

CERN

SUMMER STUDENT 2023

PROJECT REPORT

n_TOF Transport Code Update and RF Deconvolution

Author:

Tanguy CAVAGNA

Department:

SY-STI-BMI

Supervisor(s):

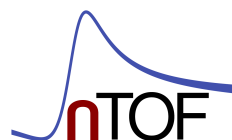
Jose Antonio PAVON

RODRIGUEZ

Francisco GARCIA

INFANTES

August 31, 2023



Abstract

The *Transport Code* constitutes a software tool to process outputs from Monte-Carlo simulation in neutron physics research, specifically catering to n_ToF. This software generates diverse histograms pertinent to neutron physics investigations. An essential feature of the *Transport Code* is its capability to derive the Resolution Function (RF), a crucial component for accurately determining the true energy of neutrons. This is imperative due to the convolution of neutron energies by the neutron generation process and subsequent requirement for meticulous processing to extract authentic energy values.

This scientific report revolves around the comprehensive advancement of the *Transport Code* framework. The advancements encompass a spectrum of activities, ranging from migrating the graphical user interface (GUI) from Python 2.7.5 to 3.6.8, to substantial enhancements in the underlying C++ codebase governing the optical simulation tool.

Furthermore, this report elucidates an antecedent exploration involving the fusion of Machine Learning and neutron physics. Specifically, the application of autoencoders to implement transpose convolutions (deconvolutions) on data from detectors is discussed. The outcome of this endeavor is deliberated upon, alongside the progression achieved through the evolution of the model architecture, transitioning from an autoencoder approach to a multi-layer Convolutional Neural Network (CNN) design.

Notably, the outcome of these codebase enhancements is the attainment of results identical to those produced by the legacy code, albeit with the advantages of contemporary robustness and maintainability. Concurrently, the Machine Learning investigation exhibits promising potential through the proposed model, pending exhaustive testing and validation, for practical implementation in real-world scenarios.

Contents

1	Introduction	5
1.1	The n_TOF facility at CERN	5
1.2	Time of flight technique (TOF) and Resolution Function (RF) . . .	7
1.3	Transport Code	9
1.3.1	Execution	10
1.4	Goals	11
I	Transport Code	13
2	Code porting and refactoring	14
2.1	Porting	14
2.2	Refactoring	14
2.3	Makefile upgrade	15
2.4	Simulation code structure	15
II	Machine learning	18
3	RF deconvolution	19
3.1	Previous attempt	20
3.1.1	Results	23
3.1.2	Analysis	25
3.2	Current attempt	26
3.2.1	Results	28
3.2.2	Analysis	32
3.3	Conclusion	33
4	Acknowledgement	35

List of Figures

1.1	n_TOF facility layout [1]	6
1.2	Energy distribution of the flux in EAR1, for the configurations with normal water (before 2009, in black) and with borated water (after 2009, in red). For comparison, the evaluated flux of EAR2 is also plotted. [1]	8
1.3	Sequence Diagram of the cluster	10
1.4	Class Diagram of the GUI	11
2.1	Transport Class Diagram	16
3.1	datagenerator structure	20
3.2	<i>Transport Code</i> output of the parameters of table 3.1	21
3.3	Random sample taken out of the datagenerator. Red is from <i>input.csv</i> and black from <i>output.csv</i> . One can see the clear difference between the convoluted sample and the original (<i>Ground-truth</i>). . .	22
3.4	Autoencoder architecture	23
3.5	Random sample taken from the generated data. <i>Ground truth</i> being the original signal, <i>Input</i> the convoluted signal, and <i>Output</i> the deconvoluted signal from the CNN model.	24
3.6	Results of the CNN model when multiple peaks are present in the sample. One can see that the model cannot predict properly the clean signal, and even creates artifacts where no peaks are present. .	25
3.7	New CNN model architecture	26
3.8	<i>Transport Code</i> output for the new attempt of RF deconvolution for EAR2 with a flight path of 19.8m	27
3.9	Random samples taken out from the generated data. As one can see, single and multiple resonance peak are well fitted and the residue in all cases are below 4%.	29
3.10	Random samples taken out from the generated data. Same conclusion as Figure 3.9.	30

3.11	Random samples taken out from the generated data. Even with very noisy signals, the prediction still manage to follow the ground truth (black line) without diverging.	31
3.12	Random samples taken out from the generated data. The top sample as a high level of noise which is not retrieved by the deconvolution. The bottom sample is well fitted, apart from the left edge peak which have $\sim 16\%$ of error.	32

List of Tables

3.1	Parameters of the RF simulation	20
3.2	Parameters used for the data generator	21

Chapter 1

Introduction

The present Summer Student project was conducted within the n_TOF team, with a specific focus on the study, enhancement, and adaptation of the Transport Code software. Additionally, the project encompasses the development of a Convolutional Neural Network (CNN) model aimed to disentangling neutron's ToF to real energy conversion, which is a very time consuming process and produce a lot of headaches. Consequently, it becomes imperative to establish a foundational comprehension of both the n_TOF experiment and the Transport Code software.

1.1 The n_TOF facility at CERN

The neutron Time Of Flight facility, n_TOF, was established in 2001 at the European Organization for Nuclear Research (CERN), is a pulsed neutrons source designed for high resolution neutron cross section measurements over a wide energy range, spanning from few milielectronvolts (meV) to several giga-electronvolts (GeV).

At the n_TOF facility, neutron production relies on spallation reactions driven by a pulsed proton beam with a momentum of 20GeV/c colliding with a substantial lead target. The proton beam, accelerated to 20 GeV by the Proton Synchrotron (PS) accelerator complex, has a temporal span of 7-14 ns RMS depending on the phase. The resulting high-energy spallation neutrons undergo moderation, transforming into a broad spectrum ranging from several GeV down to thermal energy levels. This neutron beam undergoes initial collimation, followed by the immediate application of a magnet to sweep away charged particles. A second collimator is subsequently employed to shape the neutron beam's final configuration.

As can be seen in the layout of the facility in Figure 1.1, at n_TOF there are two

Experimental Areas (EAR): EAR1 was the first one operative, with a flight path length of 184 m; and EAR2, commissioned in 2014, which is situated vertically on top of the target, and has a flight path of 19 m. The shorter flight path results in a neutron flux 30 times higher than in EAR1, but comes at the expense of a lower neutron energy resolution, and of an increased background compared to EAR1. Hence, both areas have complementary features, which make each one of them suited for different experiments. As it shown in Figure 1.2.

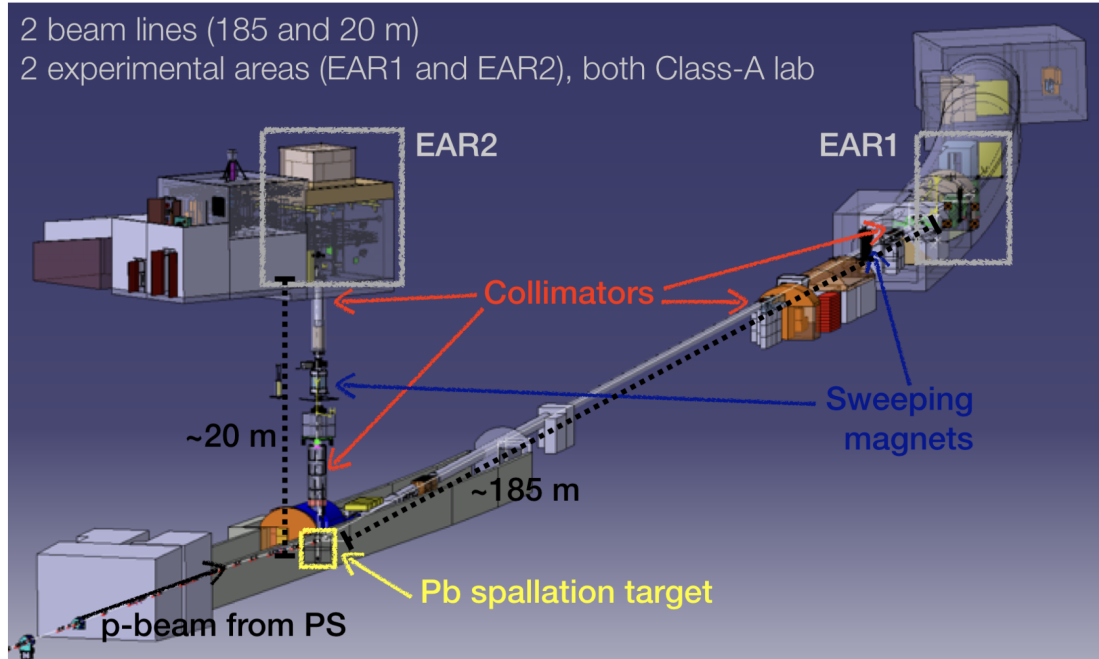


Figure 1.1: n_TOF facility layout [1]

The n TOF facility possesses a combination of features that collectively establish its exceptional suitability for cross-section measurements:

- The neutron beams produced by n_TOF exhibit an extraordinarily high instantaneous flux, rendering them particularly well-suited for scenarios necessitating a robust signal-to-background ratio. This attribute proves especially advantageous for measurements involving small quantities of radioactive samples.
- The elevated flux effectively compensates for an average repetition rate of 1.2 Hz, a characteristic governed by the operational cycle of the PS accelerator. Moreover, the low duty cycle of the facility offers its own advantages. It permits the execution of measurements across an extensive spectrum of

neutron energies, ranging from several hundred MeV down to the thermal point, all without encountering overlap between neutron pulses.

