TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DIPLOMARBEIT

Monte Carlo simulations for the calculation of differential and double differential cross sections using FLUKA

ausgeführt am Atominstitut
der Technischen Universität Wien

unter der Betreuung von

Univ.Prof. civ.ing. tekn.lic. tekn.dr. Lembit Sihver

und

Univ.-Ass. Dipl.-Ing. Dr.techn. Albert Hirtl

durch

David Ellmeyer, Matr.-Nr. 1126053, 066 461

February 8, 2017

Kurzfassung

Am Forschungs- und Therapiezentrum MedAustron in Wiener Neustadt wird die Krebstherapie mittels Protonen und Kohlenstoffionen durchgeführt. Im Unterschied zur Photonenbestrahlung zeigen Ionenstrahlen andere Eigenschaften. Es gibt ein scharfes Maximum der abgegebenen Energie bei einer steuerbaren Tiefe – der sogenannte Bragg-Peak. Dabei ist das einfallende Ionenprojektil vielen nuklearen Wechselwirkungen mit den Target-Kernen ausgesetzt. In diesen Prozessen entstehen leichtere Sekundärteilchen, die sich ebenfalls durch das Gewebe bewegen und ebenso einen Beitrag zum Tiefen-Dosis-Profil liefern, der sich direkt nach dem Bragg-Peak befindet. Dieses Auftreten muss als Teil der Behandlungsplanung miteinbezogen werden. Für das Verständnis der komplexen Fragmentierungsprozesse und damit der Beiträge der einzelnen Fragmente zur Dosis in der Ionentherapie müssen differentielle und doppelt-differentielle Wirkungsquerschnitte für alle auftretenden Prozesse bestimmt werden. Diese spielen eine Schlüsselrolle in jedem Therapieplan für die Bestrahlung von Krebspatienten. Im Zusammenhang mit der vorliegenden Diplomarbeit wurde die Simulation auf dünne Targets gelenkter Ionenstrahlen mit dem Code `FLUKA` durchgeführt. Die Wechselwirkungen von ^{12}C-Projektilen mit Energien von 10 bis 500 MeV pro Nukleon mit verschiedenen in menschlichen Geweben auftretenden Targets werden untersucht. Der Großteil der medizinisch verwendeten Teilchenarten ist bereits in `FLUKA` implementiert, andere werden mittels `FORTRAN`-Zusatzcodes ebenfalls einbezogen.

FLUKA ist ein weitverbreitetes Simulationspaket und lässt passende Abschätzungen für differentielle und doppelt differentielle Wirkungsquerschnitte zu, speziell dort, wo es keine zugänglichen experimentellen Daten gibt. Das Ergebnis dieser Arbeit ist ein Datensatz aus Wirkungsquerschnitten, der einen Beitrag zur Erstellung von Bestrahlungsplänen in der Inonenstrahltherapie sowie zum Vergleich mit ähnlichen Codes leisten kann.

Abstract

At the MedAustron facility in Wiener Neustadt ion-beam therapy tumours will be treated using protons and Carbon ions. In contrast to photon irradiation, ion-beams show different characteristics: there is a sharp peak of deposited energy, the Bragg peak, at an accurately controllable depth. The incoming projectile ions undergo a series of nuclear interactions with the target nuclei. Lighter secondary particles are produced and propagate through the target tissue and generate a fragmentation tail directly after the Bragg peak, which is an important contribution to be considered for the treatment planning. In order to improve the understanding of the complex fragmentation processes and, thus, the contribution of the various fragments to the delivered dose during ion-beam therapy, differential and double differential cross sections need to be determined for all relevant occurring processes. These cross sections also play a key-role in any treatment planning system used to plan the optimal irradiation of cancer patients when using ion beams. In the context of this master thesis `FLUKA` simulations of a particle beam directed onto a thin target were performed. Reactions of ^{12}C projectiles with energies ranging from 10 to 500 MeV/n impinging on various targets found in human tissue was studied. Most of the required particle types are already implemented in `FLUKA`, others are included by dedicated `FORTRAN` user routines. `FLUKA` is a well established simulation framework and allows for an estimation of differential and double differential cross sections where no experimental data are available. The result of this work

is a cross section dataset, which could be used for an improvement of treatment planning systems used in ion-beam therapy, as well as for comparisons with other particle and ion transport codes.

Danksagung

Zu Beginn möchte ich allen Menschen danken, die mich beim Durchführen der vorliegenden Arbeit unterstützten.

Einige Zeit ein winziger Teil des Fortschritts in einem so wichtigen Zukunftsbereich zu sein, war für mich eine ganz besondere Ehre. Ich bin sehr erfreut, nach einer sehr arbeitsintensiven Zeit im Wissen zu sein, viel dazugelernt zu haben und dies für den weiteren Verlauf meines Lebens nutzen zu können.

Ganz besonders möchte ich daher bei Dr. Sihver und Dr. Hirtl bedanken, die es mir ermöglichten, im Rahmen meines Masterstudiums in einem Feld tätig zu werden, das mein persönliches Interesse an medizinischer Physik und dessen praktischen Nutzen einerseits und theoretischer Arbeit und Simulationsmethoden andererseits vereint.

Eine weitere große Unterstützung waren meine Kollegen im Studium und innerhalb der Forschungsgruppe, die mit ihren Denkansätzen durch die Jahre hindurch immer wieder neue Impulse lieferten.

Ein ganz großer Dank gilt meiner Familie, speziell meinen Eltern und Geschwistern, die – wann immer es auch nötig war – zur Seite standen und mir auf meinem langen Lernweg eine große Hilfe darstellten.

David Ellmeyer

Contents

1. Introduction

For many decades cancer research has been one of the most regarded areas in medical resarch.

Cancer is well-known as a cruel, lenghty and much too often as a deadly disease. In biological terms, cancer is described as a malignant formation of new tissue, also including an unrestrained growth of cells.

In the last few decades, there has been enormous progress in cancer research. The methods of treating cancer patients have become more flexible, variable and not least more precise. From resection over chemotherapy to irradiation there are lots of ways to remove malignant tumors from human tissue.

Nevertheless, given the fact, that millions of people die of cancer every year, research in this field is still far from having found an ideal solution for the treatment of this disease.

As mentioned above, one promising way to treat cancer is radiation therapy. So far, the most common types of radiation used in this context are photons and electrons. Furthermore there is a relatively new approach for the treatment of cancerous tumors, which is called ion beam therapy. At the MedAustron therapy and research center in Wiener Neustadt the main focus is set on this technique. In the last few months, the clinical area opened its doors for the first patients.

In order to evaluate the ion beam therapy method, it is necessary to analyze it repeatedly in a medical and physical way. In this connection, knowledge about

the nuclear interactions included in ion beam therapy is essential.

As ion beam therapy is a relatively *young* approach of treating cancer diseases, only few experimental data about the nuclear interactions and scattering processes are accessible for that energy range. Therefore, numerical simulations can help to get a better understanding and contribute to the preparation of irradiation planning systems.

The present work gives a short overview of the physical background of radiation therapy in general and explains the advantages of therapy using heavy ion beams. The core of the following chapters will be the simulation of differential and double differential cross sections in the context of ion beam therapy. As there have been scientific studies about the simulation with various numerical codes, the package `FLUKA` [1, 2] has a special role to play in the present work. Therefore also the underlying models of the tool kit will be introduced in a compact way. The last few sections will deal with the results and analysis of the underlying calculations.

2. Ion beam therapy

In the broad field of medical physics there have been several approaches for highly efficient methods to treat cancerous tumors.

More than 50 % of the cancer patients with malignant tumors are treated with radiation. The hitherto conventional therapy methods using photons are common and accepted as relatively effective ways, especially for cancers in deeper tissue regions. In this type of therapy, high energetic x-rays and gamma rays are directed onto human tissue. Besides, electrons are utilized for superficial cancers.

Although high energetic photon therapy has often lead to demonstrable results, human tissue is usually also damaged. This is an effect of the dose profile of photons, that deposit most of their intensity at a relatively low depth. So if cancer in body regions, that are hard to reach, needs to be treated, the initial intensity of the photon beam has to be accordingly high or even does not represent an effective way for the treatment. This can lead to bad side effects and often causes a hard convalescence processes or in the worst case other diseases.

In about seven decades the method of ion beam therapy has been developed. One of the first ones to think about this type of method in a scientific way was Robert R. Wilson in 1946 [3]. In the following few years, the first physical and radiobiological experiments were started. In 1954 the patient treatments with protons were started.

The principle of ion beam therapy is based on the fact that ions, especially the

more massive ones than protons, are able to *work* a lot more effective on tumors than photons or electrons. Their intensity profile shows the advantage that ions deposit almost all of their energies at a certain position much deeper than other particle types. This is explained in detail in Section 2.1.

For the understanding of scattering processes, the concept of differential, double differential and total cross sections is explained in 2.2. Also the background of the electron stopping in heavy ion collision processes, the Rutherford (or *Coulomb*) scattering is described.

As mentioned above, heavy ions show in some points different characteristics compared to protons. Although ions in general cause fragmentation processes, in case of proton radiation, only the target gets *fragmented*, whereas using heavy ions as primary particles also the projectiles do. They produce a so called *Bragg tail* in their depth-dose profile, because of the production of lighter secondaries. These processes of nuclear fragmentation will be discussed in detail in Section 2.3.

2.1. Bethe-Bloch formula

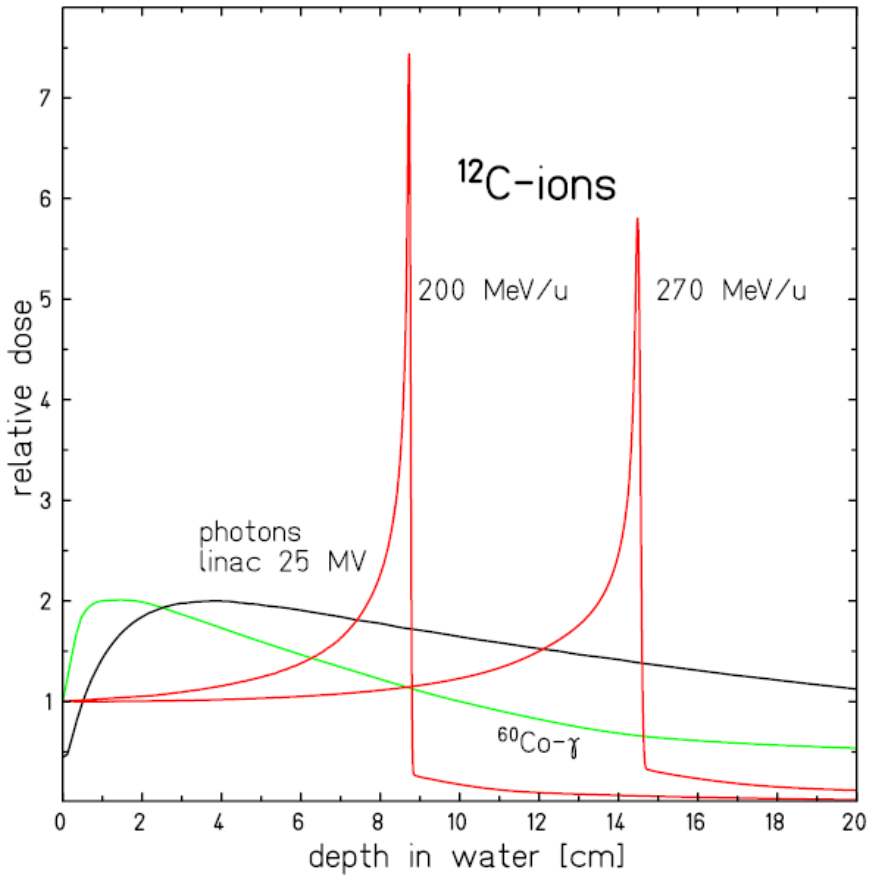


Figure 2.1: Depth-dose profile of ^{12}C ions compared to photons [4]

As mentioned above, heavy ions show completely different dose deposition characteristics than photons or electrons. This is shown on Figure 2.1. Photons deposit most of their energy in a low depth, while ^{12}C ions show a sharp dose peak at a certain depth. This is called the *Bragg peak*. In order to illustrate, why these particle types show a different behavior, the next few paragraphs will introduce the Bethe-Bloch formula, based on [5, 6]. The Bethe-Bloch formula describes the energy loss of charged heavy particles while passing through matter. In order to understand the meaning of the Bethe-Bloch formula, it is useful to derive it in

detail. First of all, there are some hypotheses that have to be considered.

