Investigation of MINERVA experiment data using NuWro Monte Carlo generator with local Fermi gas model.

Author: **Tomasz Bonus**

Supervisor: **Jan Sobczyk**

A thesis presented for the degree of

Bachelor of Physics

Faculty of Physics and Astronomy

Wrocław University

September 4, 2018

Badanie danych z eksperymentu MINERVA przy użyciu generatora Monte Carlo NuWro z modelem lokalnego gazu Fermiego

Autor: **Tomasz Bonus**

Promotor: **Jan Sobczyk**

Praca licencjacka

Streszczenie

Głównym celem tej pracy licencjackiej jest analiza danych uzyskanych z generatora zdarzeń NuWro oraz porównanie ich z danymi doświadczalnymi z eksperymentu MINERVA. Służy to poprawieniu zgodności danych z generatora z danymi doświadczalnymi. Porównanie skupia się na analizie różnicy między tymi danymi i próbach zmniejszenia jej. Odbywa się to poprzez manipulacje wkładem zdarzeń typu MEC. Prowadzi to do obliczenia funkcji przeskalowującej, która pozwala uzyskać poprawę jakości odwzorowania. Funkcja ta została uzyskana za pomocą obliczenia macierzy przeskalowania zależnej od zmiennych kinematycznych skalowanego zdarzenia.

Wprowadzenie do pracy zawiera krótki opis własności neutrin i historię kolejnych odkryć. Przedstawia też argumenty za tworzeniem i usprawnianiem generatorów zdarzeń. Drugi rozdział poświęcony jest generatorowi zdarzeń NuWro oraz przybliżeniu znajomości metody Monte Carlo. Opisuje także te rodzaje rodzaje oddziaływań neutrin z nukleonami zawarte w NuWro, które zostały użyte do wygenerowania danych na

potrzeby obliczeń. Trzeci rozdział opisuje obliczenia prowadzące do uzyskania macierzy przeskalowania. Przedstawia także algorytm genetyczny użyty w celu poprawy rezultatu. W podsumowaniu wynik zostaje porównany do przeskalowania uzyskanego przez inną grupę badawczą. Znajdują się tam także rozważania dotyczące możliwości wykorzystania innych metod które mogłyby posłużyć do znalezienia jeszcze lepszej macierzy przeskalowania.

Abstract

The main goal of this thesis is analysis of NuWro Monte Carlo generator results in comparison to experimental data from MINERvA experiment. This is done to further improve generator performance by making a better fit. The comparison is focused on the contribution of two-body current events. This leads to searching for reweighting function which can provide reduction in mismatch between the generator and experiment data. It is achieved by calculating the scaling matrix and then fine-tuning it through genetic algorithm.

The introduction is a brief description of neutrino and its history. The second chapter is devoted for NuWro event generator and Monte Carlo method. Main neutrino interaction processes simulated are pictured in short way giving only basic and necessary information needed to distinguish them. The third chapter is a description of calculations done to find reweighting function. Method of finding first matrix as well as fine-tuning by genetic algorithm are main parts of thesis so they are in-depth analyzed. In summary, there is a comparison to other reweighting and speculation about other methods that may be used to further improve the fit of the generator, thus reducing produced error.

Contents

List of Figures

Chapter 1

Introduction

Neutrinos are one of the most interesting particles. They nearly don't interact with matter which allows them to travel enormous distances without being disturbed in any way. This is the reason they can give massive amounts of information about the origin of they journey. However, this property is also an obstacle for getting this information. Neutrino measurements can be only done by investigation of their interactions with other particles and this is difficult.

Another fascinating neutrino property is that they oscillate changing their flavour. For each neutrino there is corresponding charged lepton, thus there are electron, muon and tauon neutrinos. This oscillation causes to change that flavour to correspond to another lepton. It was discovered as a solution to **solar neutrino problem**, a discrepancy between theoretically expected and observed flux of solar neutrinos. According to theory, Sun only produces electron neutrinos, so experiments were made in such a way that detectors were sensitive mainly to flow of this one flavour. This gave a much lower detection rate than predicted. It varied between half and third of the amount calculated from theory (depending on experiment it was measured). This proportion to all produced neutrinos was confirmed by Ray Davis and Masatoshi Koshiba and gave them the Nobel Prize in Physics in 2002. Early solution attempts proposed that Sun's models were incorrect. Specifically that the temperature and pressure inside the star were in past much lower than anticipated. This could be an answer due to the fact that neutrinos take a lot less time than heat to escape to Sun's surface after reaction in which they are created. However, this hypothesis was overthrown by helioseismologists which had measured interior temperatures of our star by observations and confirming them with values given by the standard solar model. Other measurements of neutrino flux gave another confirmation that this hypothesis is not correct by describing that

despite the flux would require lower temperature, the energies of detected neutrinos clearly shows that they were created in higher core temperatures.

The real solution being **neutrino oscillation** was first theoretically predicted in 1957 by Bruno Pontecorvo. Effect was observed the first time by Ray Davis and finally experimentally confirmed by the Super-Kamiokande Observatory in 1998 and the Sudbury Neutrino Observatory in 2001 and rewarded with the Nobel Prize in 2015. This was also proof that neutrinos have non-zero mass which forced modifications to the Standard Model.

To get more insight into neutrino nature experiments have to have a way to compare measurements data to theory. This is done by using event generators. They provide information got from simulation about the kinematics of neutrino beam, interaction with the target and final result: a set of particles and their features. According to [2] there are three stages of simulation, each separate from others: beam simulator, event generator and detector simulator. First one provides information about neutrinos incoming to target. The whole specification of the beam is needed to be precisely defined as it is base for event generator. The job of which is to analyze beam and use it to simulate interactions between neutrino and target nucleus and then between produced particles. This is the most crucial part of because some of the processes are currently not fully understood. It leads to producing mistakes which may be not well interpreted later. Way to deal with it is to generate distributions calculated from many events. This leads to reducing the error of single events by associating each value with weight. The result of this approach is creating a possibility to manipulate result by changing only weight functions leading to diminish produced error in comparison to experimental data. Last part of the chain of simulations is most complicated and time-consuming. Data provided by event generator is passed to detector simulator. It computes trajectory of each particle going from initial interaction point and calculates its interactions with each detector element it passes through.