- In the instance of n_TOF EAR1, the implementation of an extended flight path contributes to the achievement of remarkable neutron energy resolution. Specifically, the facility can attain a $\frac{\Delta E}{E}$ ratio lower than or equal to 10^{-3} up to 10 keV, and less than $5 \cdot 10^{-3}$ up to an energy of 100 keV.

There are only a handful of time-of-flight facilities across the globe, each possessing its unique attributes. The standout feature of n_TOF is its capability to span a wide energy range and generate a substantial number of neutrons per pulse. The data generated by n_TOF plays a crucial role in astrophysics, aiding in the exploration of stellar evolution and supernovae phenomena. Moreover, powerful neutron beams find significance in both hadrontherapy (a method for tumor treatment using beams of hadrons) and the investigation of radioactive nuclear waste management. [2].

1.2 Time of flight technique (TOF) and Resolution Function (RF)

At a time-of-flight facility like n_TOF, the energy of the incoming neutrons is determined by measuring the time t they take to travel a fixed and well-known distance L . In our particular case, L is the distance from the target to the experimental areas. In the classical kinematics approximation, valid up energies of a few MeV, the kinetic energy E_n of a neutron travelling along L with speed v is given by the expression:

$$E_n = \frac{1}{2}m_n v_n^2 = \frac{1}{2}m_n \frac{L^2}{t_{TOF}^2} \quad (1.1)$$

where t_{TOF} is the time of flight in μs , E_n is the neutron energy in eV and L is the distance in m. For a long distance L the t_{TOF} will be larger, which is a smaller relative error in the measurement for both quantities. Therefore, a better neutron energy resolution will be achieved. This time (ToF) is computed as follow:

$$t_{TOF} = t_d + t_{flash} + \frac{L}{C} \quad (1.2)$$

where t_d is the detection time, t_{flash} is the starting time given by the gamma flash created by the collision of proton-nuclei in the target, and $\frac{L}{C}$ is the time that gamma take to arrive from the target to the detector on a path of length L .

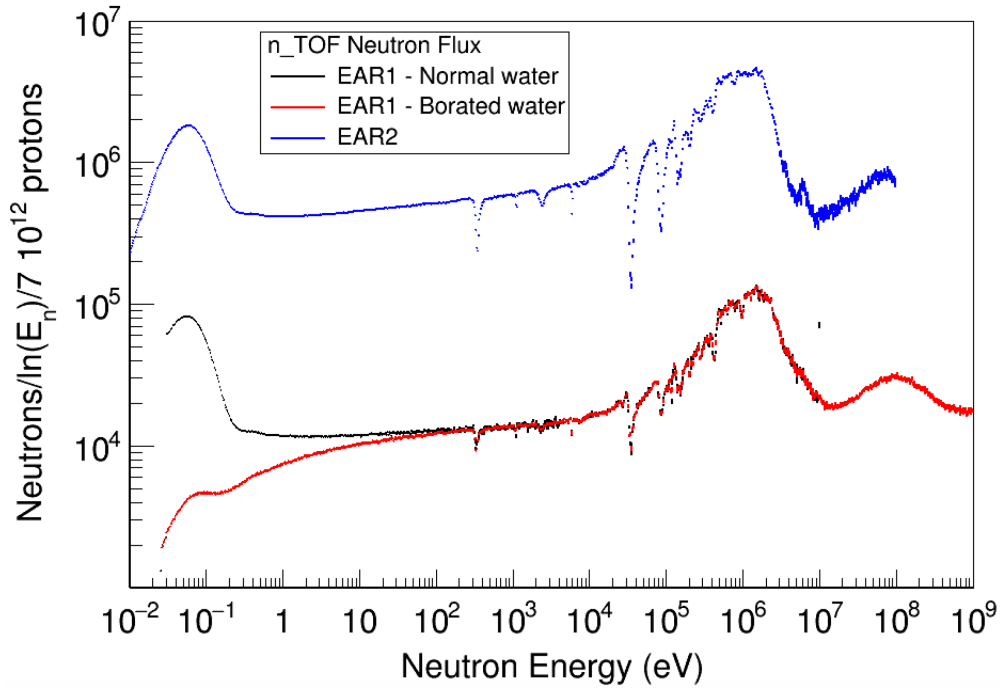


Figure 1.2: Energy distribution of the flux in EAR1, for the configurations with normal water (before 2009, in black) and with borated water (after 2009, in red). For comparison, the evaluated flux of EAR2 is also plotted. [1]

The correspondence between the energy of the neutron and its time of arrival, however, is not a one-to-one relation. Neutrons with the same energy can arrive at different times, mainly because of the different paths they follow inside the spallation target and moderator until they arrive at the measuring point. Specifically, several experimental conditions contribute to this effect. First of all, the width of the proton pulses of the PS, equal to 7-14 ns RMS depending on the phase, which is in fact the absolute upper limit in the time resolution of the facility. In second place, the interaction of the neutrons with the different elements of the target-moderator assembly, and, finally, the time resolution of the detectors. The most important of them, however, is the moderation process. This effect is quantified in the moderation length, which is the distance that neutrons arriving with a given energy have travelled in the moderator.

Quantifying the energy distribution associated with neutrons proves to be a notably challenging endeavor, substantiating the necessity for the application of the Resolution Function (RF). The Resolution Function, denoted as $R(t, E)$, delineates the distribution of neutron time-of-flight t , within a fixed length L , corresponding to a specific kinetic energy E . As mentioned before, deviations in time-of-flight

can stem from diverse experimental influences, including the temporal spread of the primary particle beam, moderation duration, and the geometric configuration of the beam line projection.

Considering the configuration of the n_TOF facility, which encompasses two distinct experimental regions (cf. Figure 1.1), it becomes imperative to factor in the presence of two distinct Resolution Functions (RF). [3]. However, the RF cannot be measured directly. Therefore, it is obtained by means of MC simulations, which include a fully detailed description of the target and moderator geometry. In these simulations, performed with FLUKA code, each neutron produced in the spallation process is followed in their path through the target and moderator. However, propagating the neutrons from there, up to the experimental hall – 185 m away – by MC means would be totally impractical due to the computational time required. Thus, particle trajectories are calculated with a dedicated optical transport code.

1.3 Transport Code

The *Transport Code* is an optical transport routine, coded in C++, designed to project neutron positions from one surface to another considering a clear path between them so that nothing can affect neutron tracks. Assuming that the angular distribution is isotropic within an angle θ , neutron are projected to a surface located at a distance L from the original place with the exception of the neutrons that are outside the circle defined by $L_0 \cdot \tan(\theta) - \rho$, where ρ is the aperture of the first collimator in the line in order to remove the beam halo created from an extended source (i.e. entering the collimator from the side instead of the dedicated windows). Neutrons trajectories are from the original scoring plane which holds the information from simulations, to every point in the new surface. All the trajectories that do not fulfill the collimation conditions are discard.

The *Transport Code* serves as a valuable tool for deriving the requisite parameters necessary to generate an accurate Resolution Function (RF) tailored to a specific experimental scenario. Starting with default parameters such as the initial flight path, properties of the sample, and evaluated cross-sections, this input information enables a comparison between the experimental count rate and the anticipated count rate derived from simulation. Through a juxtaposition of these data sets, disparities in outcomes attributable to factors like incorrect flight paths, sample misalignment, or sample mass discrepancies can be discerned.

Given the potential for these sources of error, the input parameters are systematically refined to align more closely with the experimental observations. As demonstrated, this input optimization unfolds as an iterative process. Once the iterative

sequence converges – marked by the alignment of the anticipated count rate with the experimental counterpart in the Time-Of-Flight (TOF) domain – the definitive set of parameters is selected. These refined parameters subsequently serve as the foundational input for establishing the RF. [4]

1.3.1 Execution

The utilization of the *Transport Code* is not carried out on individual machines due to the potential excessive CPU burden and the necessity for constant machine operation. To address these challenges, a cluster, known as lxplus, using HTCondor as job scheduler, is made accessible for the purpose of running diverse job types, with a particular focus on simulations. In summary, the procedure for utilizing the Transport Code unfolds as follows:

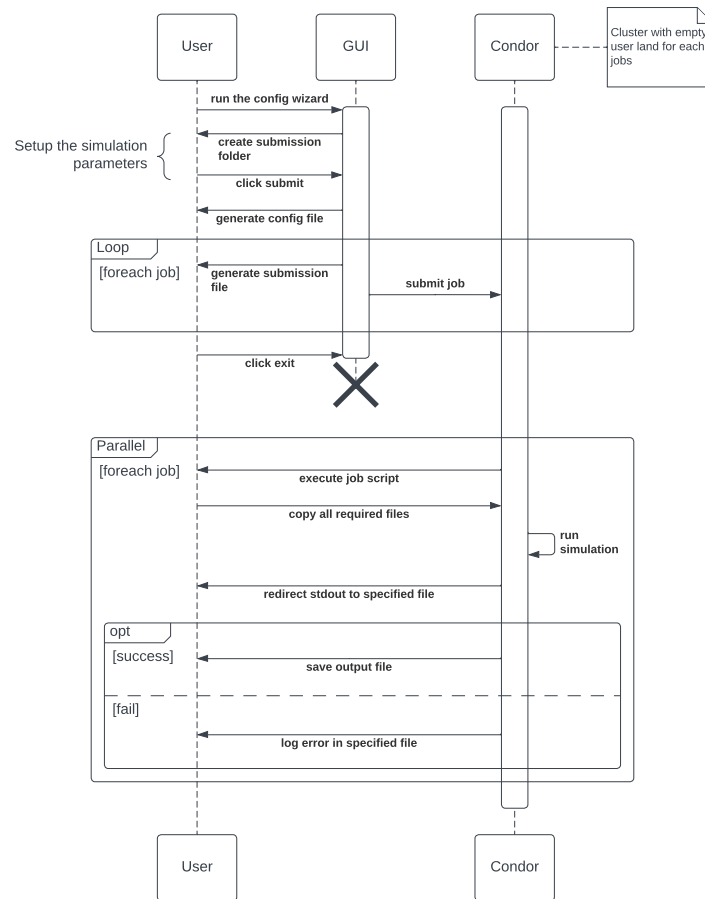


Figure 1.3: Sequence Diagram of the cluster

GUI

The Graphical User Interface (GUI) functions as a purposeful tool, primarily aimed at simplifying the intricate process of simulation setup. This process involves the meticulous configuration of a multitude of parameters. From defining scoring distances and energy cutoffs to fine-tuning histogram settings, a comprehensive range of considerations comes into play. Constructed using Python 2.7.5 and exclusively based on Tkinter [5], the GUI has been meticulously designed to fulfill a specific objective.