The incident particle with velocity v is charged with Ze, holds a mass m and passes an electron in a certain distance b. The electron is assumed to be in a free state and does not move during the interaction process. Also, it is assumed, that the incident particle is not deflected because its mass is much higher than the electron mass.

The quantification of the energy decrease begins with the calculation of the momentum, which results from the time integral over the electric force F.

$$\Delta p = \int F dt = q \int E_\perp \cdot \frac{dt}{dx} dx = \frac{q}{v} \int E_\perp dx, \tag{2.1}$$

where $E_\perp$ is the perpendicular component, that contributes to the electric field in x-direction.

In order to get a general expression for the electric field in this case, the Gaussian law is used, which allows to transform a volume integral of the divergence of a vector field into an integral of the field in the boundary of the surface. Using Maxwell's equation $\nabla \mathbf{E}(x) = 4\pi\rho(x)$ and the charge density $\rho(x) = Ze\delta(x)$ for a point charge, the following relation can be obtained

$$\int_{\partial A} E_\perp dA = \int E_\perp \cdot 2\pi b dx = 4\pi Ze \rightarrow \int E_\perp dx = \frac{2Ze}{b}. \tag{2.2}$$

Using the classical energy-momentum relation, the kinetic energy of the incident particle can be calculated.

$$\Delta p = \frac{2Ze^2}{bv} \rightarrow \Delta E_{\text{kin}}(b) = \frac{(\Delta p)^2}{2m_e} = \frac{2Z^2e^4}{m_\text{e} b^2 v^2} \tag{2.3}$$

In the next step, the energy has to be rewritten as a differential including the

infinitesimal change of position dx:

$$dE_{\mathrm{kin}}(b) = \Delta E_{\mathrm{kin}}(b) N_e dV = 2\frac{Z^2 e^4}{m_{\mathrm{e}} b^2 v^2} N_{\mathrm{e}} \cdot 2\pi b \cdot db\, dx = \frac{4\pi Z^2 e^4}{m_{\mathrm{e}} v^2} N_{\mathrm{e}} \frac{db}{b} dx, \quad (2.4)$$

with the electron density N_{e}.

The integration limits are defined by the impact parameter b. The minimum of b represents a central collision were the maximum kinetic energy is $E_{\mathrm{kin}} = \frac{\mathrm{m_e(2v)^2}}{2}$ and the maximum of b is reached, when the energy of the electron corresponds to the ionization potential:

$$\begin{aligned} 2m_e v^2 &= \frac{2Z^2 e^4}{m_{\mathrm{e}}^2 b^2 v^2} \rightarrow b_{\mathrm{min}} = \frac{Ze^2}{m_{\mathrm{e}} v^2}, \\ I &= \frac{2Z^2 e^4}{m_e b^2 v^2} \rightarrow b_{\mathrm{max}} = \sqrt{\frac{2}{m_{\mathrm{e}} I}} \frac{Ze^2}{v}. \end{aligned} \quad (2.5)$$

After a simple integration, the formula of *classical* energy loss can be obtained:

$$-\frac{dE}{dx} = \frac{2\pi Z e^2}{m_{\mathrm{e}} v^2} N_e \cdot ln \frac{2m_{\mathrm{e}} v^2}{I} \quad (2.6)$$

Finally, relativistic and quantum mechanical corrections lead to the following formula [4]:

$$-\frac{dE}{dx} = \frac{4\pi e^4 Z_{\mathrm{t}} Z_{\mathrm{p}}^2}{m_{\mathrm{e}} \beta^2 c^2} \left[ln \frac{2m_{\mathrm{e}} v^2}{\langle I \rangle} - ln(1-\beta^2) - \beta^2 - \frac{C}{Z_{\mathrm{t}}} - \frac{\delta}{2} \right] \quad (2.7)$$

As it can be seen, the $\frac{1}{\beta^2}$ behavior of the energy loss is dominant (also see Figure 2.2). The particles are ionized more strongly and therefore, the charge number changes to Z_p. In the case of lower energies, Z_p is replaced by an effecetive charge number [4] $Z_{eff} = Z_p \left[1 - exp(-125\beta Z_p^{-2\beta})\right]$. At a velocity of $v_p \approx Z_p^{2/3} v_0$ the energy loss reaches its maximum. This velocity also corresponds to the *Brag peak*.

The energy loss rate $\frac{dE}{dx}$ can be useful to calculate the particle's path length

$R(E)$:

$$R(E) = \int_0^E \left(\frac{dE'}{dx}\right)^{-1} dE'. \tag{2.8}$$

While a ^{12}C ion shows a linear like $R(E)$ behavior, because it is likelier to stay in the beam direction, the path length of lighter particles increase much faster in with increasing energy. As explained in [4], the range of different particles with the same energy scales with $\frac{A}{Z^2}$.

Moreover, the Bethe-Bloch formula allows the further derivation of the energy dose D [4]:

$$D\,[\mathrm{Gy}] = 1.6 \times 10^{-9} \times \frac{dE}{dx}\left[\frac{\mathrm{keV}}{\mu\mathrm{m}}\right] \times F\left[\mathrm{cm}^{-2}\right] \times \frac{1}{\rho}\left[\frac{\mathrm{cm}^3}{\mathrm{g}}\right] \tag{2.9}$$

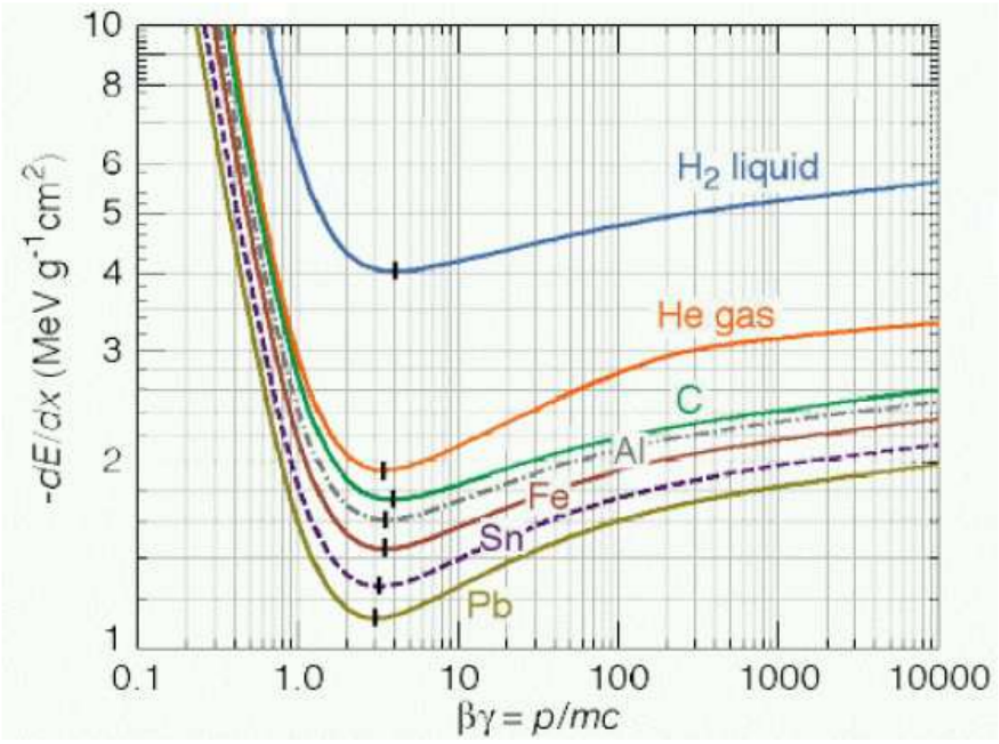


Figure 2.2: Energy loss $-\frac{dE}{dx}$ of heavy ions in different target materials [6]

where ρ in this simple calculation is the density and F is the particle fluence. This physical quantity D is especially known as essential component of irradiation planning systems and gives information about the specific exposure.

2.2. Scattering theory

In ion beam therapy, scattering processes play an essential role. The next few paragraphs introduce the fundamental concept of cross sections and give a mathematical derivation for the contribution of electrons in heavy ion collision and stopping processes.

2.2.1. Cross sections

Cross sections give information about the probability of scattering processes [7]. In order to give a verbal explanation for the physical meaning of this entity, it can be written as [8]

$$\sigma = \frac{\text{Number of scattering processes}}{\text{Number of target atoms} \times \text{time intervall} \times \text{incident current density}}. \quad (2.10)$$

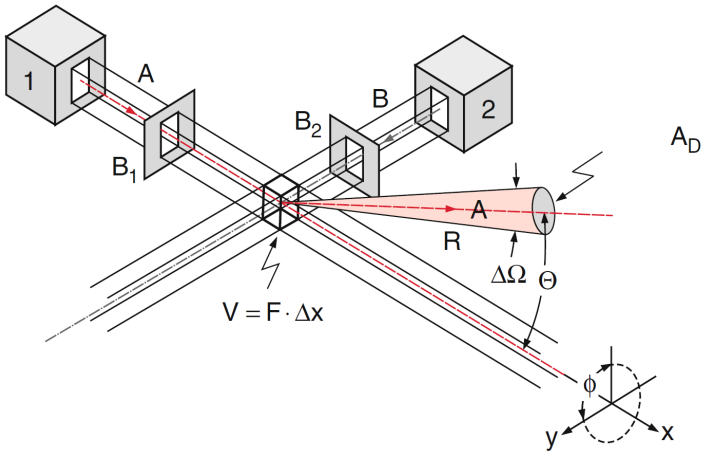
The cross section itself is an *effective surface*, its common unit is *barn*, which is 10^{-24}cm^2. Moreover one has to differentiate between three other types of cross sections.

The **differential cross section** [7] takes into account the cross section's dependency of the exit solid angle. It is given by $\frac{d\sigma}{d\Omega}$, with the solid angle Ω (see Figure 2.3). The so called **total cross section** [7] is defined by the integral of the differential cross section over the 4π solid angle.

