}$ we will have W^- and a $\bar{b}$ quark.

Since also W is instable, depending on how it decays we will have three different channels. Experimentally it decays into hadronic final states qq' with a branching fraction of approximately 2/3 and into a charged lepton l and its corresponding neutrino ν with a decay fraction of approximately 1/9. This implies the following distinctions for the $t\bar{t}$ decay channels:

- *Dilepton channel*, in which both W bosons decay into a lepton-neutrino couple:

  $$t\bar{t} \longrightarrow W^+bW^-\bar{b} \longrightarrow l^+\nu_l b\ l'^-\bar{\nu}_{l'}\bar{b} \tag{3.11}$$

  usually we have $l = e$ or μ.

- *Semileptonic channel*, where only one W decays into a doublet lepton-neutrino and the other became a couple of quark-antiquark.

  $$t\bar{t} \longrightarrow W^+bW^-\bar{b} \longrightarrow l^+\nu_l b\ \bar{q}q'\bar{b} \tag{3.12}$$

  or

  $$t\bar{t} \longrightarrow W^+bW^-\bar{b} \longrightarrow q\bar{q}'b\ l^-\bar{\nu}_l\bar{b} \tag{3.13}$$

  It has very low background but as there are undetected neutrinos it cannot be solved kinematically.

- *All hadronic channel*, in which both W decay hadronically

  $$t\bar{t} \longrightarrow W^+bW^-\bar{b} \longrightarrow q\bar{q}'b\ \bar{q}q'\bar{b} \tag{3.14}$$

In our Unfolding analysis, we will consider the *semi-leptonic channel*, as we will see in the next Section. This is shown in Fig. 3.5.

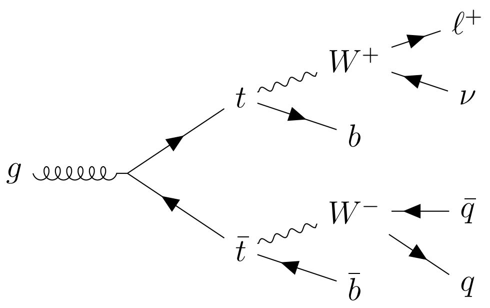


Figure 3.5: Semi-leptonic decay of $t\bar{t}$, that is the channel used in our unfolding analysis

Chapter 4

Analysis Strategy

This chapter fulfil the aim of this thesis. Up to now I have studied Quantum Annealing, Unfolding and a brief introduction of top quark physics. It is now time to combine all of this by understanding how I can translate the unfolding into a quantum annealing problem, and using this result to analyze data from a $t\bar{t}$ experiment. In the analysis carried out in this last chapter I want to know if this new unfolding technique could be better than the classical ones already introduced. My goal is also to explore the limitation of this process, by examining all of the three annealing methods provided by D-Wave system. To do this in Section 4.1 I first study how I can write the unfolding in the QUBO form, necessary for QA process. Then I give an overview of the *Qunfold* Pyhton package, explaining how it works and all the features that characterize it. Section 4.2 presents the data studied, providing all the information about the variables which I am going to unfold, showing all the distributions. In the end, Section 4.3 is the real study of the data. Here, all the unfolding techniques are used and compared to each other so that I can analyze and see the results. Clearly, it is the most important part of this work.

4.1 QUnfold

This Section deals with the introduction of *QUnfold*, a new and innovative Python package born to perform the Unfolding through Quantum Annealing techniques by using the public D-Wave systems's quantum annealer. The origin of the idea behind this framework traces back to a paper written by Di Sipio et al. in 2019 [22] where they analyzed and dephined the mathematical concept, proving that their approach was possible. However, they didn't go any further, as in their analysis they implemented a code that worked specifically for their data.

QUnfold was developed by three Ph.D students (G. Bianco, S. Gasperini, M. Lorusso) from the idea to transform this previous work into a functional and accessible tool [23].

The package starts with Di Sipio's idea, bringing it to life, and then creates an indipendent software program that uses the advantages of quantum annealing to solve unfolding problems.

In this chapter then I will explore the development of QUnfold, starting from the theoretical foundations and then moving to the main problems QUnfold's developers have faced. In particular, I examine the binarization problem and the three main unfolding methods developed by this software.

More precisely, in this Section I will study in detail how I can reconduct the Unfolding problem to a Quantum Annealing problem by using the QUBO formulation. I will then see that this problem is strictly connected to the one of binarization, which will be discussed in detail in Section 4.1.2. In the end I will discuss generally about the most significant implementations and functionalities of this program.

4.1.1 Unfolding as a QA problem

It is clear from the discussion made in Chapter 2 that each unfolding method implies matrix operations. To translate them to the quantum computing world, I need to take an intermediate step: I need to express them through binary optimizations, so that I can implement it on a quantum annealer. In particular, as explained before, the problem can be optimized by the quantum annealer machine using the quadratic unconstrained binary optimization (QUBO) formulation. The objective function associated with a QUBO can be formulated as

$$H(x) = \sum_i \sum_j H_{ij} x_i x_j \tag{4.1}$$

I now take into consideration the fact that x_i are binary variable, so that $x_i^2 = x_i$. I can then separate 4.1 as

$$H(x) = \sum_i a_i x_i + \sum_{i<j} b_{ij} x_i x_j \qquad x_i \in \{0, 1\} \tag{4.2}$$

That is the function I want now to minimize, with a_i and b_{ij} the constrains. The goal now is then to derive the QUBO formulation of the unfolding problem, expressed by equation 2.10. I start by considering the Tikhonov regularization, i.e. equation 2.23. To convert this equation into QUBO form (eq. 4.2), I start by using index notation, i.e.:

$$\begin{aligned} Ax &\longrightarrow A_{ij} x_j \\ AB &\longrightarrow A_{ij} B_{jk} \\ AB^T &\longrightarrow A_{ij} B_{ik} \end{aligned} \tag{4.3}$$

In this notation a summation over repeated indices is implicit. Thus, rewriting eq. 2.23, I have [22]:

$$\begin{aligned} H(\boldsymbol{\mu}) &= (\nu_i - R_{ij}\mu_j)(\nu_i - R_{ik}\mu_k) + \lambda(D_{ij}\mu_j D_{ik}\mu_k) \\ &= (R_{ij}R_{ik} + \lambda D_{ij}D_{ik})\mu_j\mu_k - 2R_{ij}\mu_j\nu_i + \nu_i\nu_j \end{aligned} \tag{4.4}$$

I can ignore the last term, since it is a simple constant that goes away when minimizing. Note that if I multiply each term, I have obtained the QUBO form. I thus have:

$$H(\boldsymbol{\mu}) = r_i\mu_i + \mu_j Q_{jk}\mu_k \tag{4.5}$$

I have now found the starting point in order to determine the QUBO weights, but in this notation μ_i and ν_i are still float, while I need binary variables.