All this heavy computation is done to provide a better frame of reference to the experimental data allowing to detect and analyze earlier modelled events with better understanding.

Chapter 2

NuWro

NuWro is neutrino event generator developed in Wrocław University. It is based on Monte Carlo non-deterministic method of simulation. It was started as PhD project by Jarosław Nowak and later developed under the direction of Jan Sobczyk. The main structure of the code is build using CERN's ROOT framework for numerical computation and event data storage. At the time of writing this thesis, NuWro is being developed by Jan Sobczyk, Cezary Juszczak, Tomasz Golan and Kajetan Niewczas. Generator's documentation is available at `https://nuwro.github.io/user-guide/getting-started/`.

2.1 Monte Carlo method

Monte Carlo is a mathematical modelling method of processes that are too complicated to be calculated analytically. The whole method is based on using random number generators to generate data from a known distribution instead of calculating it. Typical usage of this kind of algorithms are multi-dimensional integration, optimization and simulating various physical phenomena.

In NuWro Monte Carlo is used to determine if the event happened and whether it should be added to the pool of events from which cross sections are calculated. It is done by getting event weight and comparing it to the acceptance probability. Another usage of this kind of non-deterministic algorithm is in the genetic evolution described in rescaling matrix part.

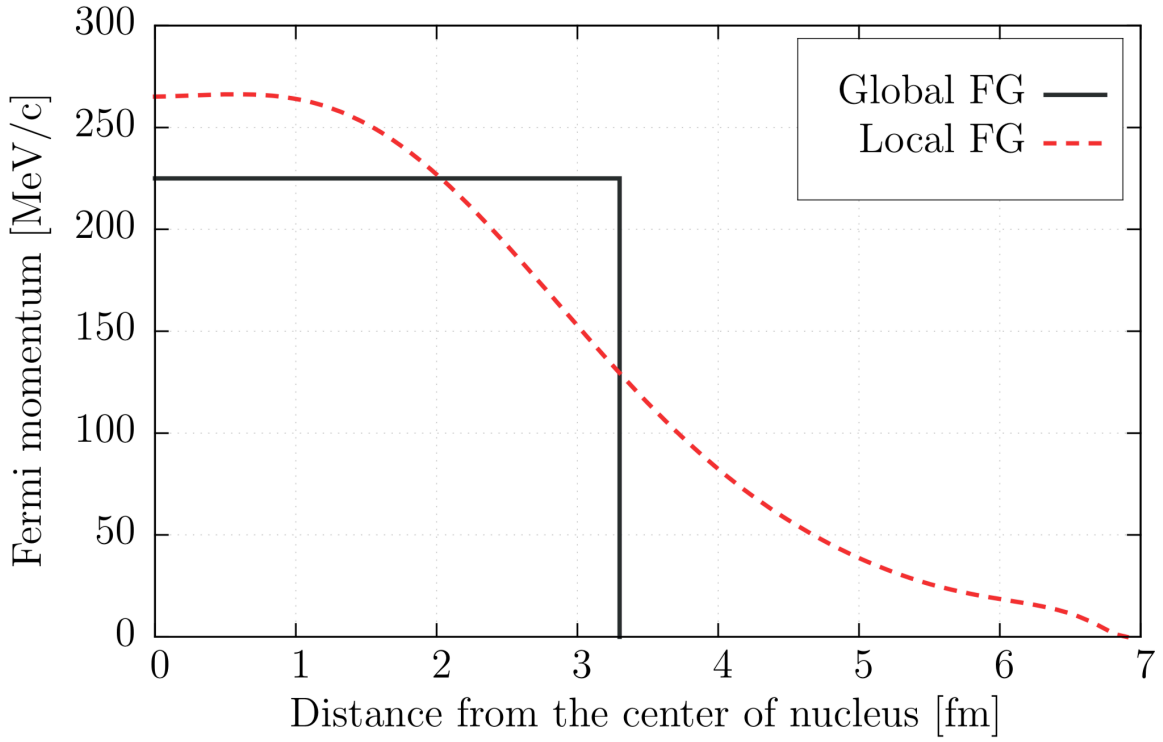


Figure 2.1: The comparison of the Fermi momentum for global and local Fermi gas in the case of carbon [3].

2.2 Local Fermi gas

To describe nucleons inside nuclei NuWro uses Fermi gas model and spectral function. This analysis is however devoted to local Fermi gas. Most common one is the Fermi gas model due to speed and ease of implementation. Model is based on assumption that protons and neutrons are independent of each other and they move freely in a spherical volume of the nucleus with constant nuclear density. Local Fermi gas changes this by introducing local density approximation as a dependency of Fermi momentum. This gives a smoothened version of momentum depending on the distance from the centre of the nucleus. This approximation seems to give theoretically better results in modeling final state interactions after interaction in the primary vertex.

2.3 Processes of neutrino interactions

NuWro implements five types of neutrino interactions: quasi-elastic (QE), meson-exchange current (MEC), resonance (RES), deep-inelastic scattering (DIS) and co-

herent scattering (COH). Each of this processes is modelled in both neutral (where the exchange of the electric charge between two initial particles does not happen) and charged current (where initial particles interact through charged boson). In simulation used to generate data for comparison with MINERVA experimental data only charged current versions of QE, MEC, RES and DIS processes were used, so only these processed would be briefly explained in following sections.