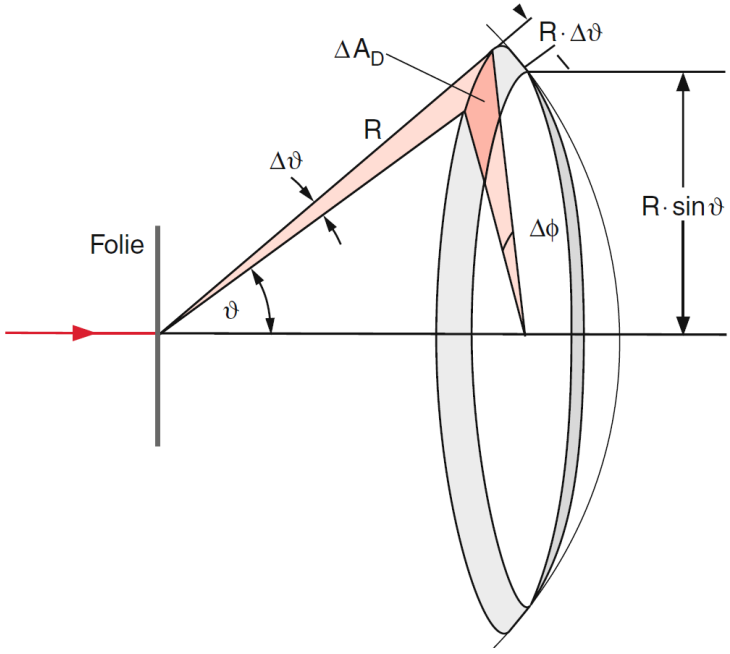
If the initial beam consists of adequately light particles (for examples electrons), differential cross sections also give information about the elementary structure of the target particles, such as nucleus parameters and distribution of charge and spin, that both can be derived from the energy loss of the primary particle.

If one has to consider different energies of the scattered particles due to inelastic nuclear processes it is useful also to calculate the **double differential cross**

section which is the derivation of differential cross section by the kinetic energy of the corresponding particle differential cross section.



(a) Differential and total cross section



(b) Differential cross section

Figure 2.3: Setup of a typical scattering experiment: An incident beam is deflected by a target into a hemisphere with the polar angle θ and azimuthal angle ϕ. The differential cross section is an angle-dependent measure for a certain angular area and the total cross section is the integral of the whole hemisphere.

As this work is written in the frame of research in heavy ion beam therapy, there will be a focus on inelastic scattering processes and especially nuclear interactions in the following sections and chapters. The different types of cross sections will then help to understand these physical mechanisms.

2.2.2. Rutherford scattering

Another contribution to collision processes is represented by Rutherford scattering. In general, energy loss of heavy ions is already described in Section 2.1, therefore only a short introduction is given.

According to the Born approximation, the differential cross section of a particle passing any local charge distribution is given by [9]:

$$\frac{d\sigma}{d\Omega} \propto \int V(\mathbf{r}) e^{i\mathbf{qr}} d\mathbf{r}, \tag{2.11}$$

which is the Fourier transform from position $\mathbf{r}$ to momentum transfer $\hbar\mathbf{q}$ of the electric potential. A Coulomb potential is typically described as $V(\mathbf{r}) \propto \frac{Z_1 Z_2}{r}$, with $r = |\mathbf{r}|$. The Fourier transform results in [9]:

$$\frac{d\sigma}{d\Omega} \propto \left(\frac{Z_1 Z_2}{E_0}\right)^2 \frac{1}{\sin^4 \frac{\theta}{2}}, \tag{2.12}$$

including the initial energy E_0 and the scattering polar angle θ. This is the calculation for the electric potential of a charged point particle. If other charge distributions are requested, the differential cross section can be derived analogously. In this case, the electric potential depends on the integration of the charge density ρ. Having considered that the incident particle interacts with the charge distribution of extent r at position t of its trajectory in a distance s, the differential

cross section is calculated by a mathematical convolution $\mathcal{F} * \mathcal{G}$ [10]:

$$\begin{aligned} \frac{d\sigma}{d\Omega} \propto \int V(t) e^{i\mathbf{qt}} d\mathbf{t} &\propto \int \underbrace{\frac{Z_1 Z_2}{|\mathbf{t}-\mathbf{s}|}}_{\mathcal{F}(\mathbf{t}-\mathbf{s})} \underbrace{\rho(\mathbf{t}-\mathbf{r})}_{\mathcal{G}(\mathbf{t}-\mathbf{r})} e^{i\mathbf{qt}} d\mathbf{t} \\ &\propto \int \mathcal{F}(\mathbf{t}-\mathbf{s}) * \mathcal{G}(\mathbf{t}-\mathbf{r}) e^{i\mathbf{qt}} d\mathbf{t} \end{aligned} \tag{2.13}$$

Using the fact that the Fourier transform of a convolution equals the product of the Fourier transforms of each factor, the differential cross section can be written as [9]:

$$\frac{d\sigma}{d\Omega} = \left(\frac{d\sigma}{d\Omega}\right)_{\text{Rutherford}} \cdot \underbrace{\int \rho(\mathbf{r}) e^{i\mathbf{qr}} d\mathbf{r}}_{\text{Form factor } F(\mathbf{q})} \tag{2.14}$$

This is called the *Mott cross section* and shows that the differential cross section of Rutherford scattering for charge distributions ρ can by corrected by a factor $F(\mathbf{q})$, which is known as *form factor*.

Regarding heavy ions one has to note that the contribution of Rutherford scattering to the energy loss is relatively low. Nevertheless, the cumulative effect of multiple scattering processes can be important for the estimation of doses.

2.3. Nuclear interactions

In Section 2.1 it was discussed, that the collision of high energetic ions with target compounds causes energy loss of the projectile. During this continuous process, inelastic nucleus-nucleus interactions result in inner excitations of both, the primary and the target particles. These new nucleon states lead to a de-excitation process consists of four main components [11]:

- Fission is an effect that becomes important for nuclei with high charge number at about $Z \gtrsim 65$. In this case, the nucleus breaks into two fragments,
- Fermi-breakup mechanism appears in light nuclei with $A \lesssim 16$. The excitation energy of the nucleus can be larger than the binding energy, therefore smaller particles are produced.
- Evaporation causes the emission of lighter ions – such as ^{1}H, ^{2}H, ^{3}H and ^{4}He (α particles) – and of particles of other types [4].
- Gamma emission can appear after other steps of de-excitation.

These effects were observed in many experiments (e.g. by Schardt, Sihver et al. in [12]) and are summarized as fragmentation. In the context of this work, especially the process of evaporation is important, because the components of human tissue, which is treated in ion beam therapy, mostly have lower Z [11].

As it can be seen in Figure 2.4, the residual projectile-like particles are not deflected very strongly and basically keep their velocity, whereas the target fragments can be found in other regions of the 4π solid angle.

It was also shown in Section 2.1, that the mean path of lighter ions increase with energy due to the $\frac{A}{Z^2}$ dependency of this quantity. Therefore, the production of fragments in evaporation cause a characteristic behavior of the depth-dose curve,

which is known as *fragmentation tail*. While in case of ^{1}H ion beams, the depth-dose curve sharply decreases after its *Bragg peak*, it does not in case of heavy ions, where a much smaller but non-zero remaining dose is still left and continuously tails off.

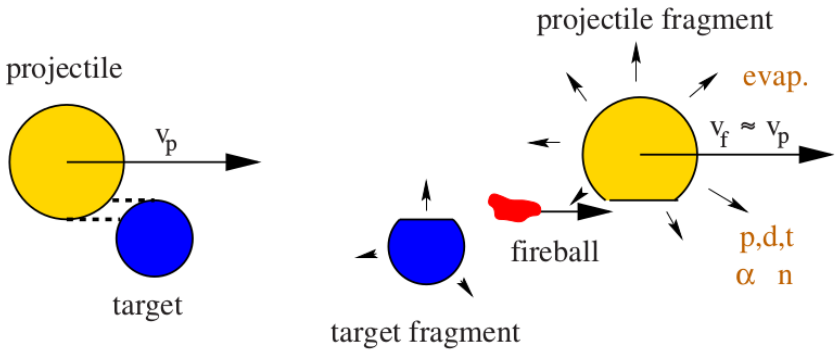


Figure 2.4: Nuclear fragmentation process: The projectile collides with an initial velocity v_p with the target, both are fragmented and produce secondary particles in a de-excitation process [4]

The mathematical background of fragmentation processes can be described by *abrasion-ablation* [4, 13]. It represents the macroscopic and geometrical basis for most of the implementation of nuclear interactions in numerical simulations. In the next few lines, in a short excursus, a summary about the assumptions made in this model will be given. For a more detailed description, [14] and [15] can be recommended. *

First, the expression *abrasion-ablation* has to be explained.

1. As mentioned above, a nucleus-nucleus collision produces excited states. In this model, they are called *prefragments* and in the step of ***abrasion***, they can be assumed as *deformed* (*abraded*). This step of the interaction process takes about $10^{-23} - 10^{-22}$ seconds.

*At this point, it should be noted, that this is one of many representations of the abrasion-ablation model. This one was implemented in other Monte Carlo codes [16, 17].

2. The ***ablation*** process includes the decay of the excitation and deformation. The prefragments de-excite by emitting (*ablating*) nuclear particles. In the result, there are particles with lower atomic mass number, projectile-like fragments and/or γ rays left. Here, the time frame is about $10^{-18} - 10^{-16}$ seconds.

Starting from the total interaction energy $\mathrm{E_{tot}}$ [14]

$$\mathrm{E_{tot}} = \frac{1}{2}\mu\dot{r}^2 + \frac{l^2}{2\mu r^2} + \frac{Z_p Z_t e^2}{r}, \tag{2.15}$$

with angular momentum l, reduced mass μ, distance between the nuclei r and its change rate $\dot{r}$, electric charge e and charge numbers of projectile and target Z_p and Z_t, respectively, one can derive the effects of heavy ion fragmentation. It is useful to define an impact parameter b by approaching the total energy as [14]

$$\begin{aligned} E_{tot} &= \frac{l^2}{2\mu b^2} \\ l^2 &= 2\mu b^2 E_{tot}, \end{aligned} \tag{2.16}$$

where the distance between the nuclei is at a minimum ($\dot{r} = 0$). Therefore the equation of energy can be rewritten as [14]

$$E_{tot} = \frac{E_{tot} b^2}{r^2} + \frac{Z_p Z_t e^2}{r}. \tag{2.17}$$

The impact parameter b is now defined by $b = r(r - \frac{Z_p Z_t e^2}{E_{tot}}) = r(r - r_m)$. In the collision process, prefragments (see above) are formed in abrasion. Depending on the nuclei, a certain fraction of the collision partners is abraded. The following formula allows to calculate the mass difference between the states before and after the collision [14]:

$$\Delta_{abr} = FA_p \left[1 - e^{-\frac{C_T}{\lambda}}\right], \tag{2.18}$$

with the fraction F of the projectile in the interaction zone, the mean free path λ, the energy E of the projectile in MeV/n, the mass number of the projectile A_p and the chord-length of the position in the target of maximum interaction probability C_T. The component $FA_p e^{-\frac{C_T}{\lambda}}$ takes into account the de-excitation and emitting of particles in the interaction zone, while FA_p describes the equilibrium before the collision.

The probability of a nucleon to be abraded from the projectile is proportional to the fraction of protons in the projectile [14]:

$$\Delta Z_{abr} = \Delta_{abr} \frac{Z_p}{A_p} \tag{2.19}$$

The wave function of the abraded nucleon can be written as [14]:

$$\psi(p) \propto \sum_{i=1}^{3} C_i e^{\frac{-p^2}{2p_i^2}} + d_0 \frac{\gamma p}{\sinh(\gamma p)}. \tag{2.20}$$

This wave function describes the number of produced secondary protons with momentum p per unit of momentum phase space. The C_i and d_0 are parameters resulting from the modelling and p_i are momenta with different lengths related to the Fermi surface.

Assuming that the energy that is needed to remove a nucleon from the projectile is $\mathrm{E_{abl}} \approx 10$ MeV [14], the number of nucleons emitted by ablation can be obtained by:

$$\Delta_{abl} = \frac{E_S + E_X}{10} + FA_p e^{-\frac{C_T}{\lambda}}, \tag{2.21}$$

with the excitation energy E_X and specific excitation energy in the surface area

E_S.