The first thing that comes to mind is to use the standard and simplest n-bit integer encoding:

$$\mu_i = \sum_{j=0}^{n-1} 2^j x_{n\times i+j} \qquad x_i \in \{0,1\} \tag{4.6}$$

However, this is a bit limiting for two main reason: first of all, I can encode only integer number, while I usually deal with float ones, and secondly current quantum computer hardware requires the bit encoding to be small. Another important motive can be also the one of optimization. I want in fact to find an encoding method that runs faster and better. Is is then necessary to use some non-standard encoding. That is one of the main problem faced when developing the unfolding as a quantum annealing problem, as I will see in the next Section.

4.1.2 Binarization

It is now time to study how the binarization can be implemented into the code. When Di Sipio faced this problem, as it was not his goal to implement an user-friendly software, decided to use an heuristic way to build this binarization. In their paper, each μ_i is encoded with n bits, using an offset and a scaling parameter so that:

$$\mu_i = \alpha_i + \beta_i \sum_{j=0}^{n-1} 2^j x_{n\times i+j} \tag{4.7}$$

The values of these parameters were fixed, but they were chosen by attempt, so they fit very well their problem, but couldn't be extended to other cases. It is therefore evident that the main problem with this approach is given by the free parameters α_i and β_i, which are inevitably different for every λ, every i and every unfolding problem. I cannot derive a universal way to binarize the Tykhonov function.

The approach used in QUnfold is a bit different [24]. I reconsider now equation 2.23, that is the function I want to minimize. I have shown that it can be represented in the QUBO form, but I want to write it in binary. I need then a method of approximating each real variable by b binary variable.

To fulfill this I introduce, given a real number x, $\boldsymbol{x}_b$ to express that number in binary notation. I then denote by $\boldsymbol{p}$ the *precision vector*, that is a vector so that:

$$x = \boldsymbol{p} \cdot \boldsymbol{x}_b \tag{4.8}$$

In this way, if I have a symmetric matrix Q, the QUBO form $\boldsymbol{\mu}^T Q \boldsymbol{\mu}$ can be rewrite

$$\boldsymbol{\mu}^T Q \boldsymbol{\mu} = \boldsymbol{\mu}_b^T P^T Q P \boldsymbol{\mu}_b \tag{4.9}$$

where the P matrix is a diagonal matrix having the precision vector $\boldsymbol{p}$ on the diagonal.

Fully general, I can define $\boldsymbol{p} = (-1, \frac{1}{2}, ..., \frac{1}{2^{b-1}})$ of length b. From this, the set of integer multiples of $\frac{1}{2^{b-1}}$ in the interval $[-1, 1 - \frac{1}{2^b}]$ is the set formed by

$$C_{1,b} := \{\boldsymbol{p} \cdot \boldsymbol{\mu}_b | \boldsymbol{\mu}_b \in 0, 1^b\} \tag{4.10}$$

If I deal with vector of n-size then,, I cam express it by simply

$$C_{n,b} := \{(I_n \otimes \boldsymbol{p}) \cdot \boldsymbol{\mu}_b | \boldsymbol{\mu}_b \in 0, 1^b\} \tag{4.11}$$

whit I_n the identity matrix. In this way, starting from a QUBO form to be minimized of eq. 4.5, I can minimize it by writing:

$$\begin{aligned} &\min_{\boldsymbol{\mu} \in [-1,1]^n} (\boldsymbol{r}^T \boldsymbol{\mu} + \boldsymbol{\mu}^T Q \boldsymbol{\mu}) \\ = &\min_{\boldsymbol{\mu}_b \in [0,1]^m} (\boldsymbol{r}^T (I_m \otimes \boldsymbol{p}) \boldsymbol{\mu}_b + \boldsymbol{\mu}_b^T (Q \otimes P) \boldsymbol{\mu}_b) \end{aligned} \tag{4.12}$$

where I finally have written the QUBO form in binary terms. It is important to underline that in QUnfold package I have vectors representing the entries of an histogram. This means that I have to binarize $\boldsymbol{\mu} \in [-\delta, \delta]^n$. This is simply obtained by multiplying $\delta \cdot \boldsymbol{p}$ in 4.12.

4.1.3 Overview of the Python package

QUnfold Python package offers a user-friendly framework able to perform the Unfolding through the Quantum Annealing. The way it works is quite simple. As it is based on Numpy library [25], the first things that are needed are the Numpy arrays of the *response* matrix and of the observed distribution that needs to be unfolded, known as the *sample*. Having these two, one can build the QUBO form of its unfolding. This is obtained through the class *QUnfolder*, which takes as parameters these two arrays and the free parameter λ. This is the parameter introduced in eq. 2.23. This class enable us to express the Unfolding problem in the QUBO form necessary for the QA formulation, as seen in Section 4.1.1. To actually construct the QUBO matrix in the binary formulation the function *initialize_ qubo_ model()* of the *QUnfolder* class must be called. Once I have then constructed the Hamiltonian, the only thing that is left to use is to solve the optimization problem using the annealing. *QUnfold* package has three unfolding methods corresponding to the three different annealing techniques, i.e. the one presented in Section 1.2.

One of the most important feature of this package is the ability to run Toys Monte Carlo simulations. That is an important aspect in HEP analysis. Monte Carlo methods involving Toys are stochastic techniques used to model complex systems by generating random samples and analyzing statistical properties. In other word starting from the *reco* distribution, I smear it with a Poisson distribution to simulate the statistic uncertainty. I then solve the QUBO problem with this new *reco*, giving in return a slightly different solution from the one obtained without that smearing. The main idea is that by "running" N toys , i.e. N simulations or "fake experiments", I am sampling the possible statistical fluctuations of the solution. However, this process is quite complex and it is still a work in progress in QUnfold package, and I will not use it in the analysis.

The feature that I will use and study instead is the possibility to repeat the same unfolding N times (without smearing or other effects). This is necessary as the annealing is a probabilistic technique, which means that repeating the process several times could give different solutions at each run of the process. Therefore, it is evident that increasing N implies a better view of all the possible solutions. This number N can be set by the user thanks to the parameter *num_ reads*. But how can I choose among these N different solutions?

Imagine I have the N solutions $(\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_N)$ sorted so that $H(\boldsymbol{\mu}_i) \leq H(\boldsymbol{\mu}_{i+1})$. It is evident that $\boldsymbol{\mu}_1$ can be considered the best approximation to the real solution. Despite that, an approach that can give great benefits in practice is to consider a linear combination of all the solutions as a better approximation of the real value:

$$\min H(\boldsymbol{\mu}) \simeq \frac{1}{N} \sum_{i=1}^{N} e^{-\beta(H(\boldsymbol{\mu}_i)-H(\boldsymbol{\mu}_1))} \boldsymbol{\mu}_i \tag{4.13}$$

Here, β is a parameter, that in *QUnfold* is set to be $\beta = 100$, representing a good value[4]. The function that provides this method of choosing the best solution is implemented in the function *_post_process_ sampleset()*.