2.3.1 Quasi-elastic

Quasi-elastic scattering is the most frequently occurring process for energies below about 1 GeV [2]. As most of the neutrinos used in experiments belong to this spectrum of energy, this is a process with an overall biggest contribution to the cross-section. According to [2] there are two definitions of 'quasi-elastic', with small difference as one relates to interaction and one to final result. One used in NuWro is being commonly described as $CC0\pi$ and refers to no pions being produced as the result of interaction as well as in the final result. However this is experimental definition as experiment cannot measure directly what interaction happened in primary vertex, only resulting particles. Type of interaction occurring in the simulation is described as:

$$\nu_l n \to l^- p. \tag{2.1}$$

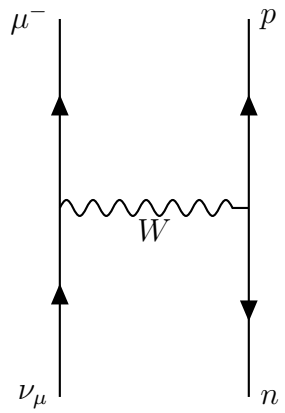


Figure 2.2: Feynman diagram picturing quasielastic interaction described by equation 2.1.

Where l is the flavour of a neutrino and charged lepton. Particles entering interaction are neutrino and neutron. They interact weakly by exchange of positively charged W boson which results in changing the electrical charge of both. Neutrino is converted

to negatively charged lepton of its flavour and neutron is converted to positively charged proton.

2.3.2 Two-body current (MEC)

Experimental data shows that quasi-elastic contribution gives not enough contribution to the cross section as it is not fully described. After further investigation it turned out that there is another important process occurring. Called $np - nh$, in this process n nucleons are emitted from the primary vertex of interaction. Some examples of this interactions are presented in figure 2.3. In NuWro these processes are grouped together and called meson exchange current, but according to [3] some authors refer to MEC as to different parts of this set it is better to call it an onwards two-body contribution to avoid confusion.
NuWro implements four models of $np - nh$:

1. Nieves model with a cut in momentum transfer,
2. Marteau model,
3. Transverse Enhancement (TE) model.

All of which are implemented for charged current but only last for neutral current. It is important to include all of these models as every one gives a different contribution to the cross-section at different parts of neutrino energy spectrum.

Equations describing this charged current process for neutrinos are:

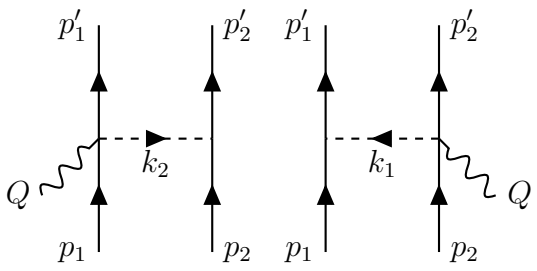
$$\nu_l + p + n \to l^- + p + p \tag{2.2}$$

$$\nu_l + n + n \to l^- + p + n \tag{2.3}$$

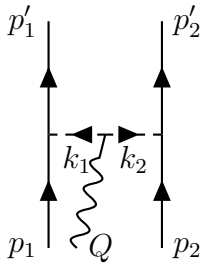
2.3.3 Resonance

Pion production in NuWro is defined by invariant mass W of the hadronic system, where $W^2 = (\sum p_i)^2$, if this value lies in the region between $M + m_\pi$, which is the sum of all nucleus mass and pion mass, and predefined value being set by default to $1.6GeV$. This interaction is similar to quasi-elastic but instead, proton/neutron changing to neutron/proton it changes to Δ particle due to higher energy exchanged in interaction. This particle decays very quickly. Example interactions are:

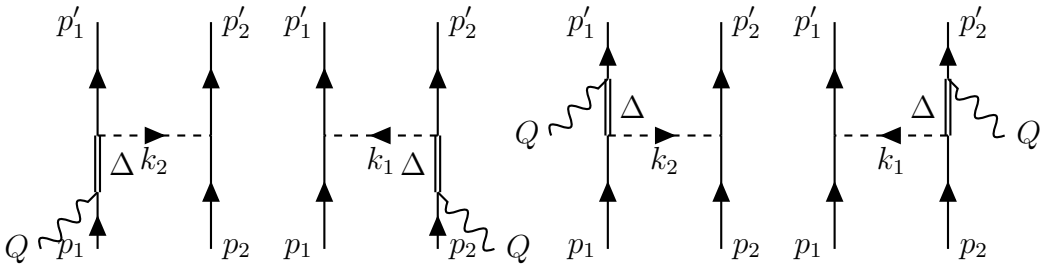
$$\nu_l + p \to l^- + \Delta^{++} \quad \nu_l + n \to l^- + \Delta^+ \tag{2.4}$$



(a) Contact terms.



(b) The pion - in - flight term.



(c) Δ pole and Δ pole crossed terms.

Figure 2.3: Examples of Feynman diagrams for the two body current contribution. Provided by Tomasz Golan [3].

where Δ^{++} decays to p and π^+ while Δ^+ to $p + \pi^0$ or $n + \pi^+$. If π^+ is absorbed in nuclei process is seen in detector as $CC0\pi$.

2.3.4 Deep-Inelastic Scattering

DIS processes are distinguished from resonance ones by invariant mass $W > 1.6GeV$. Deep-inelastic scattering dominates at high energy neutrinos. The equation describing this interaction in charged current:

$$\nu_l + N \rightarrow l^- + X \tag{2.5}$$

where N is nucleon and X is hadronic system. If N is bound inside nucleus A like pictured in Figure 2.5 then its evolution is affected by rest of the nucleus [2].

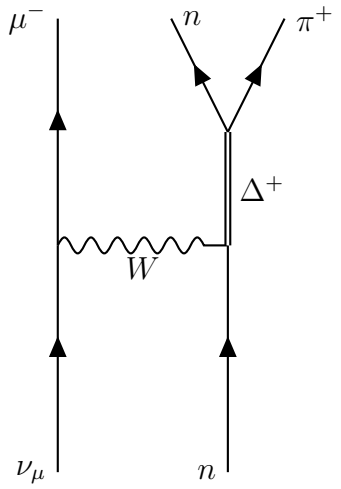


Figure 2.4: Feynman diagram picturing resonance scattering.