As angular distributions are of high physical importance, especially in the context of this present work, it should be noted that the emission of fragments after abrasion is assumed to be isotropically in the center-of-mass system and distributed as follows for the laboratory system [15]:

$$\arctan(\theta) = \frac{p_\perp(b)}{p_\parallel(b)}, \tag{2.22}$$

where $p_\perp$ and $p_\parallel$ represent the perpendicular and transverse momentum of the prefragments and b is the impact parameter. In conclusion, the abrasion-ablasion model can be seen as a simple way to compute the mechanisms of fragmentation from a macroscopic perspective. These calculations support an intuitive approach to comprehend the processes of nuclear interaction. In the next chapter, a compact overview will be given, how these processes are implemented in the simulation codes, used for the present work.

3. Materials and methods

Now that the physical background of ion beam therapy has been described, the present work comes to its more practical part. The first section of this chapter 3.1 is about the methods of Monte Carlo simulation. The execution of simulations require at least basic knowledge about the functionality of their numerics. For the following Section 3.2, the description as *practical* is more appropriate, but also here one will inevitably find theoretical parts, as the models of the underlying simulation code `FLUKA` have to be explained. However, the large part of this section is about the history, applications and the use of `FLUKA`. In the end of the chapter one will find a compact overview of high performance computing (see Section 3.3), the `FORTRAN` code of additional user routines for `FLUKA` (see Section 3.4) and the parameters used in the simulations for the present work (see Section 3.5).

3.1. Monte Carlo simulation

Monte Carlo methods are tools, originally developed in order to calculate multi-dimensional integrals numerically. Their applications can be found in many parts of natural sciences but also in the world of finance and economics. When experimental data are not easily or even not accessible, Monte Carlo methods represent an efficient way to get valuable results.

In principle developed in the 1930s – the well-known physicist Enrico Fermi had

already done preliminary theoretical work [18] – it took a long time to make use of Monte Carlo methods because their practicability is limited to applications on modern computers. One major application of Monte Carlo simulation in physics is found in the field of (classical and quantum) statistical mechanics, where it is essential to compute partition functions by integrating over many multidimensional integrals, such as position and momentum, e.g. The following sections and paragraphs are in the main based on scientific papers about Monte Carlo simulation and random numbers by Weinzierl [19] and Katzgraber [20], respectively.

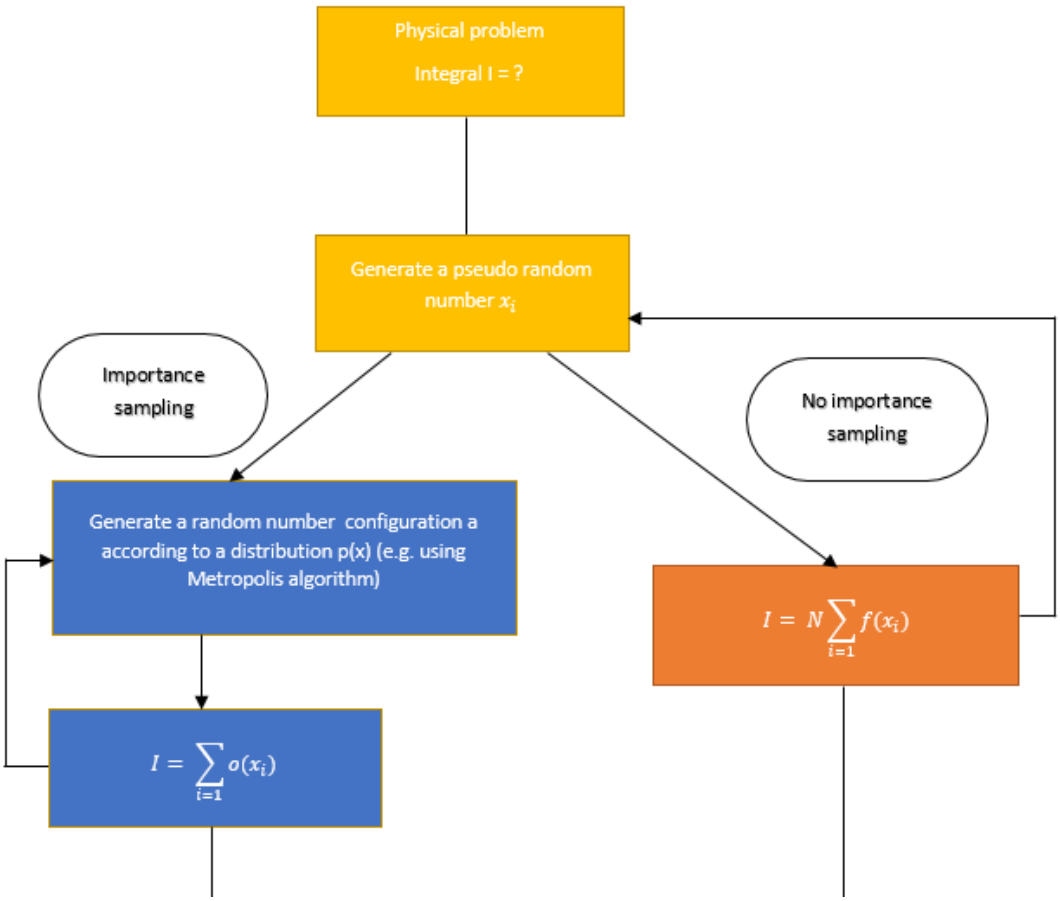


Figure 3.1: Structure of a Monte Carlo integration code

3.1.1. Principles and types of Monte Carlo simulation

After having defined the physical problem, a Monte Carlo simulation basically starts with the generation of a random number, which can be done on a personal computer.

As it will be explained in detail in Section 3.1.2, there are some criteria that make random numbers compatible with calculations using Monte Carlo methods. It can be seen in Figure 3.1, that there are two basic ways to make use of Monte Carlo methods.

First to mention is the probably most simple one, which is the simple integration without importance sampling (see Figure 3.1 right).

Here, the integral is approximated by a sum of the function values with evenly distributed random numbers x_i as arguments:

$$I = \int_{x_0}^{x_1} f(x)dx \approx \frac{1}{N} \sum_{i}^{N} f(x_i),\ x_i \in (x_0, x_1) \tag{3.1}$$

with the integrand $f(x)$ and the number of iteration steps N.

The calculation of the number π (see Figure 3.2) is one of the most descriptive examples for Monte Carlo simulations without importance sampling [21]: The computer generates a set of random numbers x and y, both less than or equal $|1|$. The circular number π is then computed by

$$\frac{\text{Number of sets, which fulfil } x^2 + y^2 \leq 1}{\text{Total number of sets}} \approx \frac{A_\circ}{A_\square} = \frac{\pi}{4}. \tag{3.2}$$

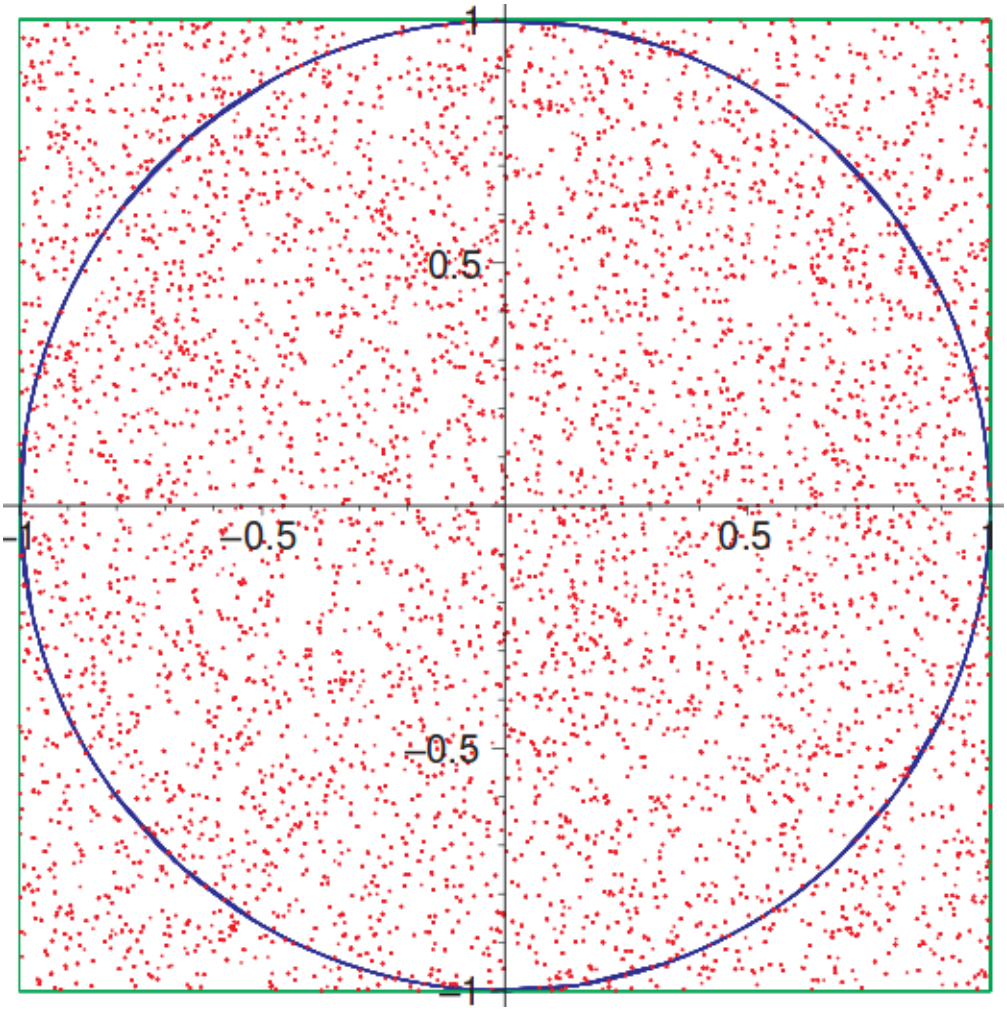


Figure 3.2: Monte Carlo simulation of the number π [21]

In general Monte Carlo simulations without importance sampling are appropriate for simple multidimensional functions.