Another fundamental aspect of *QUnfold* is the possibility to calculate the chi square χ^2 of the unfolded distribution. Obviously to have this data one must also have the truth distribution. By knowing that, and calling $\boldsymbol{u}$ the unfolded distribution, the chi square is calculated through the relation [12]:

$$\chi^2 = \sum_i \frac{(u_i - m_i)^2}{m_i} \tag{4.14}$$

As I am interested in particular in the reduced chi square, I need to divide it by the degree of freedom, represented by the number of bins.

If the user of *QUnfold* works in the High Energy Physics (HEP) field, he probably has to deal with the classical unfolding $C++$ package *RooUnfold*. This means that he does not have Numpy arrays, but he has measured histogram stored as TH1 objects and the response as a *RooUnfoldResponse* object. A very good implementation of *QUnfold* concerns the compatibility of itself with *RooUnfold*, providing functions able to convert those objects in standard Numpy arrays. In the end, *QUnfold* is able to perform several classical Unfolding methods. For the analysis in this thesis I will focus ourselves in these techinques:

- RooUnfold framework
  - Matrix inversion (MI)
  - Iterative Bayesan Unfolding (IBU)
- QUnfold framework
  - D-Wave Simulated Annealing
  - D-Wave Hybrid solver
  - D-Wave Quantum Annealing

These three D-Wave methods are the ones explained in Section 1.2.

4.2 Data visualisation

It is now time to use the new unfolding method given by the quantum annealing to unfold real variables.

In this thesis, I use data generated from a research about quantum entanglement and Bell violation at LHC [26]. In this work physicists have generated a $t\bar{t}$ pair, which has decayed as explained in Section 3.7, in particular in Fig. 3.5. They have generated $\propto 10^8$ samples of this decays, and have calculated some variables which could demonstrate the entanglement. To be more precise, they wanted to find a correlation between the directions of the decay products of t and $\bar{t}$ so they could calculate the spin density matrix, essential to see the entanglement. They consider the two angles representing the angular space (θ, ϕ) and used them to describe the direction of the decay product of interest thanks to the the three-vector $\Omega_i = (cos\phi sin\theta,\ cos\theta,\ sin\phi sin\theta)$. Here, θ is the polar angle and ϕ is the azimuthal angle with respect to a reference direction. To better visualize this, see Fig. 4.1. I have used the helicity basis $\{\hat{r}, \hat{k}, \hat{n}\}$, with the plane

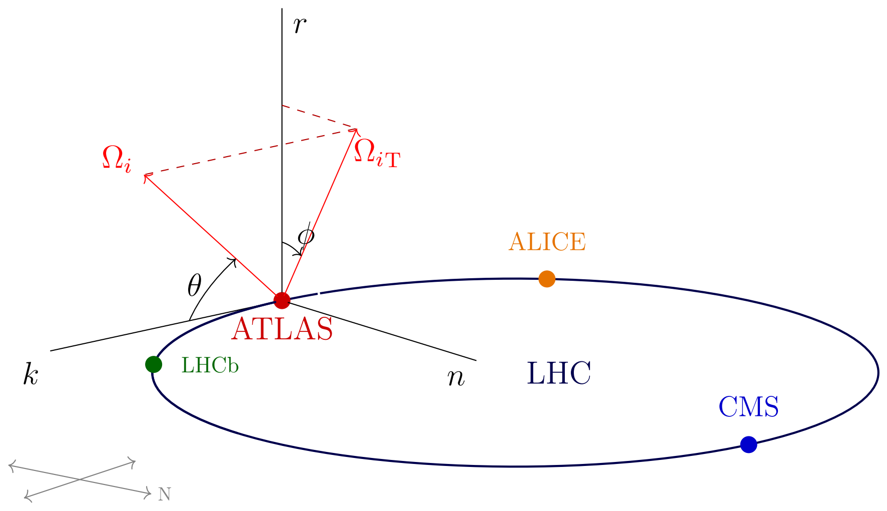


Figure 4.1: This figure shows the helicity basis $\{\hat{r}, \hat{k}, \hat{n}\}$ in the LHC frame of reference and how a general vector Ω_i can be decomposed in this basis. Note that in the ATLAS experiment the plane formed by $\hat{n}$ and $\hat{k}$ (and parametrized by ϕ) is the plane of the detector, while the direction of $\hat{k}$ is the direction of the beam, which is perpendicular to the plane.

rn being the detector plane, and k the transverse direction. As I have a two body decay, it is very useful to write θ^A to indicate that is the polar angle of the momentum of the decay product of particle A, and so with ϕ. In particular, following the goal of the paper from the which the data have been taken, I construct two main variables $\cos(\theta_P)$ and ϕ_P.[1] From Fig. 3.5, I consider the angles θ^t and ϕ^t formed by the lepton l in the top quark rest frame, and $\theta^{\bar{t}}$ and $\phi^{\bar{t}}$ the ones of the quark q in the rest frame of the anti-top quark. In this frame of reference, as $\hat{k}$ represent the transverse axis of the detector, it has the same direction of the incident beam p_t (or $p_{\bar{t}}$, i.e. the momentum of the particle that decays. Fig. 4.2 shows better this angles. Using these four variables, I can construct following the paper [26] $\cos(\theta_P) = \sum_i \Omega_i^t\, \Omega_i^{\bar{t}}$ and $\phi_P = f(\phi^t, \phi^{\bar{t}})$, where f is a certain function and Ω_i^A the variable already introduced for the decay of particle A. $Cos(\theta)$ is the variables I am unfolding. Although it is not relevant in the paper, I also want to unfold the mass distribution of the $t\bar{t}$ system, as for its peculiar shape it is an important tool to test the unfolding method.

[1]In [26], these two variables were useful to calculate the Bell operator that indicate entanglement.

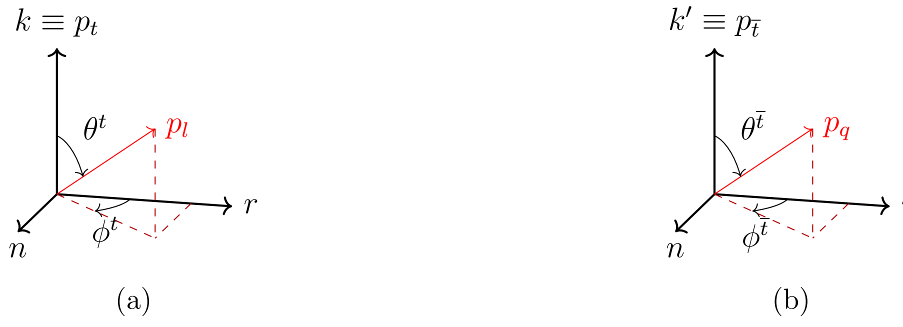


Figure 4.2: These two figures shows the two frame of reference used, by denoting that $\hat{k}$ ($\hat{k}'$) correspond to the direction of the incident beam and the vector p_l (p_q) is the one of the decay product. Note that in the rest frame of $t\bar{t}$ $\hat{k}$ and $\hat{k}'$ are anti-parallel vectors.