This objective revolves around the creation of a collection of precisely parameterized Bash files. These files play a pivotal role in submitting tasks to a distributed cluster of machines, effectively launching the simulation process. Significantly, the GUI also offers real-time monitoring of job statuses, ultimately culminating in the generation of ROOT [6] files. These ROOT files serve as essential components for visualizing and plotting the results derived from the simulation process.

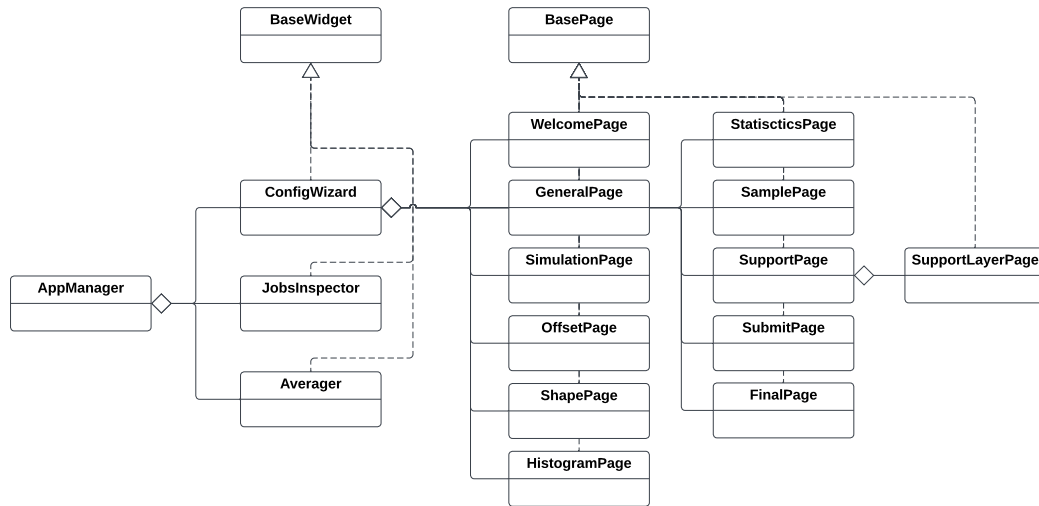


Figure 1.4: Class Diagram of the GUI

1.4 Goals

The Summer Student project encompasses a range of key objectives. Firstly, achieving an exhaustive comprehension of the existing codebase is paramount, as it serves as the foundational knowledge base for implementing subsequent goals. The second objective revolves around the migration of the Python 2.7-based *Transport*

Code to version 3.6 or 3.9 (the latest iteration available on lxplus depending on the lxplus version). Concurrently, an enhancement of the C++ codebase is intended, aimed at addressing potential edge cases while adopting a more streamlined approach. This enhancement involves the creation of a comprehensive overarching class to encapsulate simulation methods and properties, offering a coherent perspective on the simulation process.

Lastly, the project seeks to explore the integration of machine learning techniques to perform data deconvolution. This entails separating collected data (comprising both background and signal components) to extract the original signal which was convoluted with the distribution due to the experimental setup (see 1.2). Successful implementation of this aspect holds the potential to have a considerably cleaner analysis and obtain a true energy distribution directly out of the measurements.

Part I

Transport Code

Chapter 2

Code porting and refactoring

2.1 Porting

As indicated in Section 1.3.1, the GUI is constructed using Python 2.7.5. However, it's important to note that this version has exceeded its end-of-life duration. The most recent version available in the cluster is 3.6.8, which, while also surpassing its end-of-life, was developed after the major syntax changes introduced in Python 3.

The primary challenge lies in updating all imports within the GUI codebase. This is necessary due to shifts in the module structures of the employed libraries, leading to a disparity between the existing and accurate imports. An associated issue may arise from changes in the logic of these libraries, potentially impacting the expected workflow of the current GUI. Addressing this challenge may require further investigation to ensure the seamless resolution and maintenance of the desired functionality.

2.2 Refactoring

Refactoring the C++ code offers substantial advantages on several fronts. It provides a comprehensive grasp of the codebase, encompassing both its overall structure and finer details. This understanding empowers effective modifications and segregation of the code. However, approaching this task with a computer science background, rather than a physics one, introduces its own set of challenges.

Perhaps the most demanding yet intriguing aspect revolves around grasping the underlying physics principles embedded in the simulation and how they manifest in

the code. On the contrary, this challenge isn't the most time-consuming one. That distinction belongs to the actual process of refactoring. Indeed, revamping around 2500 lines of code and meticulously testing it to achieve the level of robustness demanded by research software is no straightforward feat.

Each modification necessitates thorough and repeated testing to validate its efficacy, and this process must be accompanied by comprehensive documentation. This requirement for precision and thoroughness underscores the exacting nature of the refactoring undertaking.

2.3 Makefile upgrade

Presently, the compiled objects are all situated within the same folder as the source files. While this straightforward approach is often favored and adopted by many, it proves advantageous in this context to segregate these objects from the sources. The rationale behind this decision is straightforward: by keeping binaries in a distinct "bin" folder, operations such as deletion, upgrading, deployment, or copying become more manageable and organized. Notably, the exception to this arrangement is the executable itself due to the existing user manual's specified path, which remains rooted at the folder's core.

Moreover, a noteworthy modification involves incorporating the `"-std=c++11"` argument during compilation to enforce the use of C++11 standards. This strategic addition at compile time ensures using the desired version of the C++.

2.4 Simulation code structure

The *Transport Code* simulation currently resides within a singular main file, supplemented by additional included tools. While this approach can yield efficient execution times through the minimization of context swapping, it inherently poses challenges in terms of code comprehensibility. Reading through the code and discerning the underlying processes becomes a formidable task. The motivation behind refactoring is to encapsulate all global variables utilized in the simulation within a dedicated class, thereby striving to identify segments of code amenable to translation into methods. Such a restructuring endeavors to enhance the code-base's development, maintenance, readability, and usability.

While, in some instances, this approach might marginally incur a minor slowdown due to context swapping, it maintains comparability when optimized with the `-O3` flag. For potential speed enhancements, employing profiling tools like `"perf"` `perf`

or "VTune" [7] to analyze and fine-tune the *Transport Code*'s performance could prove beneficial.

For the purpose of effective code separation, two new files named "transport-util.h" and "transport-util.cc" have been introduced to the root directory. These files incorporate pre-existing classes alongside a novel addition named **Transport**. This newly introduced class houses both simulation properties and methods distilled from the process of fragmenting the main file. Consequently, the code's structure has evolved into the subsequent form:

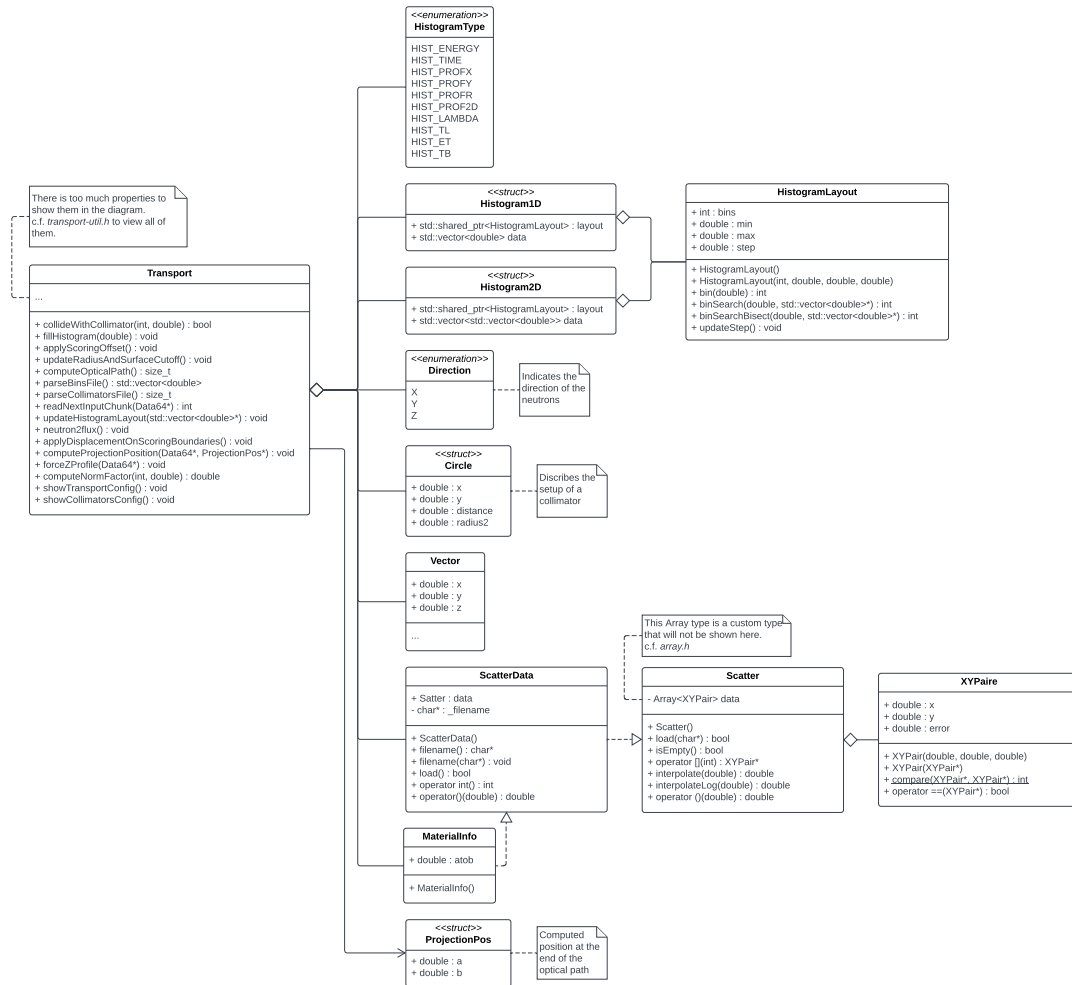


Figure 2.1: Transport Class Diagram

Moreover, generic files were present in the codebase, which yields a greater diffi-

culty to understand the code. Within those file, only a few of the function where actually used by the *Transport Code*. The said functions were extracted from the files and included in the codebase directly, which allowed the removal of multiple large files, and thus increased the code comprehension and maintenance.

Part II

Machine learning

Chapter 3

RF deconvolution

As elucidated in Section 1.2, the measurements encountered within the context of this study are influenced by the way in which neutrons are generated, thereby engendering distribution in relation to the time of flight measured, that extends to the ToF to energy conversion. These fluctuations can be characterized as a function that becomes convolved with the ToF measured. The ultimate objective of this project revolves around investigating the feasibility of employing machine learning to execute a transposed convolution (deconvolution).

Once the ToF to energy conversion is done, the problem at hand can be formally expressed as follows: Let $T(x)$ represent the true time-of-flight, while $d(x)$ signifies the ToF distribution introduced by the neutron generation process.

$$X(x) = (T * d)(x) \tag{3.1}$$

where $X(x)$ denotes the convolved outcome derived from the measurements. In light of this scenario, the objective is to develop a Convolutional Neural Network (CNN) model. The intention is to train this model in such a way that its predicted output $\hat{y}(x)$ accurately corresponds to $T(x)$.

$$\hat{y}(x) = \text{CNN}(X(x)), \quad \hat{y}(x) \propto T(x) \tag{3.2}$$

where $\hat{y}(x)$ represents the predicted time-of-flight, $\text{CNN}(\cdot)$ denotes the Convolutional Neural Network function, and $X(x)$ is the input data (convoluted result from measurements).

3.1 Previous attempt

In early 2023, an endeavor was undertaken Javier Balibrea to address this challenge by employing an autoencoder approach. The overarching concept revolved around creating two distinct files. The first file contained the pristine signal, as if each neutron was followed and the exact ToF was known, while the second file encompassed the convoluted distribution.

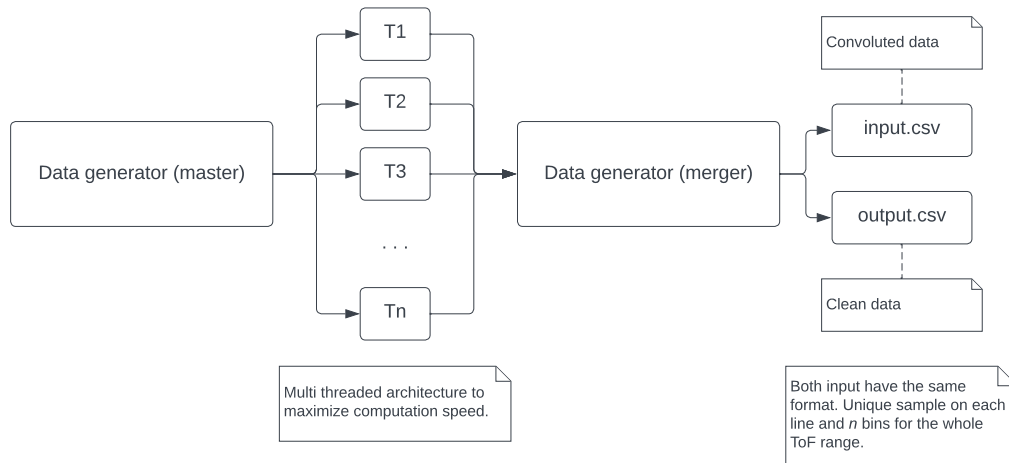


Figure 3.1: datagenerator structure

This generator operates on a simulated Resolution Function (RF) and is supplied with a designated Time-of-Flight (ToF) range for data generation, accompanied by a selection of additional parameters. The choice to not generate data across the entire RF span is rooted in the consideration that such a computation would prove exceedingly resource-intensive, leading to an unwieldy volume of output files. Moreover, opting for a smaller dataset facilitates the process of model fitting.

RF simulation parameters	
Parameter name	Value
Phase	4
EAR	2
Length	19.8[m]
Input source	P4 FLUKA 4.3
Histogram type	Lambda vs ToF

Table 3.1: Parameters of the RF simulation

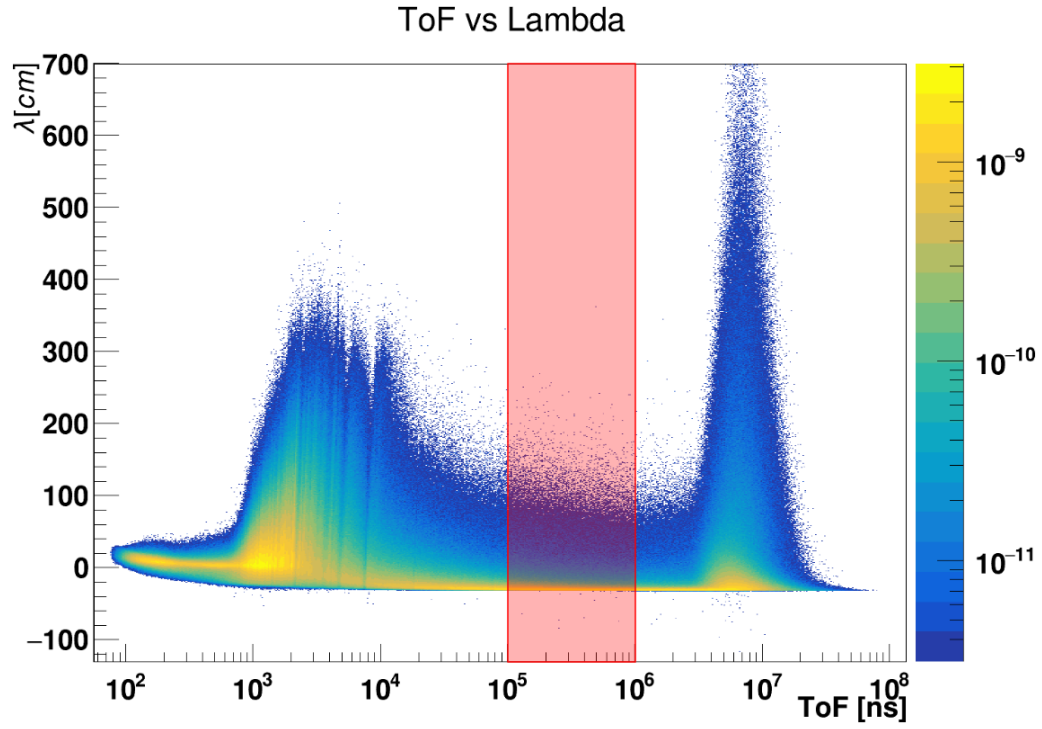


Figure 3.2: *Transport Code* output of the parameters of table 3.1

The RF is harnessed within the data generator to effectuate the convolution process on the pristine signal. Subsequently, data generation transpires with the utilization of the following parameters:

Data generator parameters	
Parameter name	Value
ToF minimum	10^5
ToF maximum	10^6
ToF distance	$19.8[m]$
Log binning	ToF
Bin count	2000
Event count	10000

Table 3.2: Parameters used for the data generator

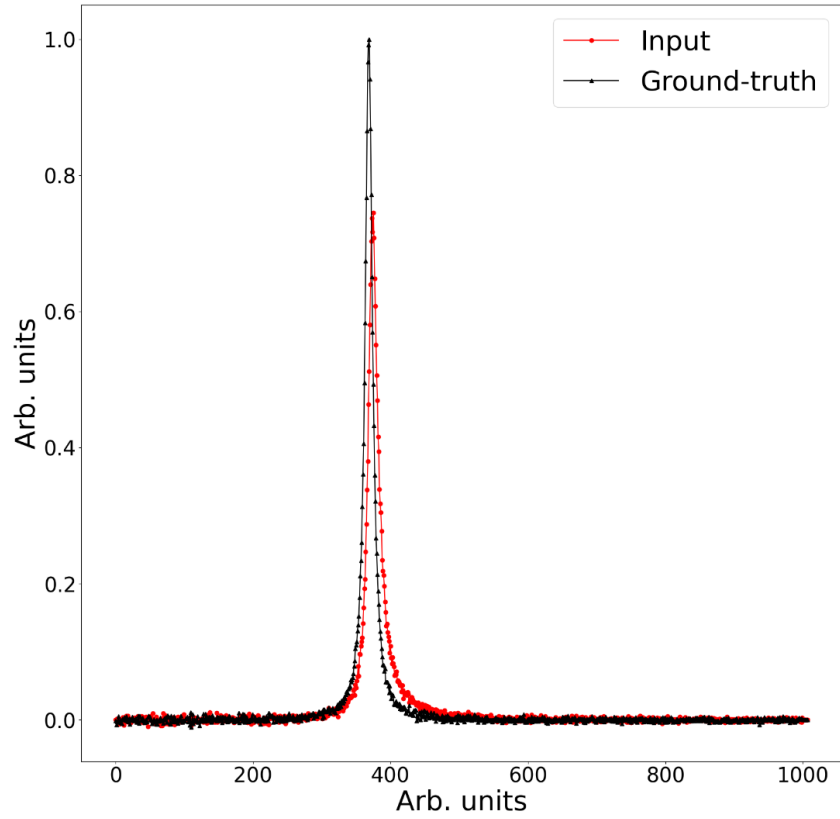


Figure 3.3: Random sample taken out of the datagenerator. Red is from *input.csv* and black from *output.csv*. One can see the clear difference between the convoluted sample and the original (*Ground-truth*).