Another possibility is a Monte Carlo simulation which considers the weight of every integrand region [19] (see Figure 3.1 left). If this method is taken, one has to separate the function, needed for the integration, in two multiplicative parts, see Equation (3.3).

$$f(x) = p(x)o(x) \tag{3.3}$$

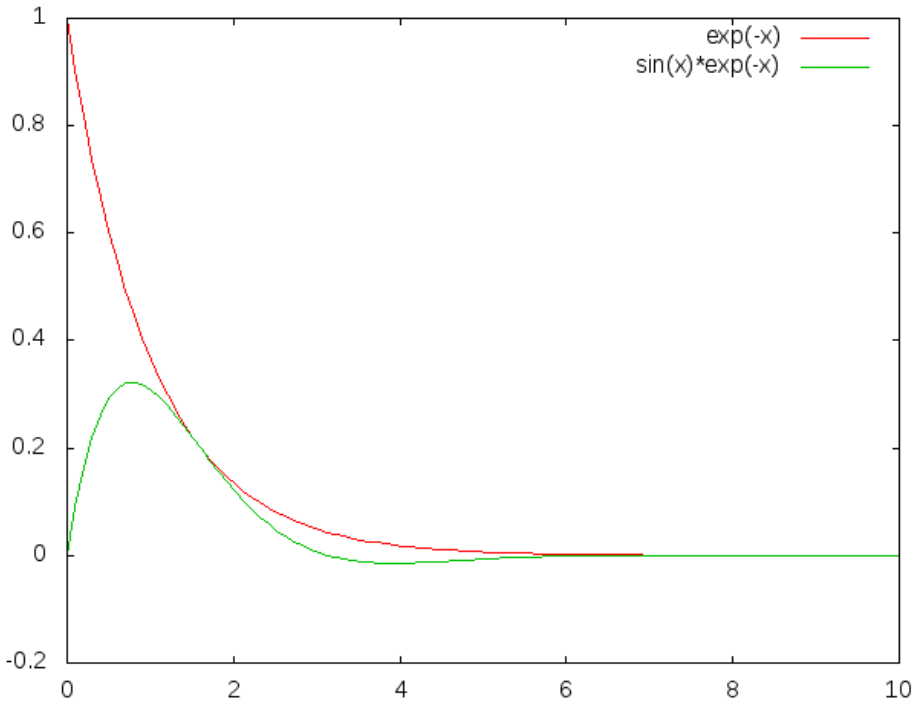
At this point, it is necessary to know the behavior of the function. The part of the function, which is closer in shape to $f(x)$, is now taken as a distribution $p(x)$ and has to be normalized. Subsequently, according random numbers are generated. In order to solve the integral, the following sum with the other part of the function, $o(x)$, as argument has to be computed in N iteration steps:

$$I = \int_{x_0}^{x_1} f(x)\,dx = \int_{x_0}^{x_1} p(x)o(x)\,dx \approx \frac{1}{N}\sum_{i}^{N} o(x_i), \tag{3.4}$$

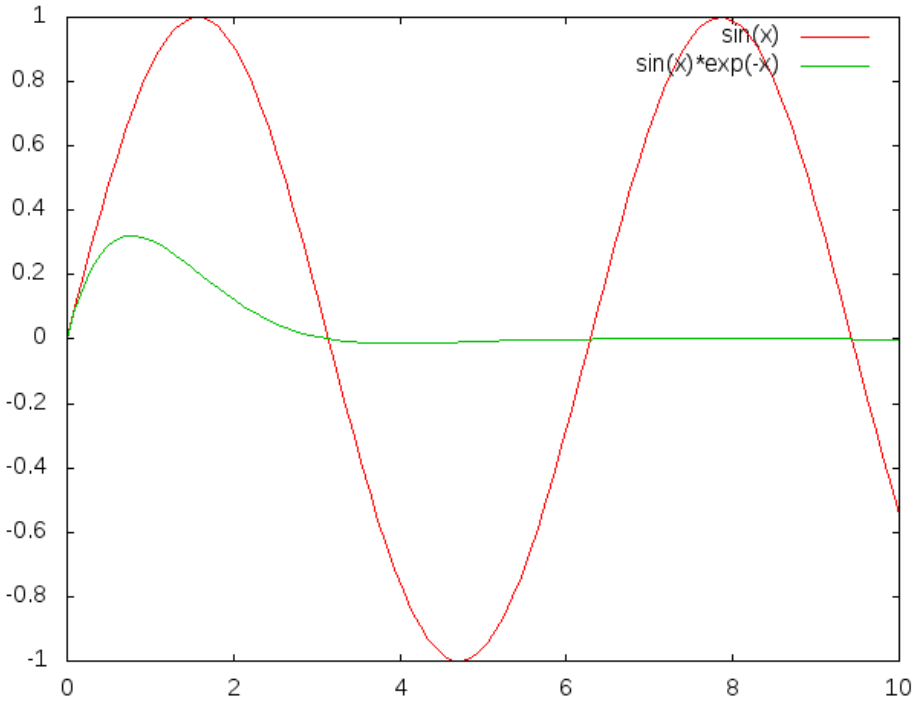
with random numbers $x_i \in (x_0, x_1)$ according to the distribution $p(x)$. In order to illustrate the practical use of this method, one can have a look at the following integration:

$$I = \int_0^5 \underbrace{\sin(x)}_{o(x)} \cdot \underbrace{e^{-x}}_{p(x)} dx \approx \frac{1}{N}\sum_{i}^{N} \sin(x_i), \tag{3.5}$$

with random numbers $x_i \in (0, 5)$ generated by e^{-x}. In Figure 3.3, it can be seen, that the exponential function in (a) is much closer in shape to the integrand than the sine function in (b). Therefore, the Monte Carlo integration is executed by generating random numbers x_i according to e^{-x} and summing up normalized values of $\sin(x_i)$.



(a) Comparison between $p(x) = e^{-x}$ and $f(x) = \sin(x)e^{-x}$



(b) Comparison between $o(x) = \sin(x)$ and $f(x) = \sin(x)e^{-x}$

Figure 3.3: Comparison of the integrand function in Equation (2.10) to its factors

3.1.2. Random numbers

In the previous sections, it was discussed that in the cycles of a Monte Carlo simulation, a random number is generated. The next few paragraphs will explain in a compact way the methods to generate random numbers.

Personal computers are able to generate random numbers but with certain limitations. The generation of random numbers is done by simple algorithms. So, after a finite number of calculation cycles based on them, one has to take into account that their quality decrease by the time because of the deterministic way to generate these numbers. Therefore the random numbers obtained by a normal personal computer, are so called *pseudo-random numbers*.

There are ways to generate natural random numbers (such as measuring the thermal noises of inner modules of electrical elements), but in the context of programming, it is useful to confine oneself to pseudo random numbers with high quality. There are some criteria, that define the goodness of random numbers [19].

- The numbers should be generated according to what one would expect from a random distribution. Furthermore an ideal set of random numbers should not show correlations.

- An algorithm, that generates random numbers, should be portable (e.g. between programming languages) and it should work on different hardware.

- Under real conditions random number generators show a periodicity. Hence, it is useful to think about the length of the period in order to avoid unwanted correlations.

- Although it is possible to produce random numbers of high quality, in applications, such as numerical simulations, the principle of proportionality must

be kept in sight. From there, one should minimize the effort of generating random numbers.

- As in every other experiment, calculations with random numbers have to be repeatable. If a computation cycle is repeated with the same numbers, the result has to be the same.
- Long disjoint subsequences

3.1.2.1. Pseudo random generators

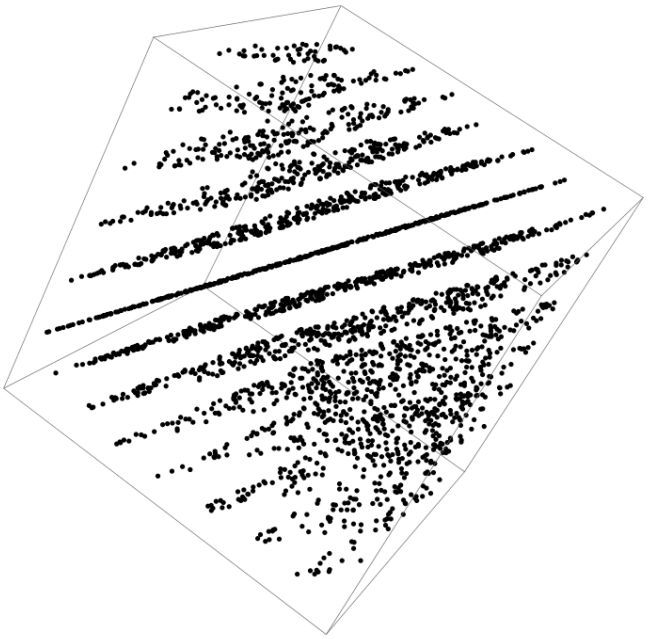


Figure 3.4: Successive random numbers generated by linear congruence [20], there should not be any visible planes, if the numbers are randomly distributed. In this case, the periodicity is too low to fully fill the cuboid with points determined by random numbers.

In order to increase the quality of computer generated random numbers one can use the linear congruential method [20], which is simple: First, a computer puts

out a random number in a defined area. Next, this value is put into the function

$$\begin{aligned} x_{i+1} &= (ax_i + b) \mod m. \\ r_i &= \frac{x_i}{m}, r_i \in [0, 1] \end{aligned} \tag{3.6}$$

This pseudo random number generator has a maximum periodicity of m. That is why one can adjust the goodness by scaling this parameter. On Figure 3.4 one can see, what happens, if m is taken too low.

3.1.2.2. Random numbers according to distributions

Inversion method In order to get random numbers according to any distribution $\tilde{y} = p(x)$, as it necessary for Monte Carlo methods with importance sampling (see Section 3.1.1), there are some ways to generate them. A seemingly very simple method is the inversion of the distribution [20, 19], thus,

$$p^{-1}(\tilde{y}) = x. \tag{3.7}$$

Although this can be an easy recipe to get random numbers for integrations, functions used for difficult multidimensional integrals often do not have a defined inverse. For example, the definition area of the tangent $\tan(x)$ is $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right)$, the square function x^2 is definite for the negative or positive branch of the x-axis. Therefore, in most of the cases, this method is not seen as practicable.

Metropolis algorithm One very often applied method to obtain random numbers according to a distribution function is the Metropolis algorithm [19]. It is an algorithm, which is based on a Markov Chain, a stochastic process depending on

the iteration step before. Thus, the conditional probability of a value x_{k+1} can be written as:

$$w(x_{k+1}|x_k, x_{k-1}, x_{k-2}, ..., x_0) = w(x_{k+1}|x_k). \tag{3.8}$$

As mentioned above, Monte Carlo methods are often used in statistical physics. The Metropolis algorithm is used in many applications in this context, because it allows an efficient calculation of multidimensional integrations.

3.2. FLUKA

FLUKA [1] (which stands for the German expression ***Fl****uktuierende* ***Ka****skade*) is a simulation tool, which is written in the programming language FORTRAN. It has been developed in the frame of research projects at CERN (*Conseil Européen pour la Recherche Nucléaire* or in English *European Organization for Nuclear Research*) in Geneva (Switzerland) and INFN (*Istituto Nazionale di Fisica Nucleare*) in Italy. Scientific contributions have also come from institutions in Leipzig (Germany) and Helsinki (Finland) [22]. It has been written to serve for many special applications, such as cosmic rays, neutronics, hadron therapy. Moreover a lot of interaction processes are implemented, e.g. nucleus-nucleus, hadron-nucleus, hadron-hadron and electromagnetic.

The FLUKA code makes use of Monte Carlo methods, which are introduced above in Section 3.1 and based on the simple principle of generating random numbers and thereby iterating multidimensional integrals.

3.2.1. History, structure and applications

FLUKA has been developed in the frame of a cooperation between many European research institutes [2]. Its history goes back to the 1960s, when the first Monte Carlo transport codes were developed at CERN. In this period the basis for FLUKA was provided. The stages of the development of the code will be described in the following passages, based on [1, 23]

In the decades after the start of FLUKA there were about three generations of the program, which have led to innovations in the field of Monte Carlo codes.

In the first generation, the applications in context of the CERN SPS project were essential for the development of FLUKA. The SPS (which means Super Proton

Synchroton) is a particle accelerator, that can achieve energies of about 300 GeV in a synchrotron system. In the construction phase the responsible research group needed relatively precise estimations for the description of hadronic inelastic interactions. Kaons, pions, nucleons and antiprotons were implemented in the program and also first approaches for the energy loss in ionisation processes and Coulomb scattering could be seen in this generation.

The second generation of the FLUKA code was characterized by many additional possibilities in the field of geometry. Furthermore, the developers wanted to find a way to create a more user-friendly interface for hadron cascades. For example, the FLUKA82 version was the first to include spherical and cartesian geometries.

Many innovations in the field of particle accelerators led to the development of the present FLUKA code generation. Because of the high energy ranges (e.g. 7 TeV on the Large Hadron Collider, *LHC*) and new designs another set of models and geometries was needed. The applications of FLUKA were broadened for other areas like neutrino physics, radiobiology, calorimetry, spallation sources and others.