As explained, I used data about the decay process of the Semi-Leptonic channel:

$$t\bar{t} \longrightarrow l^- l^+ \, \nu \, \overline{\nu} \, b \, \overline{b} \tag{4.15}$$

The data used in this thesis do not rely on actual experimental measurements from particle colliders like the LHC but rather on simulated data based on theoretical models. Events are computer generated involving tools commonly used in particle physics, such as Monte Carlo methods, to model the decay processes and analyze the quantum correlations. I can divide the generation process in two parts. Firstly, according to the SM theory, the $t\bar{t}$ distribution decays, and the Semi-leptonic channel is selected. These is called the **parton level** of the generation, representing the theoretical and ideal process. This data are then processed through *Rivet* (Robust Independent Validation of Experiment and Theory) [27], which is a system able to simulate the measurements effects. Thanks to it, quarks became jets and neutrinos became MET (Missing Transverse Energy) [2] It is then able also to simulate the geometrical acceptance of, in this case, ATLAS detector. The data processed in this way formed the **particle level**, which corresponds to the measured events in a real experiment.

4.2.1 Parton level

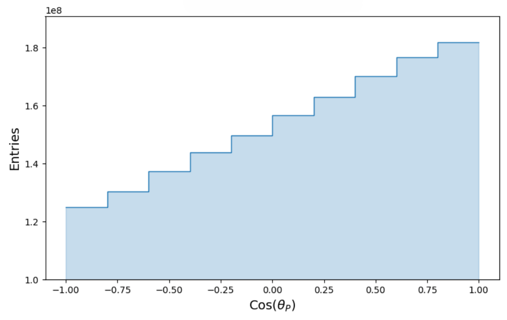
In *HEP* the term "parton level" of the analysis refers to the study of the fundamental interactions between particles according to the theoretical models. Parton-level calculations are then essential for making predictions about the outcomes of high-energy experiments and for understanding the structure of hadrons at a fundamental level. Therefore

[2]This concept of Missing Energy is usually applied in hadron colliders. MET is the measure of the difference between the total energy that is expected to be observed and what is actually measured in the transverse plane. Indeed, as neutrinos cannot be detected, there would be some energy missing, and that is what in a real experiments would be observed.

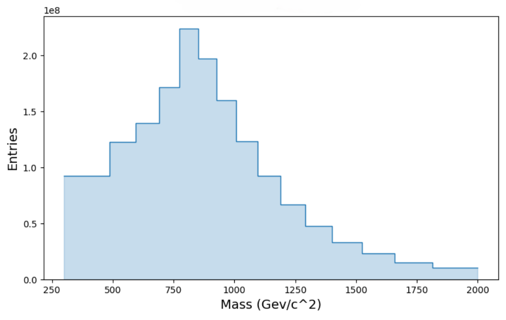
in this level the focus is only on the theoretical models, forgetting about all the measurement problems. It is clear that the distributions at *parton level* corresponds to the *truth* distributions described in Chapter 2.

To build the parton (or truth) histogram I start from the ROOT TTree file containing all the experiments generated data. Here the data are collected in various branches, where each ones represents a variable. These branches are associated with a set of data entries, each entry corresponding to a single event.

In this case I focus on two branches: $cos(\theta_P)$ and m. I decide not to consider the variable ϕ_P because as it is the angle on the detector plane it is isotropic, meaning its distribution would be plane. I create a loop so that for each event of every branch the relative histogram is filled. I obtain the following **truth** histos:



(a) Truth-Parton histo $cos(\theta)$



(b) Truth-Parton histo mass

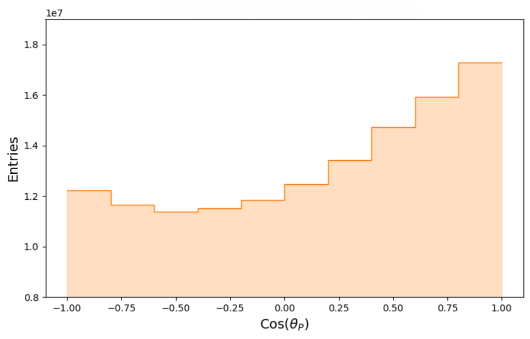
Figure 4.3: I have on the left the histogram Truth-Parton of $cos(\theta)$ distribution and on the right the one of the mass expressed in GeV$/c^2$. I see that I have bin of the order of 10^8 samples.

4.2.2 Particle level

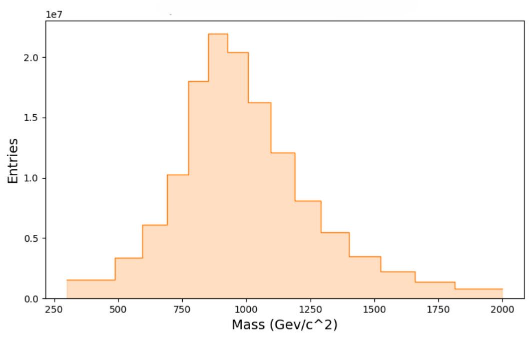
In the "particle level" the focus shifts to what I measure. While the parton predicts how the particles should behave, this level tells how this behaviour are influenced by the measurements effect. In other words, the event observed in the particle level are those that would be detected in a real experiment. The transition from parton-level to particle-level is then crucial to simulate a real experiment and to compare the result with the real ones. That is the reason why it is in this level that neutrinos are transformed in MET and quarks in jets, as it is what I would really observe.

To compare this distributions to what I know about the Unfolding, the *particle level* clearly represents the *truth* distribution. To build those histograms, as I have said for the parton level, I loop for each branch I am interested in and fill the histo if the value of the event is inside a certain range.

As for the branches choosen in the parton level, I obtain as **measured** distributions the following in Fig. 4.4



(a) Measured-Particle histo $cos(\theta)$



(b) Measured-Particle histo mass

Figure 4.4: Comparing these with the Parton, I immediatly see the difference on the shapes. Another important aspect is the number of samples for each bin. They are an order of magnitude lower compared to the truth bins.

When I compare the truth and the measured, I note that these distributions are quite different, both in term of shape and of numbers of events. Therefore, this means that because of detector effects an event in the TTree file could be for example True in parton level and False in the particle. By this, I can build the following histograms:

- The **miss** histo: this is the distribution of events that are theoretically generated, i.e. they are present in the truth histogram, but they were not detected in the measurement phase, being lost. That is why they are called "Missed events". I will call it $\boldsymbol{l}$.
- The **fake** histo: this is filled by the events that were not generated, so do not comparable with the truth, but were incorrectly reconstructed by the detector. I will refer at it as $\boldsymbol{f}$.