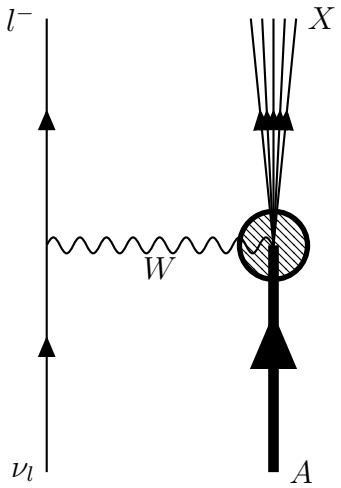


Figure 2.5: Feynman diagram picturing deep-inelastic scattering.

Chapter 3

Calculations

This section will cover the computation part of this thesis, including a description of the algorithms used. Data was generated using NuWro version 18.02.1. The initial computation was done on two sets containing 1 million events each. Final results are obtained on the set generated with 10 million events based on 200 million test events. χ^2 was calculated using the covariance matrix by following formula:

$$\chi^2 = \sum_{j,k} (\sigma_j - \sigma_j^{mc}) M_{jk}^{-1} (\sigma_k - \sigma_k^{mc}) \tag{3.1}$$

Where:

- j, k are indexes of the histogram bins from provided data
- σ_j is cross section from experiment
- σ_j^{mc} is cross section calculated from event generator
- M_{jk} is covariance matrix

3.1 Events analysis

Initial analysis was done using the code provided by Jan Sobczyk. It is written in C++ using the ROOT framework to iterate through events in generated data. The provided code includes a part for calculating cross sections, necessary assumptions according to values of particular variables and histogram generation.

Data to which generator results were compared was obtained from MINERvA experiment at Fermilab. This project is using high-energy neutrino beam to measure and

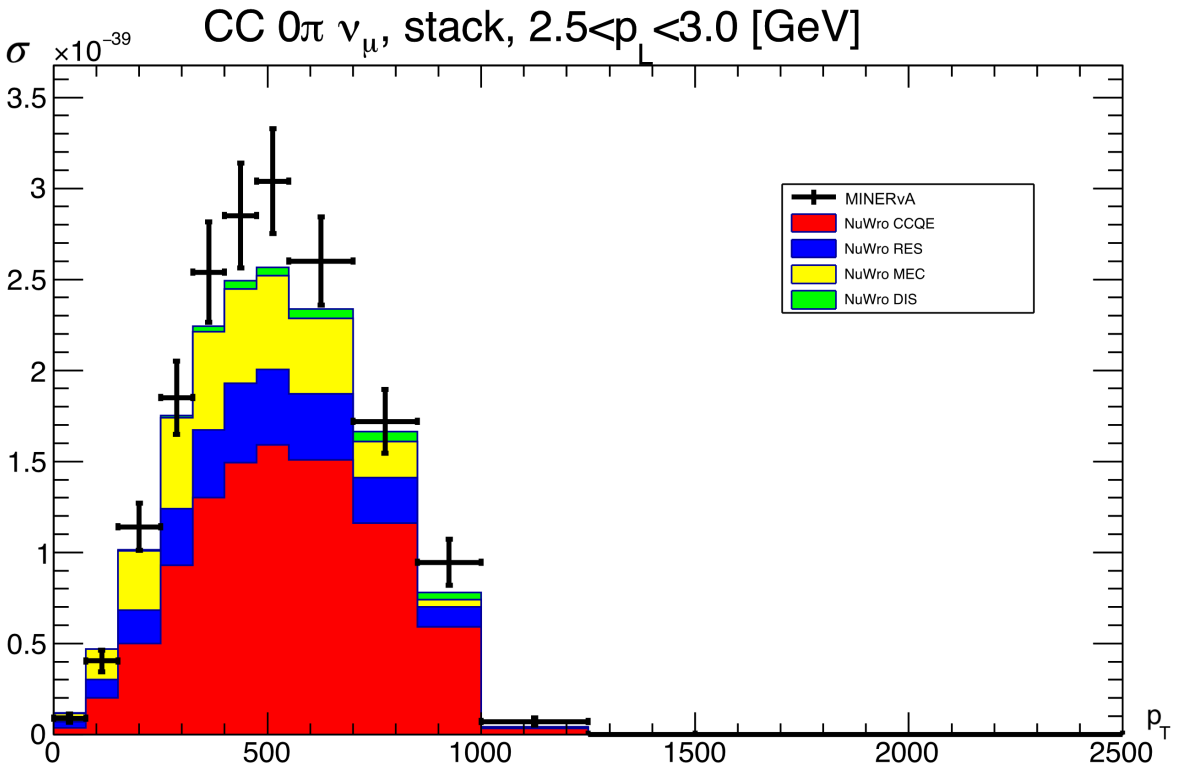


Figure 3.1: One of the 13 histograms picturing comparison between experimental and generated data. There is visible deficiency in calculated cross-section in region with transverse momentum values between 300 and 1000 MeV.

study neutrino interactions. For years it provides experimental data allowing better understanding oscillations and various interactions. It uses five different nuclei targets: carbon, iron, lead, helium and water. MINERvA is located 100 meters below surface level and it is scintillator-based detector dedicated to measurement of signal and background reactions relevant to oscillation experiments. Right next it is located MINOS, near detector which provides information about charge and momentum of outgoing muon [5].

Provided dataset consists of 156 data points with measurement uncertainties from experiment with neutrino flux interacting with carbon nucleus. Each of them is measurement of cross-section corresponding to transversal and longitudinal momentum (respectively p_L and p_T) of muon outgoing from primary vertex of interaction. Analysis is based on assumption that problem with discrepancy exists only due to improper scale of MEC events.