3.2.2. Prerequisites

For a successful execution of a FLUKA simulation a Linux operating system is needed [23]. In the present case Ubuntu 16.10. and Windows 7 are used. Moreover one of the compilers gcc, gfortran (64 bit) and g77 (32 bit) respectively is required.

3.2.3. Physics in FLUKA

The FLUKA simulation package includes a variety of physical models [23] covering different energy ranges (see Table 3.1). In the next few lines a short overview including the main physical processes described in FLUKA will be given. The relevant theory for the present work will be described in the following sections and is explained as well above in 2.1 and 2.2.

Hadron-nucleon interactions are implemented in terms of two models. At momenta below 3–5 GeV/c, it is the *Generalized Intra-Nuclear Cascade* (GINC) and at higher energies the *Gribov-Glauber multiple collision mechanism* comes into play.

Also the transport of charged hadrons and muons is included in `FLUKA`. Besides, Bethe-Bloch theory and Mott corrected Rutherford scattering effect contribute to the simulation of charged particle transport. Moreover delta-ray production, spin effects and ionization fluctuations are taken into account.

`FLUKA` moreover contains a library for low energy neutron cross section for e.g. gamma ray generation or transport of proton recoils.

	Secondary particles	**Primary particles**
charged hadrons	1 keV – 20 TeV	100 keV – 20 TeV
neutrons	thermal – 20 TeV	thermal – 20 TeV
antineutrons	1 keV – 20 TeV	10 MeV – 20 TeV
muons	1 keV – 1000 TeV	100 keV – 1000 TeV
electrons	1 keV – 1000 TeV	70 keV – 1000 TeV (low Z)
		150 keV – 1000 TeV (high Z)
photons	100 eV – 10000 TeV	1 keV – 10000 TeV
heavy ions	$<$ 10000 TeV/n	$<$ 10000 TeV/n

Table 3.1.: Transport limits of `FLUKA` [23]

Nucleus-nucleus interactions generated by ions are represented by *Boltzmann Master Equations* (BME), *Relativistic Quantum Molecular Dynamics* (RQMD) and *Dual Parton Model* (DPM), which will be described in detail in the following sections.

3.2.4. Boltzmann Master Equations (BME)

The theory of *Boltzmann Master Equations* (BME) is based on a set of coupled equations. As it can be read in the two main papers about the implementation of this model [24, 25], it describes the *thermalization*, the process of continuously getting into thermal equilibrium after a collision of two ions.

In this model, the nucleus is considered as a two-fermion gas consisting of proton and neutron. The set of master equations are a time evolution of the occupation probability of definite neutron and proton states and thereby represent the relaxation process of the nucleus. The equations are numerically integrated up to a pre-equilibrium emission time and give as a result the occupation probabilities of states of the nucleus and the unbound particles, respectively [24].

In practice, a Monte Carlo code using Boltzmann Master Equations generates random numbers for the comparison (as outlined in Section 3.1.2) of the multiplicities and the energy of the fragment particles. At this point it is decided, whether a and what type of particle is evaporated by the nucleus. Moreover the energy is fixed.

3.2.5. Relativistic Quantum Molecular Dynamics (RQMD)

The model of *Relativistic Quantum Molecular Dynamics* (RQMD) is a Lorentz-invariant extension of *Quantum Molecular Dynamics* (QMD). For energies higher than 125 MeV/n `FLUKA` makes use of it. The following chapter will focus on a mathematical – and in the final paragraphs more intuitive – description of the underlying assumptions and is basically oriented on [26] and [27].

First, the model of QMD, which represents the non-relativistic basis of RQMD, needs to be introduced. In this theory, the nucleon states are described as Gaussian wave packets with width L:

$$\phi_i(\mathbf{r}) = \frac{1}{(2\pi L)^{3/4}} exp\left[-\frac{(\mathbf{r}-\mathbf{r}_i)^2}{4L} + \frac{i}{\hbar}\mathbf{r}\cdot\mathbf{p}_i\right], \tag{3.9}$$

where r_i and p_i are the centers of positions and momentum of i-th wave packet. The total wave function is a product of all the nucleon states

$$\Psi(\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3, ..., \mathbf{r}_n) = \prod_{i=1}^{n} \phi_i(\mathbf{r}_i) \tag{3.10}$$

As in the classical Hamiltonian theory, the time evolution of position and momentum are given by:

$$\begin{aligned} \frac{d\mathcal{H}}{d\mathbf{p}_i} &= \frac{d\mathbf{r}_i}{dt} \\ \frac{d\mathcal{H}}{d\mathbf{r}_i} &= -\frac{d\mathbf{p}_i}{dt} \end{aligned} \tag{3.11}$$

In this theoretical framework, the colliding ions collide, if the centers of their wave functions come closer than a distance $\sqrt{\frac{\sigma_{\text{tot}}}{\pi}}$. A Monte Carlo code would then be simulating a collision process with respect to the Pauli principle. Therefore only final states are not accessible for nucleons, if they are already occupied.

The QMD Hamiltonian can be written in the so-called *Skyrme* representation, where the overlap density of quantum mechanical states are summed up.

In order to transform QMD into a Lorentz invariant theory, the nucleons have to be described by their four momentum p_i^μ and four position q_i^μ [27], which means, that an N-particle system produces an 8N-dimensional phase space.

In the resulting Hamilton operator, phase-space and world line constraints are taken into consideration in order not to violate relativistic laws. The wave function for nucleus-nucleus interaction is then represented by the eigen function of this Hamiltonian. Since the present work is not meant to go too deep into theoretical details, one can find an explicit derivation in [27].

In Monte Carlo codes, the RQMD model is implemented similarly to the classical QMD. Besides the already mentioned reasons (*relativistic vs. non-relativistic* approach), the main difference between the application of both systems, is that in RQMD the violation of the relativistic world line constraints (oultined in [27]) would be conserved for a calculation cycle, which is not the case for QMD. Therefore only input that fulfils the constraints can lead to a valuable result [27].

3.2.6. Dual Parton Model (DPM)

Finally, the Dual Parton Model is introduced, which represents the basis of the event-generator `DPMJET`. The name comes from the assumption, that at high collision energies, nucleons show the behavior of smaller, hypothetical particles, called *partons*, which we today know as *quarks* and *gluons*. This theory is applied at high energies of about 5 GeV/n and higher and is based on the knowledge of quantum chromodynamics. Because in the present work, only particle energies up to 500 MeV/n, just a shord description will be given to describe the principles of this theory. In the theory of DPM, a hadron, seen as an excitation state, consists of an open string with quarks, antiquarks or diquarks at its end [28]. The collision between hadrons are causing the production of particles with low-transverse momenta. The fragmentation processes take into account that the produced particles move on straight trajectories in space and time. In a collision process, the so-called *prefragments* are stated to be in an equilibrium state, where the excitation energy is dissipated by evaporation of fragments. For more information about the DPM, see [29], the introductory work of Capella et al.

3.2.7. General input

Before `FLUKA` can be started, the user has to prepare the input for the simulations. `FLUKA` requires ASCII formatted "card"-like input files (*.inp) [23]. Because of the

fact that this simulation package has been developed for decades (and partially in times when punch cards were used for programming), the structure of the input is seemingly simple. This *simplicity* also involves problems. Every input option has a defined number of characters for its arguments (see Figure 3.5).

```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
BEAM            200.       0.2       1.5       1.2       0.7       1.0PROTON
```

Figure 3.5: Example of an ASCII input taking into account the card structure: The second line describes the property of an incident proton beam with an energy of 200 MeV, with an extent of Δx = 1.2 cm, Δy = 0.7 cm, a beam divergence of $\Delta\phi$ = 1.5 mrad and a flat momentum of Δp = 0.2 GeV/c.

An input file basically consists of the following options:

- A description of the experimental geometry
- Definition of the materials that should be included in the simulation
- Assignment of the materials to regions of the geometry
- A particle source (e.g. beam)
- Initialization of the random number sequence
- Starting signal and number of requested histories (which is the number of primary particles in principle)

These are the mandatory input parameters. There are many types of geometrical bodies that are available in `FLUKA`, for example rectangular parallelepipeds or ellipsoids. There are some predefined materials that can be used in this package, others can be created specified by the user. In the option of particle source there is a variety of types of beams that can be chosen. In this connection, one also has to define the geometrical measurements of the beam, its spread in Δx and Δy

directions. The initialization of a random number is required to start the Monte Carlo simulation (see Section 3.1).

Next to the mandatory input parameters there are many other options that can help to define the physical processes that should be included in the simulation. One essential type option is the detector. `FLUKA` has widely spread possibilities to measure physical quantities. In the present case, especially the *usryield* detector is of high importance. It can be used to estimate (double) differential yields, such as fluxes or cross sections. In the input file, one has to include this detector in the style of Figure 3.6, where it is outlined, that two successive lines are needed for the use of this option. Un this Figure, it is also shown, that with the exception of the detector name, only numbers can be found in both lines. These are assigned to certain parameters, which will be explained in the next paragraph. The first entry in the first line determines the first and second physical variable, the (double) differential yield is derived by. Subsequently, the particle type, the output unit, the two regions defining the boundary and a normalization factor have to be defined in the first line. In the second line, the upper and lower intervals for the scoring regions of both quantities, the bins for the first quantity and the type of (double) differential yield is determined. An example of such input lines is given in Figure 3.6: The first line starts with the number 1399, which stands for a derivation by the total energy in GeV and the polar angle in radiant, 13 stands for the particle type Pion, 21 is the suffix for the output file, 2.0 and 3.0 define the region of the particle and 1.0 is multiplicative factor for the normalization. The second line begins with 50.0 and 0.001 which are the limits for the energy in GeV, 100 defines the number of bins, and 3.13159265 and 0 is the angular interval.

```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7...+...8
USRYIELD      1399.0       13.       21.0       3.0       2.0       1.0TotPi+(E)
USRYIELD        50.0     0.001     100.03.14159265       0.0       3.0 &
```

Figure 3.6: Example of a typical *usryield* input line [23].

With regard to the input files, there is a variety of different other options, such as detectors or physical and statistical biasing, but since the present work is focussed on differential and double differential cross sections, the most substantial information has already been mentioned.

3.2.8. Run

A FLUKA run is started by writing */pathto/flutil/rfluka -N0 -MX inputfile.inp* in the command line [23]. */pathto/* represents the path on the computer leading to the directory of the FLUKA software environment. X is the number of cycles – in order to obtain better statistics for the simulation, it is more useful to choose less primary particles and more cycles of runs than the other way around. As to be expected, *inputfile* is the name of the input file.

```
================================ Running FLUKA for cycle # 1 ====================

Removing links

Removing temporary files

Saving output and random number seed

Saving additional files generated
     Moving fort.47 to /home/fasso/Fluka/test/example01_fort.47
     Moving fort.48 to /home/fasso/Fluka/test/example01_fort.48
     Moving fort.49 to /home/fasso/Fluka/test/example01_fort.49
     Moving fort.50 to /home/fasso/Fluka/test/example01_fort.50
     Moving fort.51 to /home/fasso/Fluka/test/example01_fort.51
```

Figure 3.7: FLUKA run started via Ubuntu terminal [23]: Temporary files are deleted, data of the results are output in *fort.xx* files.

The standard FLUKA executable is *flukahp*. In order to access physical models of higher primary energies, the library for the Dual Parton Model (see Section 3.2.6) and the Relativistic Quantum Molecular Dynamics (see Section 3.2.5), can be linked by writing *ldpmqmd* into the command line instead of *rfluka*. If the

primary beam does not consist of heavy ions, the *physics* option additionally has to be used in the input file to link the high energy library.