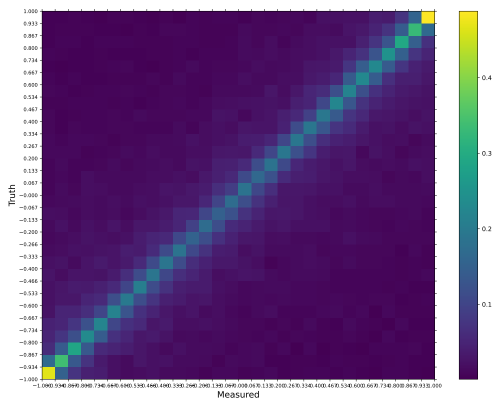
As for the other histograms, I loop over all the events for each branch and fill these two new histos. The results are grouped in Fig. 4.6 to Fig. 4.9. Thanks to these, I am able to calculate an important variable of the Unfolding problem, the **efficiency**. Following eq. 2.7, I can identify the histogram of the true events successfully reconstructed as the difference between the truth distribution and the miss one. Therefore I have:

$$\epsilon_i = \frac{\mu_i - l_i}{\mu_i} \tag{4.16}$$

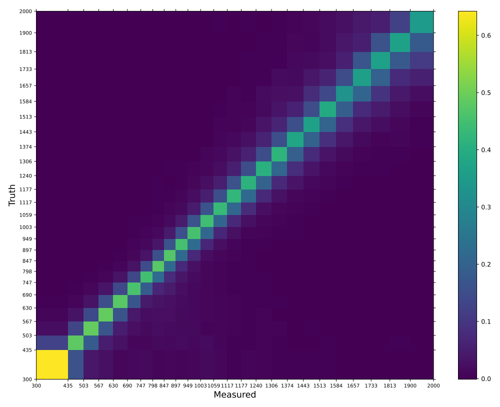
The efficiency histograms are plotted in Fig. 4.10 and 4.11.

4.2.3 Response matrices

One of the key element of the unfolding is the response matrix. This matrix contains all the information about smearing, efficiency etc... and basically tells us all the effects that the truth distribution has undergone. In order to build it from the ROOT TTree file, I need to loop over all the events and fill the matrix only if the event is both generated and detected. In this way I cancel out all the fakes and the misses. However, this matrix is not yet the response. By simply filling it in this way, I only simulate the smearing effect, but I do not consider the efficiency effect. This means that I now have the *Migration matrix*. As in eq. 2.6, to find R I need to multiply with the efficiency in eq. 4.16. Fig. 4.5 shows the Migration matrices



(a) Migration matrix of $cos(\theta)$



(b) Migration matrix of $cos(\theta)$

Figure 4.5: Here are shown the two Migration matrices. I have not plot the Response so that I can visually separate the effects of migration from the ones of the efficiency in Fig. 4.12

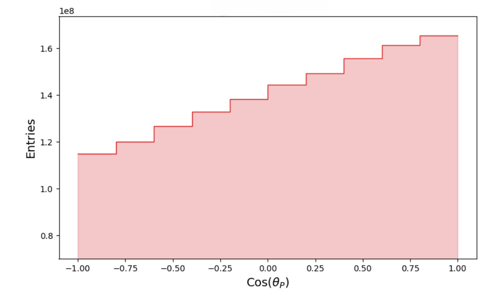


Figure 4.6: Miss histo of $cos(\theta)$

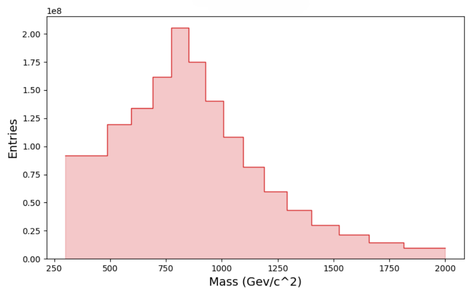


Figure 4.7: Miss histo of mass

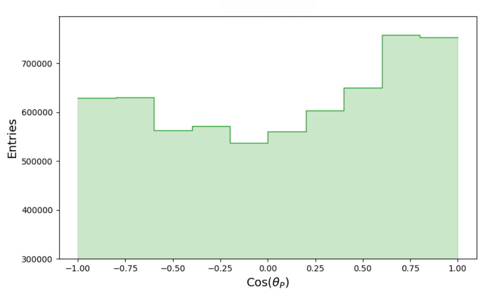


Figure 4.8: Fake histo of $cos(\theta)$

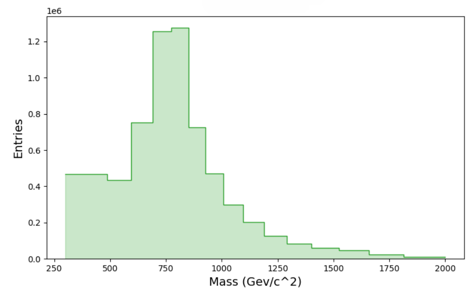


Figure 4.9: Fake histo of mass

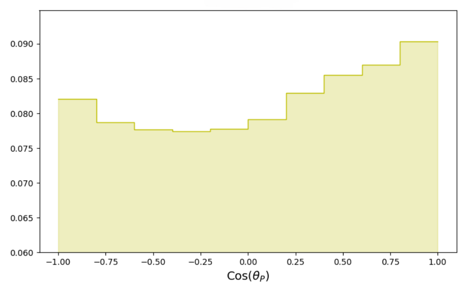


Figure 4.10: Efficiency histo of $cos(\theta)$

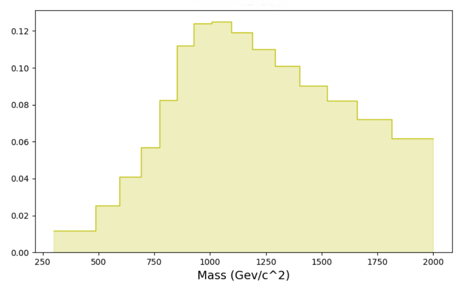


Figure 4.11: Efficeincy histo of mass

Figure 4.12: I have plot in this page all the histos of the effects to which the truth distribution is subjected during the measurement phase. These histos are necessary to the unfolding procedure

4.3 Unfolding results

To unfold using both RooUnfold and QUnfold methods, I need first to understand what kind of data I have. In Fig. 4.3 I have the truth distribution. That is what I want to obtain, i.e. the goal of the unfolding. Fig. 4.4 shows instead the measured distribution, obtained after the effects of the Miss, the Fake and the Efficiency. Following eq. 2.11 I understand that I need to pass to the unfolder constructor not the measured (ν_j in the equation) but the measured without all the fakes ($\nu_j - \beta_j$). Then, as I have the Migration and not the Response, I need to multiply M with the efficiency. By passing these two new object, I can safely call the unfolder ($RooUnfoldBayes()$ for IBU and $QUnfolder()$ for QUnfold). Note that I have not mentioned Matrix inversion (MI). In fact, in this analysis MI is meaningless, as the response is generated with the same generation of the other distributions. As explained before, this is not the general case of the unfolding as the truth generally is unknown, and this fact implies that MI is perfect without errors or uncertainties ($\chi^2 = 0$). That is why I will not compare it to the others.