Following the completion of data generation, the subsequent step involves model training. The design of the employed model was characterized by its simplicity, and its structure is illustrated as follows:

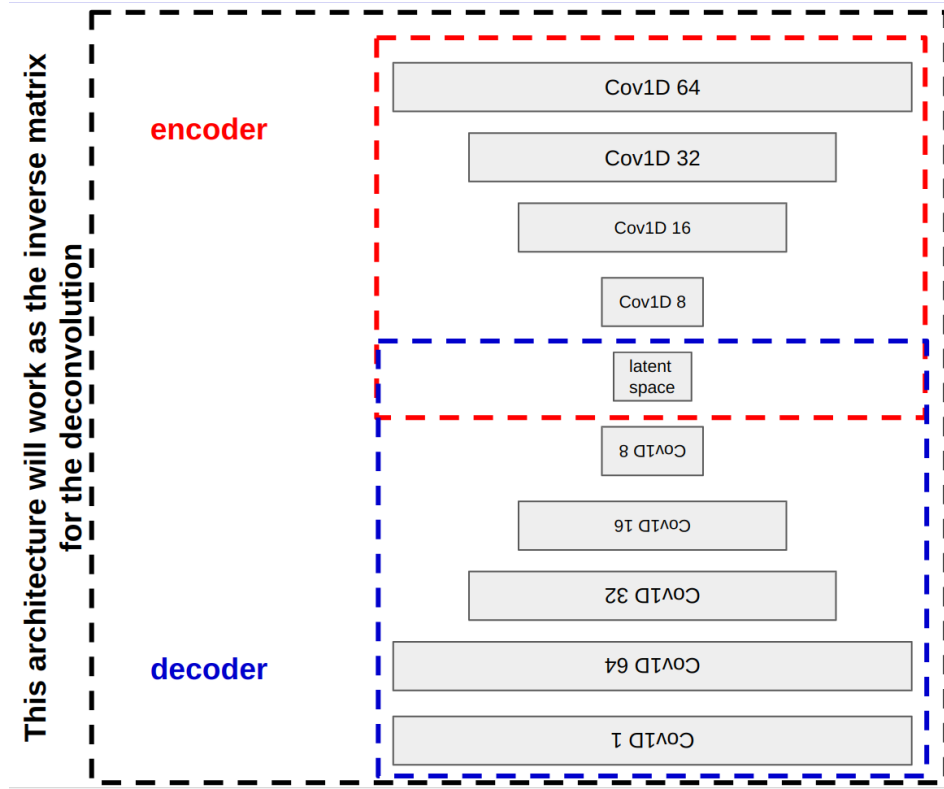


Figure 3.4: Autoencoder architecture

The decision to opt for an autoencoder was rooted in its typical application for signal noise reduction. This choice is guided by the notion that the distribution inherent in a measurement's signal can be approximated through the convolution of the authentic signal with a form of noise.

3.1.1 Results

The outcomes achieved were simultaneously promising in specific scenarios and less compelling in others. For instance, when considering a random sample characterized by a singular peak within the distribution, the resultant vector $\hat{y}(x)$ exhibited an error rate ranging between 4-8% as shown bellow. The residue formula is expressed as $R(x) = X(x) - \hat{y}(x)$.

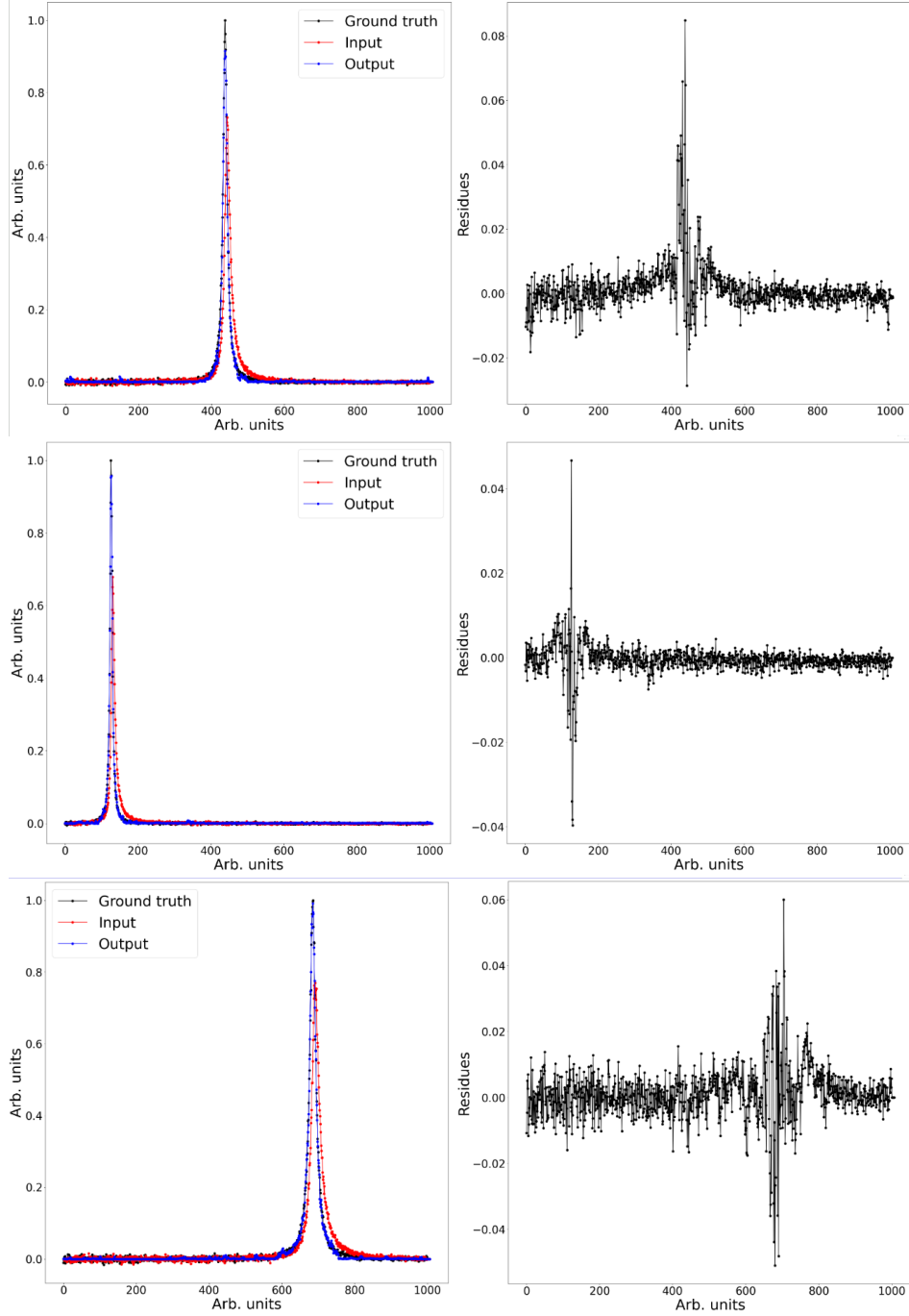


Figure 3.5: Random sample taken from the generated data. *Ground truth* being the original signal, *Input* the convoluted signal, and *Output* the deconvoluted signal from the CNN model.

However, when multiple peaks are present within the sample, the model encounters difficulties and produces an output that adheres to only one of these peaks.