3.2.9. flair

`flair` (***FL****UKA* ***A****dvanced* ***I****nte****r****face*) [30] is a graphical user interface for the use of `FLUKA`. Because of the less uncomfortable practicability, relating to the format of the input files, it can be very helpful in the use of the simulation package and save a lot of time and effort (see Figure 3.8). The public version of `flair` was first released in 2007. `flair` is written in the programming language `Python` and uses the package `Tkinter` for the graphical interface. One of thee advantages of `Python` are according to Vlachoudis, one of the `flair` developers, is a *very clear syntax* [30]. When the program is started, it gets compiled and optimized by a pseudo-code assembly language and a virtual machine interprets the code. Regarding the plots, that are included in `flair`, the software package of `gnuplot` is implemented.

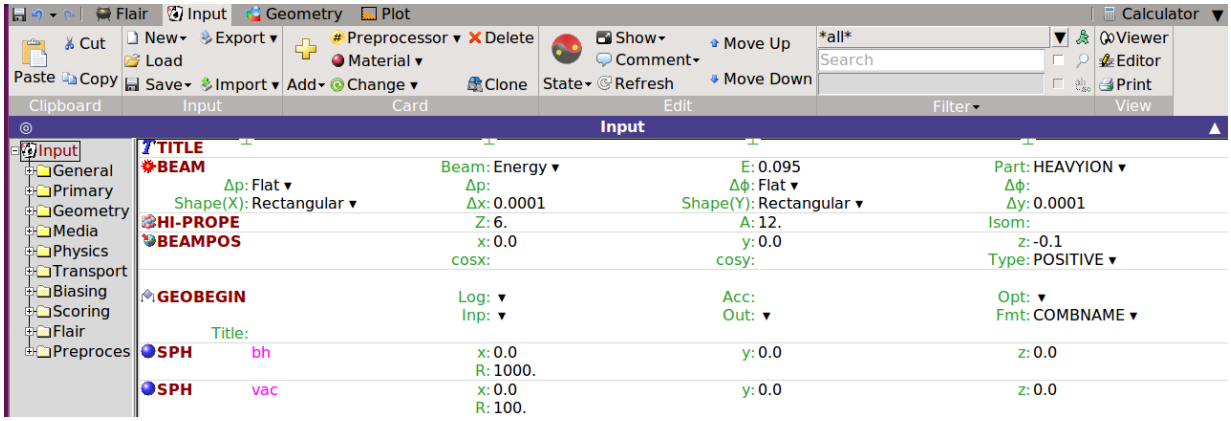
The user interface `flair` is working with the concept of `FLUKA` projects and contains many tools that are made to allow a more fluent work with the simulation package. It also creates its own file types (**.flair*) that log the cycles one has executed using `flair`. In the year 2010 the graphical user interface was advanced by the *Geometry Viewer*, which allows to illustrate the input geometry for the simulation (see Figure 3.13).

3.2.10. Post processing

When a `FLUKA` simulation is performed, four types of output files are automatically created [23].

- in the ***.err** files error messages can be seen in case of canceled cycles

(a) FLUKA input file using *gedit*



(b) FLUKA input file using `flair`

Figure 3.8: The use of `flair` can help to create input files: While the user of a *conventional* text editor in (a) has to take into account the well-defined structure of the ASCII input, it is automatically done in (b) by the user interface `flair`

- documents with extension ***.log** include messages of fatal errors (e.g. data overflow)
- ***.out** files are the standard output, they include additional information, such as properties of included detectors and *echo* the input file
- ***fort.xx** contains the results of estimator and detector computations assigned to the number *xx* in the input file.

The `FLUKA` *usryield* detector writes a *fort.xx* text file. It can be obtained in two ways, either in ASCII or in binary format. Both file types are output in the style of a matrix, where the binaries can be converted into a table (*_tab.lis file) of measurement data, including error analysis. Executing the *usysuw.f* `FORTRAN` tool, this conversion can be performed.

When *usysuw.f* was started in the command line, the exact name *fort.xx* files for every simulation cycle needed to be entered. This certain part of process was automatized using a simple `Expect` script, that runs through the numbered output data.

In order to plot the data in `Gnuplot` there is no necessity for another conversion, because the *_tab.lis is compatible with the software package. The units of the output data must be determined in the input file. In the present case of particles detected by *usryield* the results are given in units of b/(sr· GeV/n) for double differential cross sections and b/sr for differential cross sections.

3.2.11. Other simulation packages

Besides `FLUKA` there are some other Monte Carlo simulation codes. One of the most well-known is `GEANT4` [16, 17] (*GEANT* stands for ***Ge****ometry* ***an****d* ***T****racking*), which has also been developed in the *CERN* and is based on the programming

language C++. Another widely used code is PHITS [31, 32] (acronym for ***P**article and **H**eavy **I**on **T**ransport code **S**ystem*), which is FORTRAN based.

- GEANT4 was originally developed for the use in the context of the *Large Hadron Collider* (LHC) CERN project in Geneva. It has a complex structure that makes use of many different physical models. In a case relevant for the present work using ^{12}C projectiles, GEANT4 includes various codes. There is the possibility to use the semiclassical G4BinaryLightIonReaction (BLI), the QMD based (also see Section 3.2.5) G4QMDReaction and the Intra-Nuclear-Cascade Liège models. All of these have to be linked with a de-excitation model [11].

- PHITS can be used for many simulation applications, such as spallation neutron source, space radiation or heavy ion therapy. The models of nucleus-nucleus collisions are valid for delimited energy ranges. From 0 to 10 MeV/n, the ionization process is taken into account, above and up to 100 GeV/n, the JQMD (JAERI Quantum Molecular Dynamics) model is applied [32].

3.3. High performance computing

Figure 3.9: Logo of HTCondor [33]

Every `FLUKA` run needs a big amount of computer resources. Therefore, a simulation can take a lot of time and workload. In order to avoid inefficient work there are various tools for high performance computing. One is called `HTCondor` [34] (formally known as `Condor`), where HTC stands for *High Throughput Computing*, which is able to split a job into any number of *sub*jobs and to partition it for every single processor unit. This program was originally made for Unix but then basically expanded to Windows, Linux and Mac.

For the use of simulation interfaces, there are many scripts that implement the use of `HTCondor`. One was written in the programming language `Python` by Niels Bassler, a professor from the Department of Physics and Astronomy at Aarhus University in Denmark [35].

The `Python` code, mentioned above, was also used for the present work in order to reduce time effort. The `HTCondor` high performance computing tool parallelized the simulation jobs on up to 42 processor cores on a central server.

For the execution of the `Python` program (in the present case it was called *rcfluka.py*), the following line was written into the command line of the Ubuntu terminal:

```
/pathto/rcfluka.py -MXXX inputfile.inp -l ldpm3qmd -c,
```

where *XXX* stands for the number of simulations cycles, *inputfile* is the name of the input file, *-l ldpm3qmd* links `FLUKA` with the *DPM* and *RQMD* (see Figure 3.2.5 and 3.2.6) library and *-c* initiates the cleaning procedure for temporary files, which are automatically created during the calculations.

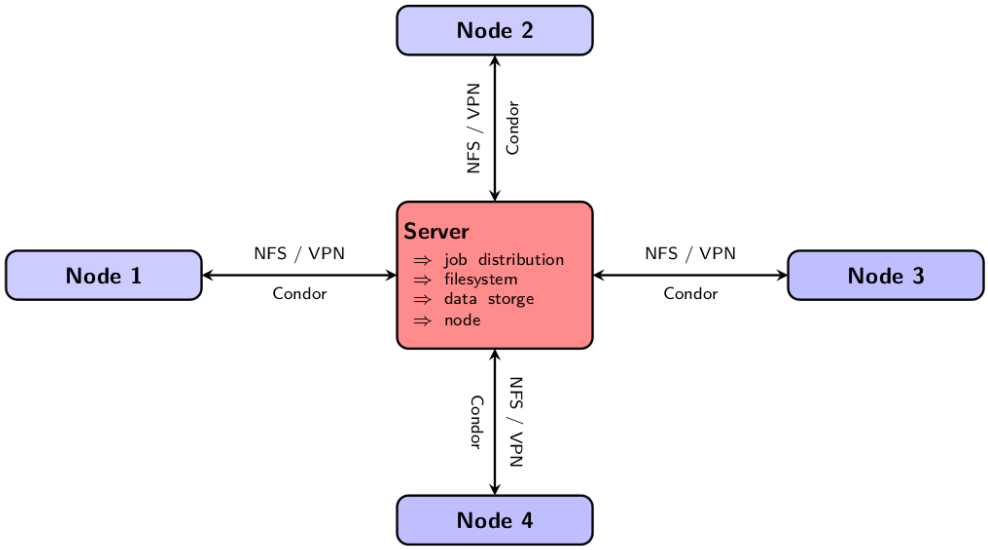


Figure 3.10: Structure of the grid (Image, courtesy of A. Hirtl)

Another aspect, which facilitated the execution of computations was the use of a server with 64 processor cores and two PC with 12 cores, such with internal memory of 256 GB. The scheme in Figure 3.10 shows the according basic working structure. The server, where also the required data files are stored, distributes the jobs to the nodes, where the calculations are executed. After this process the resulting data are sent back to the server.

3.4. Self-written codes

FLUKA contains predefined functions, that allow the simulation of differential and double differential cross sections. Although many particle types, such as elementary particles and light ions (^{1}H, ^{2}H, ^{3}H, ^{3}He, ^{4}He and primary particles) are accessible in FLUKA, some can only be examined by self written FORTRAN routines.

```
*$ CREATE MGDRAW.FOR
*COPY MGDRAW
*                                                                     *
*=== mgdraw ===========================================================*
*                                                                     *
      SUBROUTINE MGDRAW ( ICODE, MREG )

      INCLUDE '(DBLPRC)'
      INCLUDE '(DIMPAR)'
      INCLUDE '(IOUNIT)'
*
*----------------------------------------------------------------------*
*                                                                      *
*     Copyright (C) 1990-2006      by        Alfredo Ferrari           *
*     All Rights Reserved.                                             *
*                                                                      *
*                                                                      *
*     MaGnetic field trajectory DRAWing: actually this entry manages   *
*                                        all trajectory dumping for    *
*                                        drawing                       *
*                                                                      *
*     Created on   01 march 1990   by        Alfredo Ferrari           *
*                                              INFN - Milan            *
*     Last change  05-may-06       by        Alfredo Ferrari           *
*                                              INFN - Milan            *
*                                                                      *
*----------------------------------------------------------------------*
*
      INCLUDE '(CASLIM)'
      INCLUDE '(COMPUT)'
      INCLUDE '(SOURCM)'
      INCLUDE '(FHEAVY)'
      INCLUDE '(FLKSTK)'
      INCLUDE '(GENSTK)'
      INCLUDE '(MGDDCM)'
      INCLUDE '(PAPROP)'
      INCLUDE '(QUEMGD)'
```

Figure 3.11: Self-made screen shot of the first few standard lines of a FLUKA user routine, structured according to [23]

These self-written programs, used for this work, starts with a standard input (see Figure 3.11), that included certain functions of the FLUKA simulation package environment. The main part consists of FORTRAN *if*-operators, that ensure, that only particles with certain mass and charge numbers (dependent on which particle is studied) in the interaction region are taken into account for the computation.

In order to include these programs, there are FLUKA libraries that are linking the codes with the simulation interface. The input file then has to be expanded with the so called *userdump* option.

For every single fragment, an output file, also called *collision tape* [23], was created. It lists the recorded charge and mass numbers of the nuclei. Moreover, the kinetic energies of these fragments and the θ exit angle were obtained.