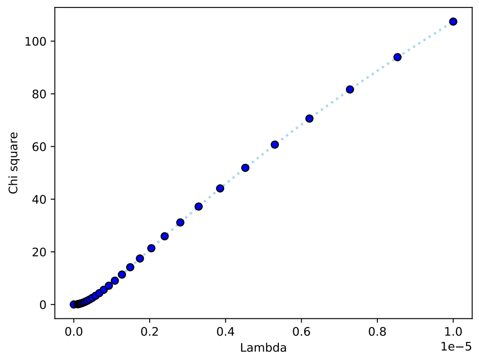
I need to underline another important aspect of this analysis. The data distributions are highly populated, and the technique offered by QUnfold using D-Wave is still in phase of development. This means that the problem is properly defined and the binarization works correctly, but there are still some problems with the dependency on statistical fluctuations. That is why to perform the unfolding of all this data I use QUnfold only to transform the unfolding problem in a QUBO problem, and then I compute the calculations with a classical solver. In order to have results more reliable at this developement stage I compare IBU with the unfolding via QUBO problem. However, in order to represent the unfolding in the QUBO form I need to choose λ, following what I have learnt in Section 4.1.1. As a good choice of this improves the results, I should theoretically run the unfolding several times each of it with a different value. In this way I obtain Fig. 4.13. I then choose for the unfolding the values of λ presented in Table 4.1. With this

	$\boldsymbol{cos(\theta)}$	$\boldsymbol{mass}$
$\boldsymbol{\lambda}$	0.0	0.0
$\boldsymbol{\chi^2}$	0.01759	0.00351

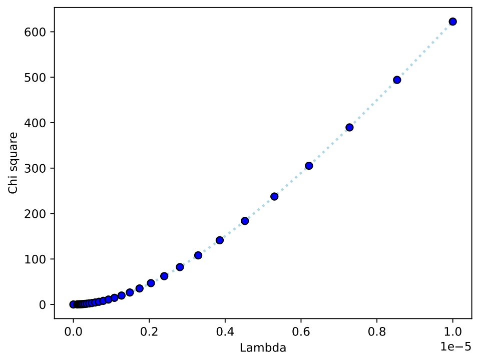
Table 4.1: From this table I notice that both for $cos(\theta)$ and mass, the best regularization parameter is 0, which means zero regularisation.

choices, I can run the two methods and compare them. I obtain the results shown in Fig. 4.14 and 4.15.

As said before, *QUnfold* is not yet able to give consistent results when facing complex problems. That is why to run the Simulated and the Hybrid Annealing I will consider just the $cos(\theta)$ variable. However, I need to reduce the quantity of data. To do this, I simply cut through the mass. I choose to consider only the data that were given by particles having a mass bigger than $1100 GeV/c^2$.



(a) Chi square for $cos(\theta)$



(b) Chi square for mass

Figure 4.13: This picture shows the value of the chi square in function of the regularisation parameter λ. By these I can easily see that the best for each distribution is $\lambda = 0$.

I can now start the analysis via SA and HA. I know that the annealing is a statistical method. This means that every time I run it I could obtain a different solution. That is the reason why it is very important to choose a valid number of reads, changing the parameter $num\_reads = N_R$ in the QUnfolder object. In this way I can repeat the unfolding and choose the best solution, as previously explained in Section 4.1.3. To better visualize this process, I run Simulated Annealing in function of N_R and calculate the chi square for each iteration. I imagine that by using small N_R the result should fluctuate much more despite running it with a big num_reads. In other words, an increased N_R correlates with a broader range of potential solutions to select from. To calculate the error on the χ^2 then I run each unfolding with all the parameters fixed $n = 10$ times, and calculate it as

$$\Delta\chi^2 = \frac{\sigma_{\chi^2}}{\sqrt{n}} \tag{4.17}$$

In this way I am able to obtain the graph in Fig. 4.16. By these I clearly see that increasing the reads implies a lower chi square. However it is still quite far from the one obtained via the classical solver of the QUBO problem. I therefore imagine that only with an infinite number of reads the χ^2 would tend to that value. Obviously this aspect of *QUnfold* needs to be improved both in precision of the solution and in compiling time, as using thousands of reads takes some time.

An important difference between Simulated Annealing and the Hybrid Solver is that I have much more control on the first one. In fact, the Hybrid Annealing is internally managed by D-Wave, and I can not set the parameter num_reads.

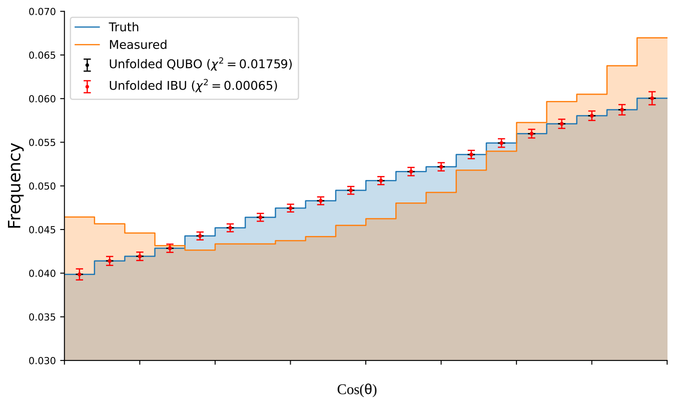


Figure 4.14: Here the comparison between the unfolding obtained through the QUBO form and the results of IBU method is shown. From the χ^2 I see that for this variable IBU is slightly better.