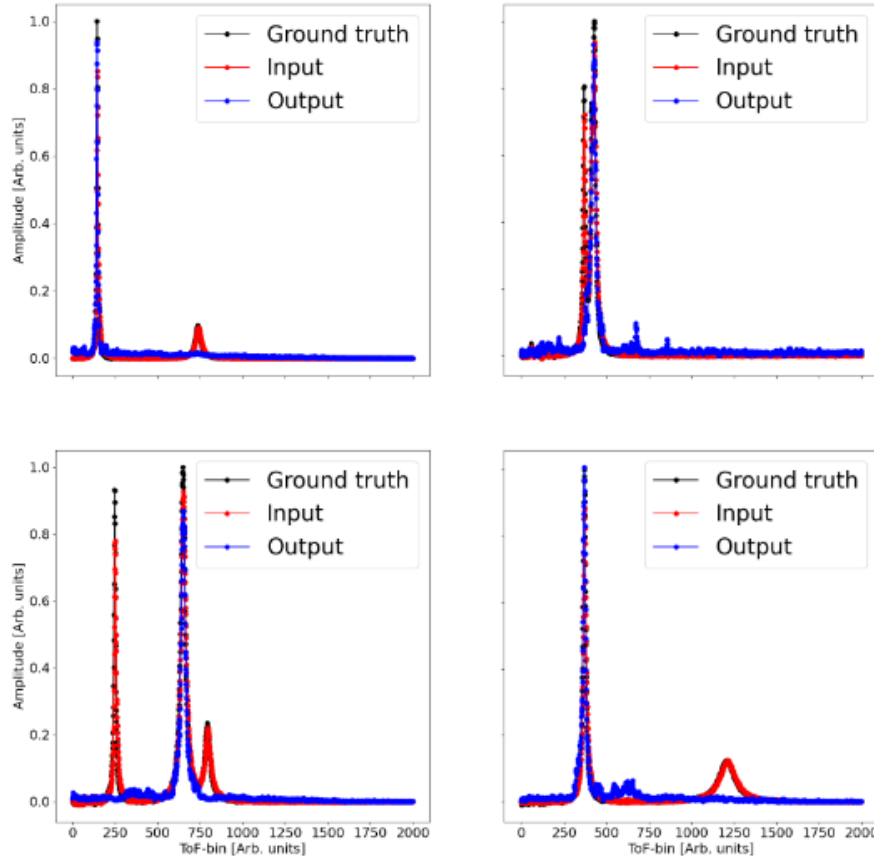


Figure 3.6: Results of the CNN model when multiple peaks are present in the sample. One can see that the model cannot predict properly the clean signal, and even creates artifacts where no peaks are present.

3.1.2 Analysis

Evidently, this model cannot be employed as a reliable tool due to its significant imprecision when confronted with samples exhibiting multiple peaks. This limitation, coupled with the limited available time for further development, essentially halted the pursuit of an improved solution. The Summer Student Programme provided a valuable opportunity to rekindle this research endeavor.

3.2 Current attempt

The attempt undertaken during the Summer Student Programme adheres to the same approach in terms of data generation. However, a notable alteration has been made in the realm of machine learning models. Instead of employing a convolutional autoencoder, a simplified alternative has been adopted – the utilization of a multilayer Convolutional Neural Network (CNN).

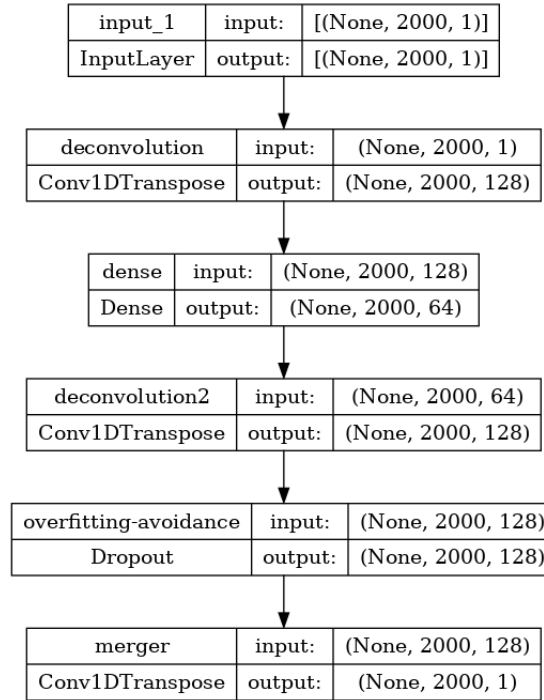


Figure 3.7: New CNN model architecture

The network architecture commences with the input data, which is presented as a 1D array containing around 2000 bins, contingent upon the number of bins generated. This input then flows into a transposed convolution layer, responsible for mapping the 1D array onto a 2000x128 matrix. Then, a dense layer is used before the following transposed convolution layer. Subsequently, a dropout layer follows, strategically integrated to avert any likelihood of overfitting. This is accomplished by randomly setting a proportion of weights to 0. Lastly, the network concludes with a merger layer, which takes the 2000x128 matrix and reconvenes it into a 1D array. This resultant array constitutes the prediction output of the CNN.

In relation to the dataset, an alternative approach has been employed for data segmentation. In this updated strategy, the dataset has been divided into three distinct subsets, as opposed to the previous two. The initial subset functions as the training set, containing 80% of the data. The subsequent subset is allocated for validation purposes during the fitting procedure, encompassing 10% of the data. Finally, the third subset serves as the testing set, utilized post-fitting to facilitate predictions. This testing subset, representing 10% of the data, is crucial for averting potential prediction biases that could emerge from the model's prior exposure to specific samples during the fitting phase.

$$\text{Training Set} : \text{Validation Set} : \text{Testing Set} = 8 : 1 : 1 \quad (3.3)$$

The RF simulation parameters are still the same as 3.1, and it produced the following RF.

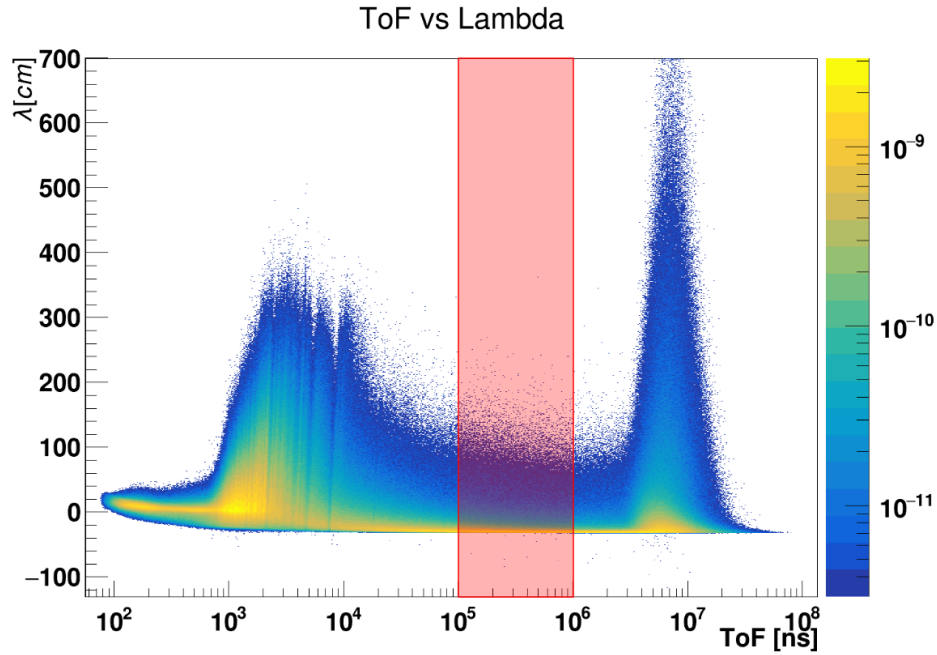


Figure 3.8: *Transport Code* output for the new attempt of RF deconvolution for EAR2 with a flight path of 19.8m

Finally, the loss function has been changed from the standard MSE, to the Huber function [8]:

$$a(x) = |\hat{y}(x) - X(x)| \tag{3.4}$$

$$\mathcal{L}_\delta(a_i) = \begin{cases} \frac{1}{2}a_i^2 & \text{for } |a_i| \leq \delta, \\ \delta \cdot \left(|a_i| - \frac{1}{2}\delta\right), & \text{otherwise.} \end{cases} \tag{3.5}$$

This function is quadratic for small values of a , and linear for large values, with equal values and slopes of the different sections at the two points where $|a_i| = \delta$, and δ is a hyperparameter that controls the threshold between the quadratic and linear regions of the loss function.

3.2.1 Results

The achieved outcomes are very promising. Similar to the previous model, samples characterized by a single peak exhibit remarkable precision. However, a notable advancement lies in the model's capacity to predict samples featuring multiple peaks with a notably higher degree of accuracy. The residue formula is expressed as equation 3.4.