3.1.1 Energy and momentum transfer correlation analysis

The first stage of analysis consisted of getting information about the correlation for MEC events between two kinematic variables and if particular events belong to ones which contribute to excess or deficiency of calculated cross-section in comparison to experimental data. Analyzed kinematic properties of event are:

- ω - transfer of energy. It is defined as the difference between energy of incoming neutrino and particle produced in result of interaction: $\omega \equiv E_{\nu_l} - E_l$.
- q - norm of momentum transfer. Defined as norm of the difference between four-momentum of incoming neutrino and resulting particle: $\|\vec{q}\| \equiv \|\vec{k} - \vec{k'}\|$.

Information that $\|q\| > \omega$ gives computational advantage in analysis due to ability to skip unnecessary calculations of zero elements of matrices.

This analysis gives information about the regular pattern of regions with characteristics of events which have a contribution to an excess of the calculated cross-section. Result is shown on Figure 3.2.

3.1.2 Distribution of MEC events

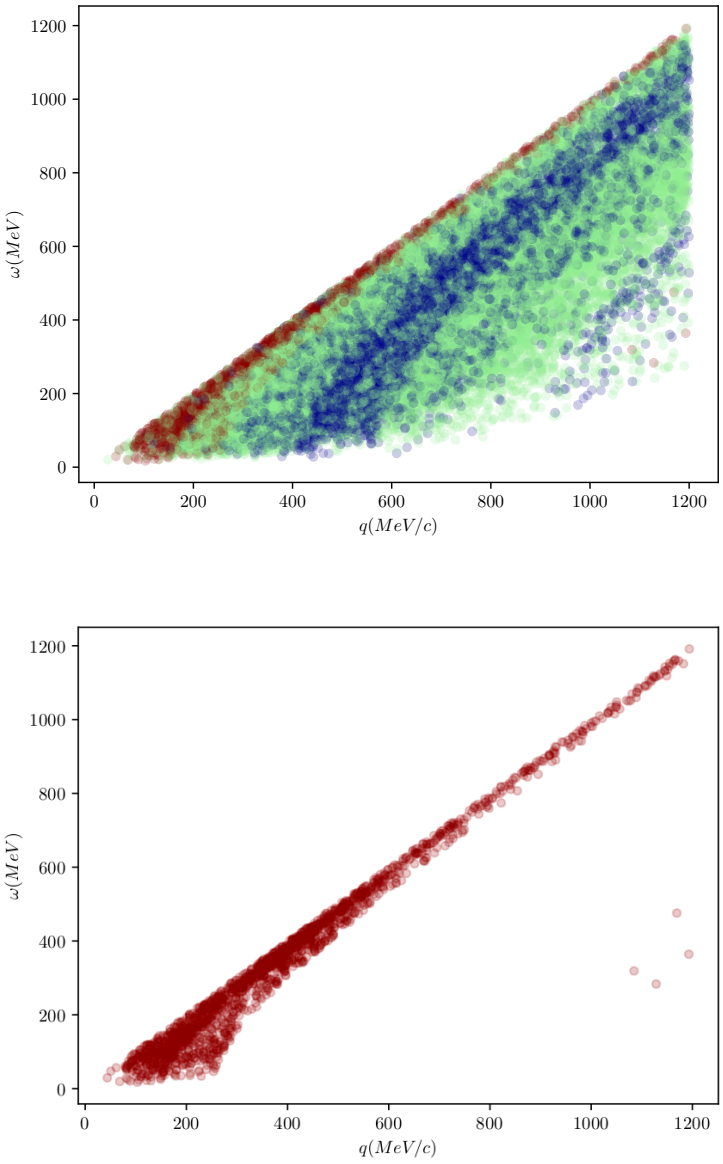


Figure 3.2: Distribution of events in $\omega(q)$ diagram. The top image shows all of the MEC events. The colour of each event is information if the value of cross-section is below, in range or above experimental data. Red dots are representing events which contributed to the excess of the computed cross-section and needs to be scaled down. Green colour represents events which contributed to σ with values inside error bars of experimental data. Blue represents events with insufficient cross sections in comparison to experiment. The lower picture shows only excess of events.

3.2 Searching for a scale function

The main point of this thesis work was to find a function which returns scale for a given energy and momentum transfer.

Scaling is defined as changing event contribution to cross section by multiplying its importance, as if events of this type happened more or less frequently depending on calculated scale. Result of this operation is multiplication of MEC contribution to calculated cross-section by such function that reduces difference between generated and experimental data. This function takes the form of matrix called from now S. Resulting matrix with the best score from preliminary tests is a square matrix with dimensions 24×24. All values in the matrix have to be positive since the event cannot contribute with the negative cross-section.

$$scale = f(q, \omega) = S_{\lceil q/50 \rceil, \lceil \omega/50 \rceil} \tag{3.2}$$

For clearance: ceil operator ($\lceil x \rceil$) returns the smallest natural number bigger or even to value inside. Division by 50 is to give matrix proper shape, knowing that acceptable values of ω and q belong to the range $(0, 1200)$.

3.2.1 Visible error reduction approach

The first method of searching for a scaling matrix was to calculate how much each bin of MEC contribution needs to be rescaled. Values for rescaling each bin in p_L, p_T histogram were calculated from the following formula:

$$S_{bin}^{p_L, p_T} = \begin{cases} max(0, \frac{\sigma - \sigma^{mc}}{\sigma_{MEC}^{mc}} + 1) & \text{if} \quad \sigma_{MEC}^{mc} \neq 0 \\ 1 & \text{if} \quad \sigma_{MEC}^{mc} = 0 \end{cases} \tag{3.3}$$

After obtaining scale for each bin two additional histograms are calculated. One with count of how much events have particular values of q and ω (pictured on Figure 3.3) and second one created in the same way but each event is added with proper weight equal to scale calculated from above formula.