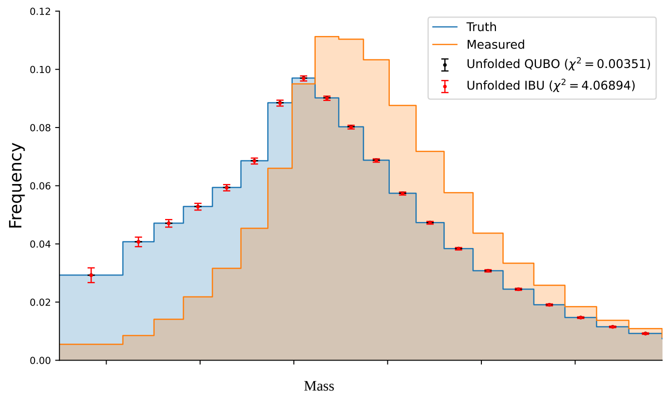


Figure 4.15: Here the second result of this analysis is presented. This is the cmparison between QUBO and IBU, showing that the first is much more precise than the Bayesian method

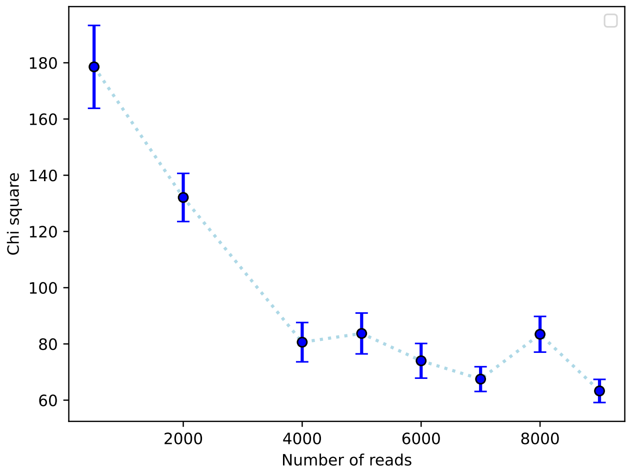


Figure 4.16: This graph shows how the chi square decreases with the number of reads. Obviously having these values for the χ^2 is still not sufficient to be compared with the classical methods, but I imagine that by having n_reads$\longrightarrow \infty$ I would reach the χ^2 given by the classical solver of the QUBO form.

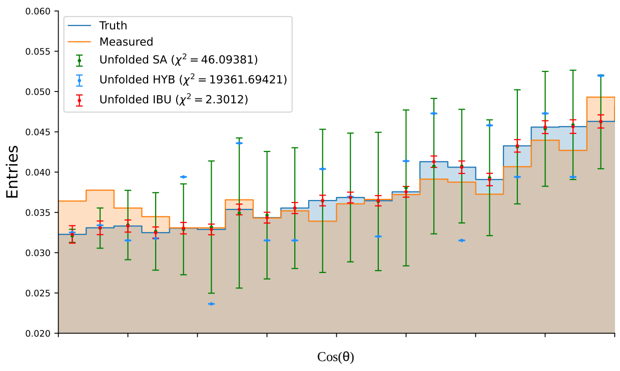


Figure 4.17: Here is presented the comparison between IBU, SA (with $N_R = 7000$) and HYB. It is evident that IBU is much more stable and precise, but SA visually still gets the unfolding, despite its chi square. The hybrid method however inevitably fails, as I cannot control it as the others but it is internally managed by D-Wave's.

This means I am not able to plot Fig. 4.16 for HA, I can only unfold. Once I have chosen for SA the best value of N_R, I can finally plot the annealing results, and compare them with IBU. The results are shown in Fig. 4.17

An important aspect to notice is that despite the SA chi square is consistently different from IBU ones, visually the unfolding does not represent a failure. Different is the situation for the Hybrid sampler, as the impossibility to increase the number of reads makes impossible for us to improve the solution, which remains inconsistent.

4.3.1 Quantum annealing

It would be interesting to see if Quantum Annealing could provide a solution, but the complexity of the problem prevents embedding it into D-Wave's QPU. In theory, as I did with simulated annealing, I could simplify the data by cutting down the energy. However, this approach would essentially invalidate the analysis, as it would require reducing the data by at least four orders of magnitude. Note that reducing the data does not imply reducing the complexity. In fact, simply cutting down the energy means losing control over the data, having response matrices very far from being diagonal.

That is why in this Section I just present an unfolding example with Monte Carlo generated data. To do so, I consider he same generation used in Section 2.3, but I have reduced the number of bins to 8 and also the number of samples has been decreased significantly. I am now able in Fig. 4.18 to compute both Hybrid and Quantum Annealing, and to compare them with IBU and matrix inversion.

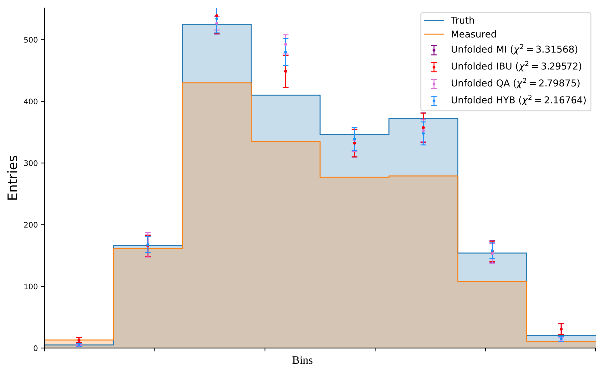


Figure 4.18: This is at the moment the best I can obtain with the total quantum annealing, for all the reasons previously explained. I note that the two quantum methods are better than the classical ones.

Conclusions

In this work we presented a new way of solving the Unfolding problem based on the Quantum Annealing technique. We have then introduced *QUnfold*, a Python package that implements three annealing methods to solve the unfolding: Simulated, Hybrid and Quantum Annealing. These methods rely on the popular quantum computer freely accessible to users: D-Wave's Quantum Annealer. However, despite in these last years significant progress have been made, the total quantum annealing is not able to solve complex optimization problems yet. Therefore, the aim of this work is to test these three methods provided by *QUnfold* thanks to D-Wave's systems in order to see the advantages given by this new approach to the unfolding problem, and to see also the current limitation of this framework. From the analysis carried out in Chapter 4, we have obtained the following results:

- the unfolding problem can be correctly written in terms of a QUBO problem, the only category solvable by QA. In fact, once having found the best regularization parameter λ for the problem, in our case $\lambda = 0$, we find values for the χ^2 consistent with the ones of the classical methods. To be more precise the values are reported in the Table

Distribution	**IBU χ^2**	**QUBO χ^2**
$Cos(\theta)$	0.00065	0.01759
Mass	4.06894	0.00351

- The solution to the Simulate Annealing, i.e. the classical implementation of the annealing, provided by *QUnfold* still have some problems. The approach is correct and visually the unfolding works correctly, but there are some statistical fluctuation dependencies that increase the chi square value and the error of the unfolded distribution. Despite that we see that increasing the parameter N_R decreases the χ^2, slowly approaching the value given by the classical solver of the QUBO problem. This suggest that by working on the stability of the method *Qunfold* developer could improve a lot the solution.

- The hybrid solver, which combines the advantages of quantum and classical annealing algorithms, has another kind of limitation. In fact it is internally managed by D-Wave, and the user cannot control the number of reads. While the simulated annealing with its limitation gives a consistent solution, with a $\chi^2 = 46.09381$, the Hybrid solver finds a wrong unfolding solution, as it is clear from the χ^2 of over 19000 and the visual solution of Fig. 4.17.