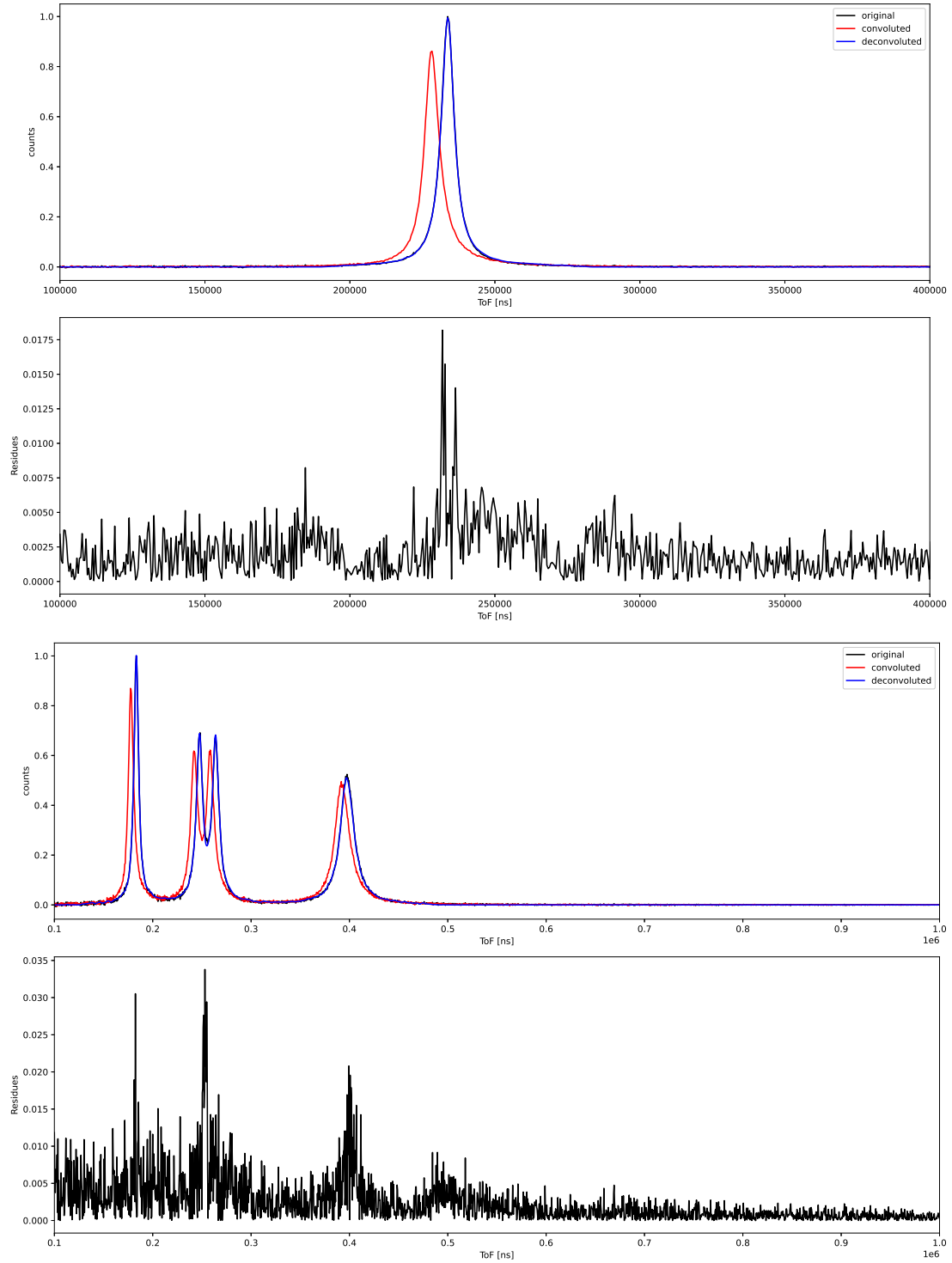


Figure 3.9: Random samples taken out from the generated data. As one can see, single and multiple resonance peak are well fitted and the residue in all cases are below 4%.

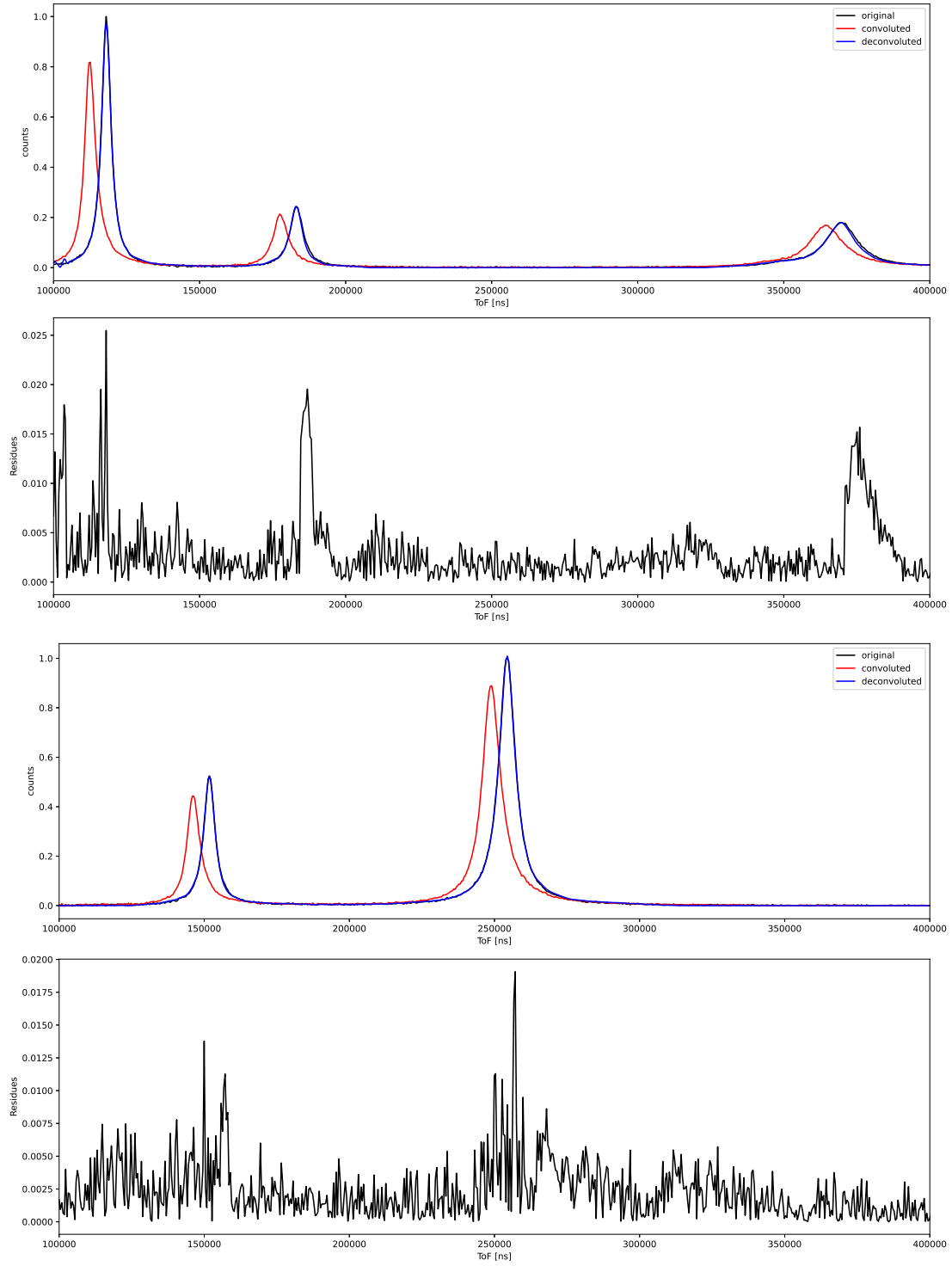


Figure 3.10: Random samples taken out from the generated data. Same conclusion as Figure 3.9.

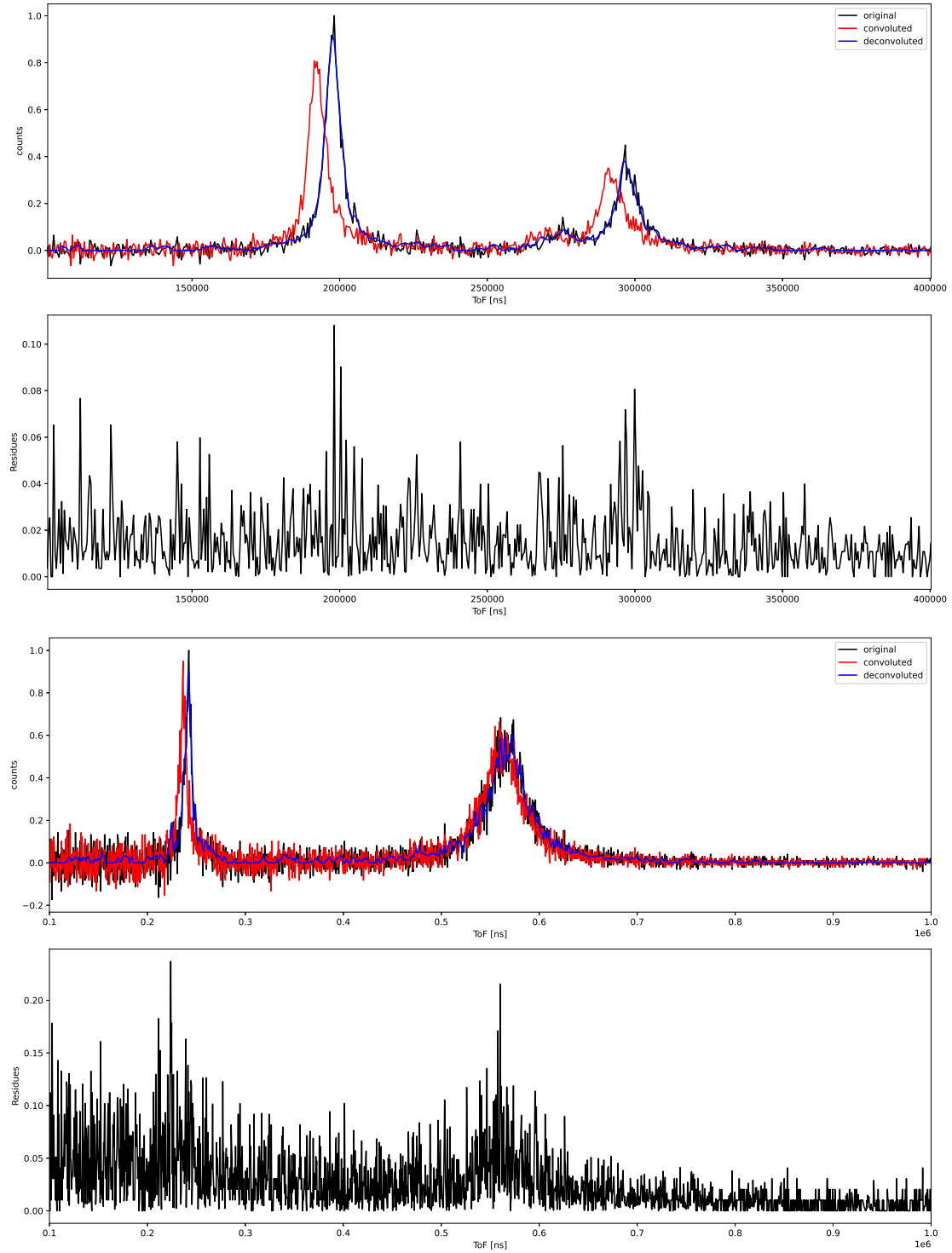


Figure 3.11: Random samples taken out from the generated data. Even with very noisy signals, the prediction still manage to follow the ground truth (black line) without diverging.