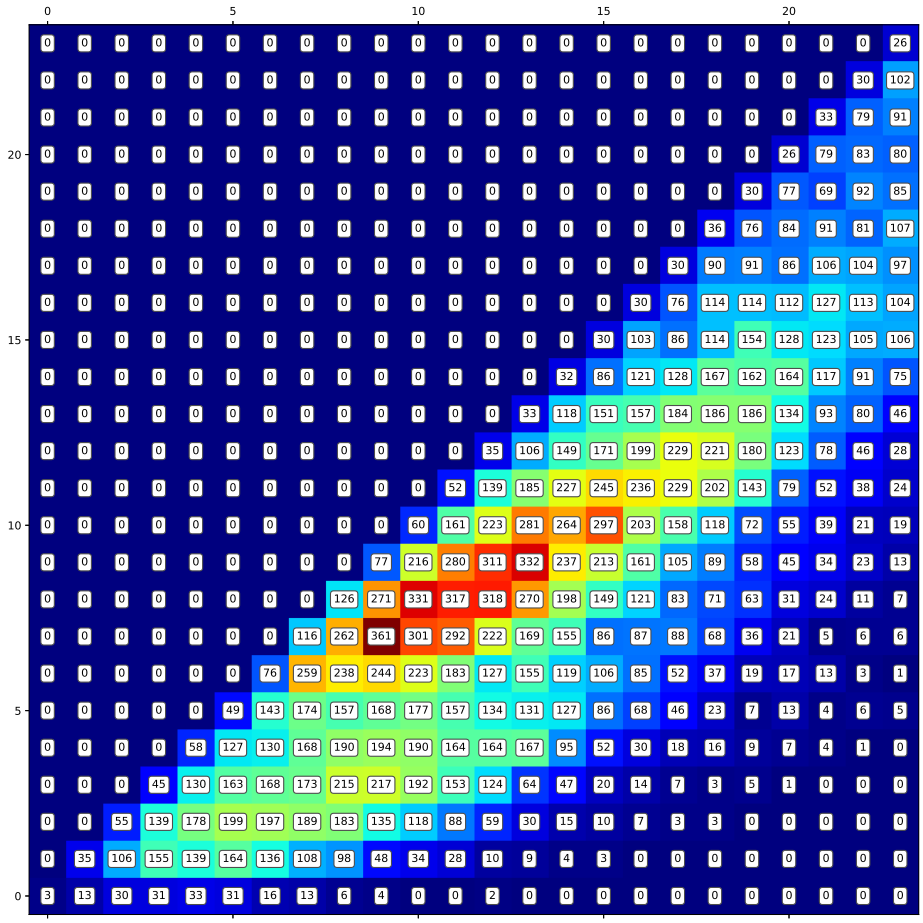


Figure 3.3: Histogram with count of events in $\omega(q)$ correlation generated from about 37000 events. Bins with event number below 10 (chosen arbitrarily) does not contribute to calculating rescaling and scale in corresponding elements is set to 1.0. It is binned representation of Figure 3.2.

Resulting scaling matrix is then calculated by element-wise division of weighted by unweighted matrix with some additional restrictions to avoid overfitting to data on which calculation was run.

$$S_{q,\omega} = \begin{cases} W_{q,\omega}/C_{q,\omega} & \text{for} \quad C_{q,\omega} >= 10 \\ 1.0 & \text{for} \quad C_{q,\omega} < 10 \end{cases} \tag{3.4}$$

where C is unweighted and W is weighted histogram. Result of this operation

is pictured on Figure 3.4. Such calculated matrix allows for over a dozen percent of improvement of χ^2 error (reduction is show along with reduction from next method in Table 3.1).

Whole algorithm runs in such way:

1. Iterate through events and calculate cross-section $\sigma(p_L, p_T)$ for each data point
2. Calculate MEC scale factor for each of these points as described in Formula 3.3
3. Iterate through events second time creating $Count(\omega, q)$ and $Weight(\omega, q)$ histograms
4. Calculate $S(\omega, q)$ matrix by following Formula 3.4
5. Iterate through events third time, applying scale from $S(\omega, q)$ matrix and calculate cross-section $\sigma(p_L, p_T)$ for each data point

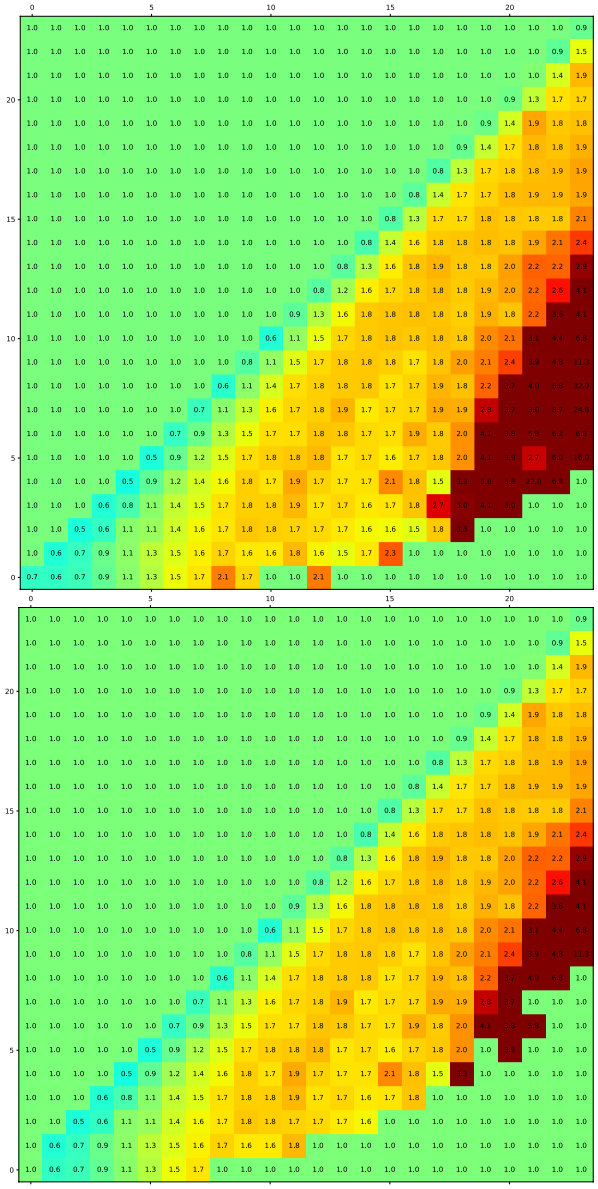


Figure 3.4: Result of calculating rescaling matrices. Top one is without any restrictions on number of events in each bin, while in lower one scales in bins with event number lower than 10 is set to 1.0. This operation is performed to reduce influence of events that are much less likely to happen as they carry very large changes to single values in scale matrix. This results in quite visible difference between them and is the reason of slightly lower performance of latter due to not being able to reduce error of events happening with very small probabilities.