- In the end we have tested also the quantum annealing method. As we could not perform it with the data of the $t\bar{t}$ distribution for their complexity, we have run a Monte Carlo simulation of 2500 samples and 8 bins. The unfolding did not work precisely because of the small quantity of data, but comparing the chi square we see that HYB and QA method works better than the classical IBU and MI. We report this data in the Table.

Unfolding method	χ^2
Matrix inversion	3.31568
Iterative Bayesian Unfolding	3.29572
Hybrid Annealing	2.79875
Quantum Annealing	2.16764

By these result, we can conclude that the QUBO formulation of the unfolding problems gives better result than the other classical method commonly used to solve the unfolding. However, the current implementations for the annealing (i.e. SA, HYB, QA) needs to be improved. They have too much dependency on statistical fluctuation and are not able to give for complex problems reliable solutions. In particular, QA cannot deal with complex problems due to the limitedness of the D-Wave machine.

This work has been of great importance to improve the development of *QUfold* tool, underling limits, possible issues and points to be improved. My work has been performed in close and continuous contact with the developers that are working on the code. Possible developments of my work could be the improvement of the statistical stability and of the code efficiency, also on sight of the continuous improvements of the D-Wave system that will allow in the future to handle a greater quantity of data and reading sample, improving the process presented here and making it a more challenging.

Bibliography

[1] Richard Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, June 1982. S2CID 124545445; DOI:10.1007/BF02650179.

[2] Tadashi Kadowaki and Hidetoshi Nishimori. Quantum annealing in the transverse ising model. *Phys. Rev. E*, 58:5355–5363, Nov 1998. DOI: 10.1103/PhysRevE.58.5355; https://link.aps.org/doi/10.1103/PhysRevE.58.5355.

[3] Fock V. Born M. Beweis des adiabatensatzes. *Zeitschrift für Physik*, March 1928. DOI: 10.1007/BF01343193.

[4] Dutta Amit Chakrabarti Bikas Rajak Atanu, Suzuki Sei. Quantum annealing: an overview. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, December 2022. DOI: 10.1098/rsta.2021.0417.

[5] R. Allgood Nicholas. A tour of adiabatic quantum computing-from quantum computing [working title]. *IntechOpen*, September 2023. DOI: 10.5772/intechopen.1002604.

[6] D-Wave System Documentation. Getting started with d-wave solvers. `https://docs.dwavesys.com/docs/latest/doc_getting_started.html`https://docs.dwavesys.com/docs/latest/doc$_getting_started.html$.

[7] M. P. Vecchi S. Kirkpatrick, C. D. Gelatt Jr. Optimization by simulated annealing. *Science*, 220:671–680, May 1983. DOI:10.1126/science.220.4598.671.

[8] Rasson J.-P. Granville V., Krivanek M. Simulated annealing: a proof of convergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):652–656, 1994. DOI:10.1109/34.295910.

[9] R. Shankar. *Principles of Quantum Mechanics.* Springer, 2012.

[10] Doi T. et al Irie H., Liang H. Hybrid quantum annealing via molecular dynamics. *Scientific Reports*, April 2021. DOI:10.1038/s41598-021-87676-z.

[11] Spano Francesco. Unfolding in particle physics: a window on solving inverse problems. *EPJ Web Conf.*, 55:03002, 2013. DOI:10.1051/epjconf/20135503002.

[12] G. Cowan. *Statistical data analysis.* Oxford University Press, USA, 1998.

[13] Giulio D'Agostini. Improved iterative bayesian unfolding. *arXiv: Data Analysis, Statistics and Probability*, 2010. https://api.semanticscholar.org/CorpusID:18359350; DOI: 10.48550/arXiv.1010.0632.

[14] Lydia Brenner, Rahul Balasubramanian, Carsten Burgard, Wouter Verkerke, Glen Cowan, Pim Verschuuren, and Vincent Croft. Comparison of unfolding methods using roofitunfold. *International Journal of Modern Physics A*, 35(24):2050145, 2020. DOI: 10.1142/S0217751X20501456.

[15] Mark Thomson. *Modern particle physics.* Cambridge University Press, New York, 2013.

[16] G. Bianco. Study of the quantum interference between singly and doubly resonant top-quark production in proton-proton collisions at the lhc with the atlas detector. Master's thesis, University of Bologna and INFN, December 2020. DOI: 10.13140/RG.2.2.34038.42561.

[17] G. Giacomelli S. Braibant and M. Spurio. *Particelle e interazioni fondamentali: Il mondo delle particelle.* UNITEXT. Springer Milan, 2010.

[18] R. P. Feynman. Space-time approach to quantum electrodynamics. *Phys. Rev.*, 76:769–789, Sep 1949. 10.1103/PhysRev.76.769.

[19] CERN(Conseil Européen pour la Recherche Nucléaire). URL: https://home.cern/.

[20] Makoto Kobayashi and Toshihide Maskawa. CP-Violation in the Renormalizable Theory of Weak Interaction. *Progress of Theoretical Physics*, 49(2):652–657, 02 1973. DOI: 10.1143/PTP.49.652; eprint = https://academic.oup.com/ptp/article-pdf/49/2/652/5257692/49-2-652.pdf.

[21] Ulrich Husemann. Top-Quark Physics: Status and Prospects. *Prog. Part. Nucl. Phys.*, 95:48–97, 2017. DOI: 10.1016/j.ppnp.2017.03.002.

[22] Wittek Peter Cormier K., Di Sipio R. Unfolding measurement distributions via quantum annealing. *Journal of High Energy Physics*, 2019(11):128, nov 2019. DOI: 10.1007/JHEP11(2019)128; URL: https://ui.adsabs.harvard.edu/abs/2019JHEP...11..128C.

[23] Gianluca Bianco and Simone Gasperini. QUnfold, jul 2024. https://github.com/JustWhit3/QUnfold; https://zenodo.org/badge/latestdoi/652649177.

[24] Negre C. Krakoff B., Mniszewski S. Controlled precision qubo-based algorithm to compute eigenvectors of symmetric matrices. *PLOS ONE*, 17(5):1–15, 05 2022. DOI: 10.1371/journal.pone.0267954.

[25] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. https://doi.org/10.1038/s41586-020-2649-2.

[26] Tao Han, Matthew Low, and Tong Arthur Wu. Quantum entanglement and bell inequality violation in semi-leptonic top decays, 2023. https://arxiv.org/abs/2310.17696.

[27] Christian Bierlich, Andy Buckley, Jonathan Butterworth, Christian Holm Christensen, Louie Corpe, David Grellscheid, Jan Fiete Grosse-Oetringhaus, Christian Gutschow, Przemyslaw Karczmarczyk, Jochen Klein, Leif Lönnblad, Christopher Samuel Pollard, Peter Richardson, Holger Schulz, and Frank Siegert. Robust independent validation of experiment and theory: Rivet version 3. *SciPost Physics*, 8(2), feb 2020. DOI: 10.21468/scipostphys.8.2.026.