Conversely, instances of imprecision arise when the signal is situated near the end of range of the trained model or when the sample contains substantial levels of noise.

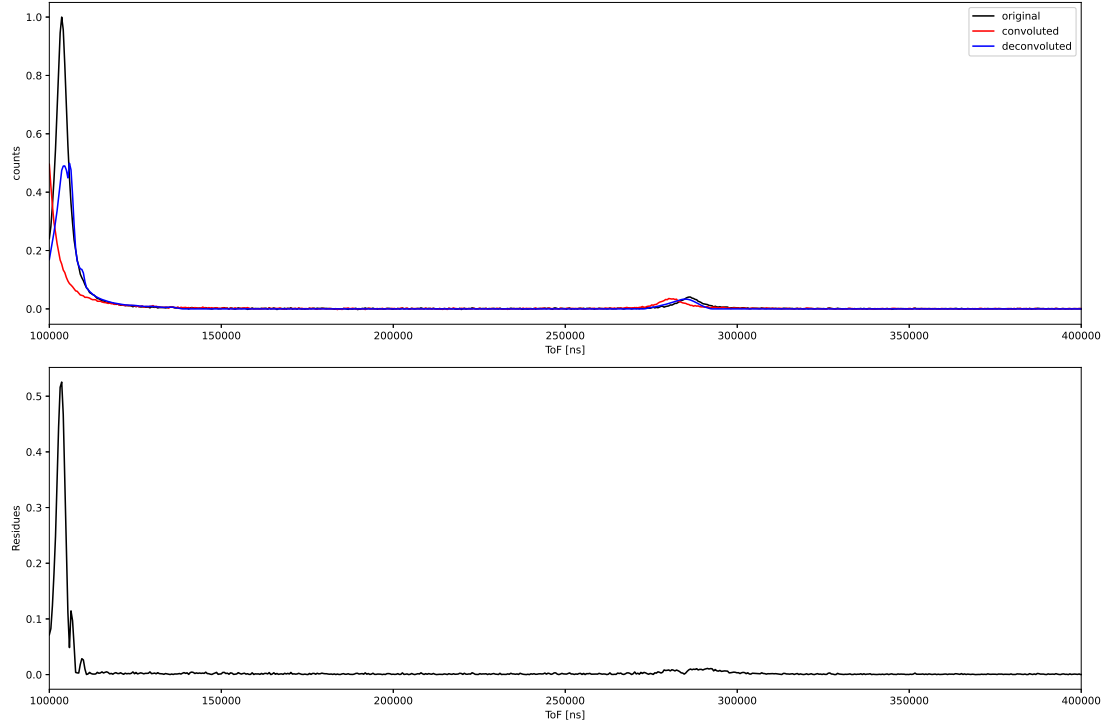


Figure 3.12: Random samples taken out from the generated data. The top sample as a high level of noise which is not retrieved by the deconvolution. The bottom sample is well fitted, apart from the left edge peak which have $\sim 16\%$ of error.

3.2.2 Analysis

The obtained results exhibit significant improvement in comparison to the earlier attempt. The challenge associated with multiple peaks has largely been mitigated; however, precision diminishes when the distribution closely approaches the end of range of the trained model. Additionally, as one can see, when a significant amount of noise is introduced, the fitting is smoother and do not take into account the noise. This can be an issue if the noise is an important feature.

Notably, the models are trained on relatively clean signals without consequent noise. Moreover, each model is made for a particular ToF range, and cannot be used to predict samples from another one. This involves training models for each ToF range, incorporating a degree of overlap to counteract range limitations.

Finally, thorough testing was made for the model targeting the $10^5 - 10^6$ ToF range to validate the performances. By changing various parameters of the data being fed into the model to be deconvoluted, the model kept performing very well and the predictions remained very good.

3.3 Conclusion

This report embarked on a comprehensive journey involving the modification and enhancement of the existing optical simulation code at n_ToF. The central aim was to enhance the codebase's accessibility, promote a better understanding among potential future maintainers, and facilitate smoother code modifications, all while ensuring the preservation of the code's intended functionality and output consistency. Subsequently, the exploration extended into the integration of machine learning techniques, addressing the intricate convolution challenges inherent in the neutron generation process.

The successful transformation of the *Transport Code* served to validate the modifications undertaken, resulting in the expected behavior. However, the crux of our investigation lay in the endeavor to incorporate machine learning, leading to promising outcomes. Extensive experimentation and rigorous iterations yielded a model that showcased advancements over previous attempts and exhibited potential in untangling complex resonance scenarios.

The implications stemming from this innovative approach are multifaceted. By leveraging machine learning, researchers could have much cleaner data and thus having a more straight forward analysis. This prospect, although enticing, also prompts us to be mindful of potential limitations and unintended consequences.

Yet, the journey undertaken is far from its terminus. The path forward involves addressing residual limitations and refining untested aspects. The model's performance on experimental data remains a pivotal yardstick in assessing its practicality and reliability. But despite of those tests, the model as been tested on other ToF intervals from 10^2 to $10^8 \mu s$. Each model with the same architecture exhibits high performances across the entire RF, however, as said, further tests are required to conclusively validate this deconvolution approach for recovering the original time-of-flight.

In projecting our gaze toward the future, this report lays the groundwork for an evolution in measurement correction methodologies. Aspirations include the automation of the model's deployment, streamlining the integration into the measurement workflow. The synergy between the realms of neutron physics and machine learning holds the potential to unlock innovative solutions for enduring challenges

at n_ToF.

In conclusion, this report not only bridges machine learning and neutron physics but also introduces a noteworthy chapter in the narrative of data refinement through predictive capabilities. The fusion of data-centric techniques with established simulation frameworks may redefine operational practices at n_ToF, offering renewed avenues for research. Navigating this uncharted trajectory, we embrace a landscape rich with potential, shaping the future of research methodologies and yielding nuanced insights and problem-solving strategies.

Chapter 4

Acknowledgement

I would like to express my sincere gratitude to all those who have been instrumental in the completion of this project and have supported me throughout this journey.

First and foremost, I am deeply thankful to my supervisors, Jose Antonio Pavon Rodriguez and Francisco Garcia Infantes, for their unwavering guidance, invaluable insights, and constant encouragement. Your expertise, patience, and dedication have been instrumental in shaping the direction of this project and my growth as a student.

I extend my heartfelt appreciation to Marta Sabate Gilarte and Javier Balibrea Correa for introducing me to this deconvolution project and consistently pushing me to think outside the conventional boundaries. Your fresh perspectives and thought-provoking discussions have been pivotal in expanding my horizons and approaching challenges with a creative mindset.

I am grateful to the n_ToF group for welcoming me as a Summer Student and providing me with the opportunity to collaborate with a team of brilliant individuals. Your openness to collaboration and willingness to share knowledge have contributed significantly to my learning experience.

To my family and friends, I extend my deepest thanks for standing by me and allowing me the space to discuss and comprehend the intricacies of the work I was entrusted with. Your unwavering support, patience, and understanding have been a huge strength.

Finally, I would like to acknowledge the countless mentors, colleagues, and acquaintances who have shared their expertise, advice, and perspectives along the way. Your contributions, no matter how small, have collectively shaped this project and my growth as a student.

Bibliography

- [1] Federica Mingrone et al. “Development of a Neutron Imaging Station at the n_TOF Facility of CERN and Applications to Beam Intercepting Devices”. In: *Instruments* 3.2 (July 2019), p. 32. ISSN: 2410-390X. DOI: 10.3390/instruments3020032. URL: <http://dx.doi.org/10.3390/instruments3020032>.
- [2] CERN. *n_TOF*. 2023. URL: https://home.cern/science/experiments/n_tof.
- [3] P. Žugec et al. “A direct method for unfolding the resolution function from measurements of neutron induced reactions”. In: *Nucl. Instrum. Methods Phys. Res., A* 875 (2017). 12 pages, 5 figures, pp. 41–50. DOI: 10.1016/j.nima.2017.09.004. arXiv: 1710.07443. URL: <https://cds.cern.ch/record/2290599>.
- [4] Vasilis Vlachoudis et al. “On the resolution function of the n_TOF facility a comprehensive study and user guide”. In: (2021). The document provides an extensive description of the resolution function of n_TOF, methodology of extracting it from the simulated data and how to use it. URL: <https://cds.cern.ch/record/2764434>.
- [5] Python. *tkinter - Python interface to Tcl/Tk*. URL: <https://docs.python.org/3/library/tkinter.html>.
- [6] Rene Brun et al. *root-project/root: v6.18/02*. Version v6-18-02. Aug. 2019. DOI: 10.5281/zenodo.3895860. URL: <https://doi.org/10.5281/zenodo.3895860>.
- [7] Intel. *Intel® VTune™ Profiler*. 2023. URL: <https://www.intel.com/content/www/us/en/developer/tools/oneapi/vtune-profiler.html>.
- [8] Wikipedia contributors. *Huber loss* — *Wikipedia, The Free Encyclopedia*. [Online; accessed 14-August-2023]. 2023. URL: https://en.wikipedia.org/w/index.php?title=Huber_loss&oldid=1167558827.