3.2.2 Genetic algorithm approach

The idea to further improve result was to use a genetic algorithm [4]. This kind of numerical optimisation algorithm is based on random changes directed to reach a goal or a certain level of performance. In this case, its usage is described by following algorithm:

```
# create list that will holds matrices
population = []
# put original matrix as first element of the list
population.append(original_matrix)
# fill list with mutated matrices
for _ in range(1, population_size):
    population.append(mutate(original_matrix))
# calculation loop
while True:
    for matrix in population:
        matrix.calculate_score()
    # sort elements by result in descending order
    population.sort(key=lambda m: m.score, reverse=True)
    # save best performing one to file
    # allowing to end computation at any time
    save(population[0])
    # create new generation of matrices
    new_generation = create_generation(population)
    # overwrite population with new matrices
    population = new_generation
```

Listing 3.1: Algorithm for genetic evolution of scaling matrices

Probability of giving offspring is given by following formula:

$$p(M) = \frac{score_M - score_{min}}{\sum_i^N (score_i - score_{min})} \tag{3.5}$$

where N is size of the population, $score_i$ is χ^2 error of i^th matrix and $score_{min}$ is lowest score in this particular population. Subtracting worst result value ensures that worst performing matrices will have significantly lower chance to produce offspring. This also provides better differentiation across better performing ones because difference between

them stays the same, its magnitude compared to score is now much bigger. For this specific case it also removes need for changing performance score from χ^2 to something else as it gives best performing matrices biggest absolute value of score. Algorithm that generates new generation have following form:

```
def create_generation(population):
    new_generation = []
    # get best performing matrix from last generation
    new_generation.append(population[0])
    # calculate probability of giving offspring for each
       matrix
    scores = [matrix.score for matrix in population]
    score_min = min(scores)
    reduced_scores = [score - score_min for score in scores]
    scores_sum = sum(reduced_scores)
    probabilities = [score / scores_sum for score in
       reduced_scores]
    # generate new matrices to fill population
    for _ in range(1, population_size):
        # choose parents
        parent_a = choose_element(population, probabilities)
        parent_b = choose_element(population, probabilities)
        # create child
        child = parent_a.copy()
        for i in range(0, 24):
            for j in range(0, 24):
                # fill elements randomly from parents
                if random(0, 1) > 0.5:
                    child[i][j] = parent_b[i][j]
        # mutate newly created matrix
        mutate(child)
        # add child to population
        new_generation.append(child)
    return new_generation
```

Listing 3.2: Creating new generation

Modifying matrices is done by following algorithm:

```
def mutate(matrix):
    # iterate over allowed elements
    for i in range(0, 24):
        for j in range(i, 24):
            # decide if element will be changed
            mutation_chance = random(0, 1)
            if mutation_chance < mutation_probability:
                # choose mutation type
                mutation_type = random(0, 1)
                if mutation_type < 0.5:
                    matrix[i][j] *= random(-2, 2)
                else:
                    matrix[i][j] += random(-0.5, 0.5)
```

Listing 3.3: Algorithm for modifying matrices

Generating new matrix from previous generation is done by randomly choosing two different matrices and then filling its values with corresponding values from parent matrices with probability for inheriting each subsequent one from each of parents equal to:

$$p(A) = \frac{p_A}{p_A + p_B} \quad p(B) = \frac{p_B}{p_A + p_B} \tag{3.6}$$

where A and B are parent matrices, and p_x are their calculated probabilities of giving offspring.

Simplifying, algorithm works like this:

1. Take one or multiple matrices
2. Create similar matrices from previous ones by making small changes
3. Evaluate performance of each one
4. Select best performing ones
5. Repeat steps 2-4 until satisfying result

Preliminary tests allowed to obtain better results by reducing χ^2 error by additional few percents compared to previous method. Outcome is pictured in Table 3.1.

	χ^2 on small dataset	χ^2 on big dataset
Unscaled result	448	401
Scaling with 1^{st} matrix	392 (-12.5%)	341 (-15%)
Scaling with matrix from GA	369 (-17.6%)	332 (-17.2%)

Table 3.1: Result of scaling with matrix obtained through genetic algorithm.

Chapter 4

Summary

Neutrino experiments gather smaller amounts of data than others high-energy physics experiments like CERN. This is the reason for the need of data generators which are helping to develop a better understanding of the phenomenon and provide a spectrum of information that is not possible to obtain using detectors.

Calculating the scale matrix allowed to achieve some improvements. Usage of genetic algorithms minimizing the value of χ^2 leads to further reduction of produced error. Additional improvement could be achieved by more sophisticated methods like usage of gradient descent algorithm to calculate vector in hyperspace which points to regions with a smaller error. However, this is more time-consuming and requires a higher amount of fine-tuning to avoid falling into local minima and finally achieve much better results. Comparison between unscaled data and after usage of scaling functions described in this thesis and one provided by other researchers are pictured in Figure 4.1. Last rescaling function was calculated by fitting two-dimensional Gauss function to computed scales across values of energy and momentum transfer.

	χ^2 on small dataset	χ^2 on big dataset
Unscaled result	448	401
Scaling with 1^{st} matrix	392 (-12.5%)	341 (-14.9%)
Scaling with matrix from GA	369 (-17.6%)	332 (-17.2%)
Genie rescaling	618 (+37.9%)	474 (+18.2%)

Table 4.1: Result of scaling with various methods (Calculated χ^2 takes correlations between data points into account).

As seen on Table 4.1 result of scaling with function provided by other group per-

forms much worse than rescaling described in this thesis, giving result even worse than unscaled generator data. However if calculated error does not take correlations between data into account, but only relies on differences between data point and generator result this scale gives positive result.

	χ^2 on small dataset	χ^2 on big dataset
Unscaled result	402	382
Scaling with 1^{st} matrix	268 (-33.3%)	269 (-29.5%)
Scaling with matrix from GA	246 (-38.8%)	275 (-28.0%)
Genie rescaling	375 (-6.7%)	342 (-10.4%)

Table 4.2: Result of scaling with various methods without taking correlations into account.

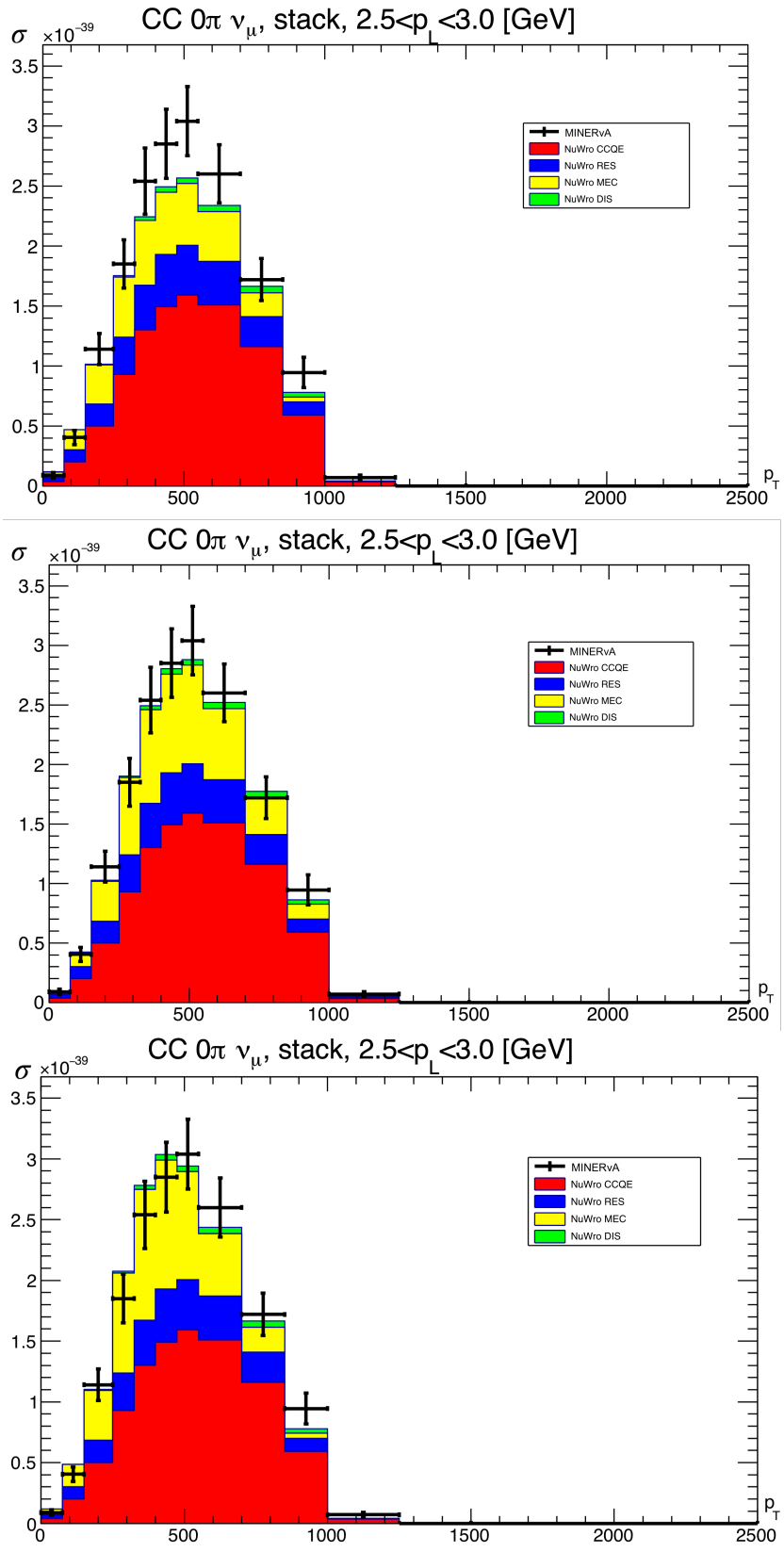


Figure 4.1: Comparison of different rescalings:

Top: data without applied scaling.

Middle: data with calculated scaling matrix.

Bottom: rescaling function calculated by researchers developing Genie generator.

Bibliography

[1] *Nuwro user guide webpage*, nuwro.github.io/user-guide/.

[2] L. Alvarez-Ruso, M. Sajjad Athar, M. B. Barbaro, D. Cherdack, M. E. Christy, P. Coloma, T. W. Donnelly, S. Dytman, A. de Gouvêa, R. J. Hill, P. Huber, N. Jachowicz, T. Katori, A. S. Kronfeld, K. Mahn, M. Martini, J. G. Morfín, J. Nieves, G. N. Perdue, R. Petti, D. G. Richards, F. Sánchez, T. Sato, J. T. Sobczyk, and G. P. Zeller, *Nustec*[1] *white paper: Status and challenges of neutrino-nucleus scattering*, Progress in Particle and Nuclear Physics, (2018).

[3] T. Golan, *Modeling nuclear effects in nuwro monte carlo neutrino event generator*, (2014).

[4] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*, MIT Press, Cambridge, MA, USA, 1992.

[5] The MINERvA Collaboration, *Minerva in a nutshell*, MINERvA-doc-6444-v25, (2017).