For the use of post-processing the data, another self-written program was executed. As the main part of the present work included codes in `FORTRAN`, it was also chosen as language for this purpose. The program starts with the initialization of every single variable.

The length of the output file name is dependent on the index of the cycle, that is computed (see Figure 3.12). In the program it is a *character* array with length n for cycles $0-9$ and $n+1$ for the cycles $10-99$. Since the number of recorded events differ from fragment to fragment, the variables, which describe the quantities in the *collision tape* (energy, angle, mass number, charge number), are dynamic arrays. The program detects their length by counting the numbers of lines in the output files.

```
if(j .lt. 10) then
        write(str1, "(i1)") j
        filename1 = 'neuesbsp00'//str1//'_Li-7_production.dat'
end if

if(j .ge. 10) then
        write(str2, "(i2)") j
        filename2 = 'neuesbsp0'//str2//'_Li-7_production.dat'
end if

if(j .lt. 10) open(UNIT = 44, FILE=filename1)
if(j .ge. 10) open(UNIT = 44, FILE=filename2)

k = 0
statx = 0
do while (statx == 0)
        read(44,*, iostat = statx) dummy
        k = k+1
end do
```

Figure 3.12: In this part of the program, first, the file name is read. Then the *do while* loop counts the numbers of lines in the output file.

The program's *core* consists of an *if*-loop, which counts the recorded events of the *collision tape* in bins of angle (and energy) for (double) differential cross sections.

Derived from the cross section formula in Equation (2.10), the differential and double differential cross sections were calculated as follows:

$$\frac{d\sigma}{d\Omega} = \frac{N_{\mathrm{fn}}}{N_{\mathrm{pp}} \times \rho_{\mathrm{t}} \times th \times 2\pi \times (\cos(\theta + \Delta\theta) - \cos(\theta - \Delta\theta))} \tag{3.12}$$

where the product of thickness th and density ρ needed to be input in units of *barn* (mostly abbreviated as b) and the 2π represents the ϕ angle component (also see it visualized in Figure 2.3). In the Equation (3.12), N_{fn} stands for the number of fragment nuclei and N_{pp} for the number of primary particles. The double differential cross section was computed by dividing the differential cross section by ΔE_{kin}, the energy interval.

3.5. Simulation parameters

The simulation of the differential and double differential cross sections were executed for heavy ion energies 120, 600, 1200, 1800, 2400, 3000, 3600, 4200, 4800, 5400 and 6000 MeV, in the angular regions of (2.5 ± 2.5)°, (7.5 ± 2.5)°, (12.5 ± 2.5)°, (17.5 ± 2.5)°, (22.5 ± 2.5)°, (27.5 ± 2.5)°, (32.5 ± 2.5)°, (37.5 ± 2.5)°, (42.5 ± 2.5)°, (52.5 ± 7.5)° and (75 ± 15)°. Moreover 16 different targets and 17 fragments were studied.

As it was explained in Section 3.3, a `Python` script was used to parallelize the simulation cycles. In [35], it is recommended, that a single computation job should not take more than 1 to 4 hours. Therefore, the `FLUKA` input files were kept simple and created for every single energy interval and target.

3.5.1. Geometry

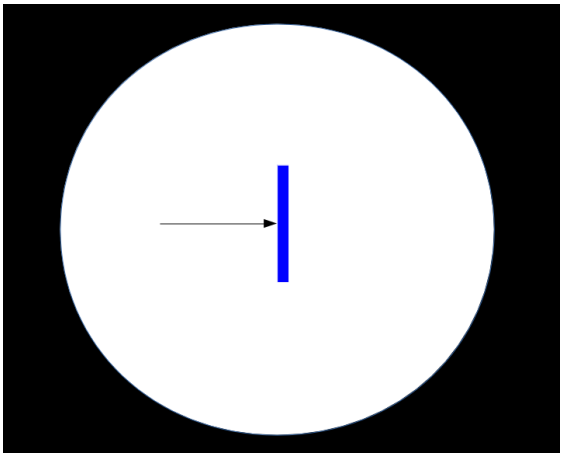


Figure 3.13: Geometry of the `FLUKA` simulation; not to scale: Black hole sphere – vacuum – target. The arrow represents the incident beam.

The geometry of the simulated setup in principle consists of (also see Table 3.2 and Figure 3.13):

- A *black hole* sphere in order to avoid unwanted backscattering effects from the boundary of the vacuum, containing
- a vacuum sphere centered at the zero point, which allows an unpolluted experimental environment, with
- a cuboid in the center, which stands for the target.

Geometry	$\Delta\mathbf{x}$ [μm]	$\Delta\mathbf{y}$ [μm]	$\Delta\mathbf{z}$ [μm]	$\mathbf{r}$ [cm]
Cuboid (target)	200	200	10	$*$
Sphere (vacuum)	$*$	$*$	$*$	100
Sphere (black hole)	$*$	$*$	$*$	1000

Table 3.2.: Extent of the geometrical bodies

3.5.2. Projectile

As mentioned in the beginning, ^{12}C ion beams work highly efficiently in the context of cancer therapy because of their depth-dose profile (see Section 2). Thus, they are important for the research at the MedAustron facility. In the simulation, every projectile beam contained 10^7 particles, multiplied by 100 cycles, thus, a single computation of heavy ion collision resulted in 10^9 primary particles. In Table 3.3, the properties of the incident ion beam are listed.

Shape	$\Delta\mathbf{x}$ [μm]	$\Delta\mathbf{y}$ [μm]	**Energies** $\left[\frac{\text{MeV}}{\text{n}}\right]$
Rectangular	0.1	0.1	10 – 500

Table 3.3.: Properties of the incident beam

It is obvious that the beam spread is much smaller than the extent of the target in x- and y-direction. In the setup of ion beam therapy the *targets* (i.e. cancerous

tumors surrounded by human tissue) are much larger and therefore boundary effects caused by too small targets need to be avoided in the simulation.

3.5.3. Targets

Particle	**State**	$\rho \left[\frac{\text{g}}{\text{cm}^3}\right]$
^{1}H	liquid	0.071
^{12}C	solid	1.644*
^{14}N	liquid	0.807
^{16}O	liquid	1.141
^{23}Na	solid	0.971
^{24}Mg	solid	1.738
^{27}Al	solid	2.699
^{31}P	solid	1.830
^{32}S	solid	1.960
^{35}Cl	liquid	1.563
^{39}K	solid	0.856
^{40}Ca	solid	1.550
^{48}Ti	solid	4.540
^{63}Cu	solid	8.690
^{120}Sn	solid	7.310
^{200}Hg	liquid	13.534

Table 3.4.: Targets used in the simulation ordered by mass number, state and density

*This value for the density ρ was taken as a continuation of the comparison to the `E600` (see above), where this was one of the parameters.

In order to get valuable data for the simulation set, a variety of different targets – liquid and solid – were taken. Elements that often appear in human tissue were of special interest. The basic component of organic material is carbon (C). Because of the amount of water, human body carries, also oxygen (O) and hydrogen (H) are included. The differential and double differential cross sections two elements were studied for the liquid state. Mercury (Hg), which was also part of the simulation, can be found in chemical bonds of dental amalgam fillings. Titan (Ti) is a component of some types of prostheses.

An overview about the three mentioned and other elements included in the simulation data set is given in Table 3.4.

4. Results and discussion

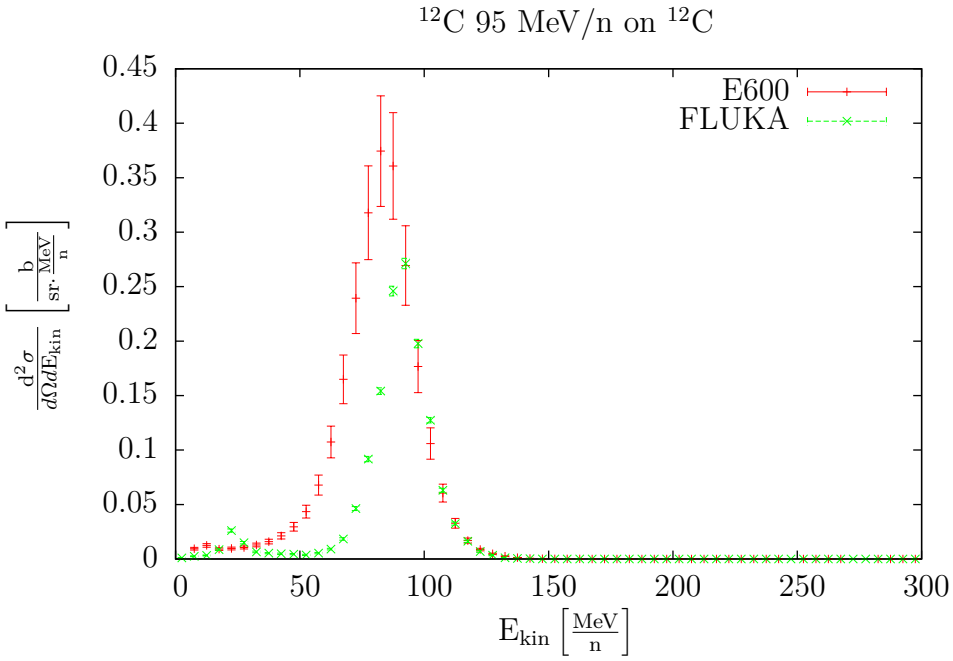


Figure 4.1: Simulation data of a preceding work compared to experimental data: ^{4}He double differential cross section for $\theta = 4°$

In the frame of a preceding work, it was shown that `FLUKA` simulation results agree to some extent with experimental data [36]. The differential and double differential cross sections were simulated for secondary ^{12}C particles and light fragments ^{1}H, ^{2}H, ^{3}H, ^{3}He and ^{4}He. A compact data set was calculated for ^{12}C primary particles with 95 MeV/n directed on a carbon target. The data were compared to the results of the `E600` [37] experiment and mostly showed a behavior, one could have

expected (see Figure 4.1).

Based on this conclusion, the results of the new simulation set are expected to be in a span that allows valuable estimations for the derivation of doses. In the present simulation, the differential and double differential cross sections of the fragments in table 4.1 were calculated. The resulting data set contains more than 35,000 tabular files, taking into account every energy and angle bin for every studied fragment and target.

Isotope	**Mass** [u]	**Half life**
^{12}**C**	12.077	stable
^{1}H	1.007	stable
^{2}H	2.014	stable
^{3}H	3.016	12.33 a
^{3}He	3.016	stable
^{4}He	4.001	stable
^{6}He	6.018	806.7 ms
^{6}Li	6.015	stable
^{7}Li	7.016	stable
^{7}Be	7.017	53.12 d
^{8}B	8.005	770 ms
^{9}Be	9.012	stable
^{10}Be	10,014	$1.51 \cdot 10^{6}$ a
^{10}B	10.013	stable
^{10}C	10.016	19.255 s
^{11}B	11.009	stable
^{11}C	11.011	20.39 min

Table 4.1.: Properties of the fragments included in the present FLUKA simulations

4.1. Differential cross section

First, the focus is placed on differential cross sections. Cross sections are effective planes, that give information about the scattering probability of a particle (also see Section 2.2). As mentioned above, a lot of targets have been used for the simulation and therefore also a variety of fragmentation processes can be observed. Next to the scattering probability, the differential cross section provides a distributional information about in which angular region a particle is deflected. In Section 2.3, where the background of fragmentation processes is summarized, it is also explained, that light fragments – contrary to primary particles – have higher probability to be found in regions of higher angles. This is one important criterion which will be deciding for the evaluation of the simulation quality.

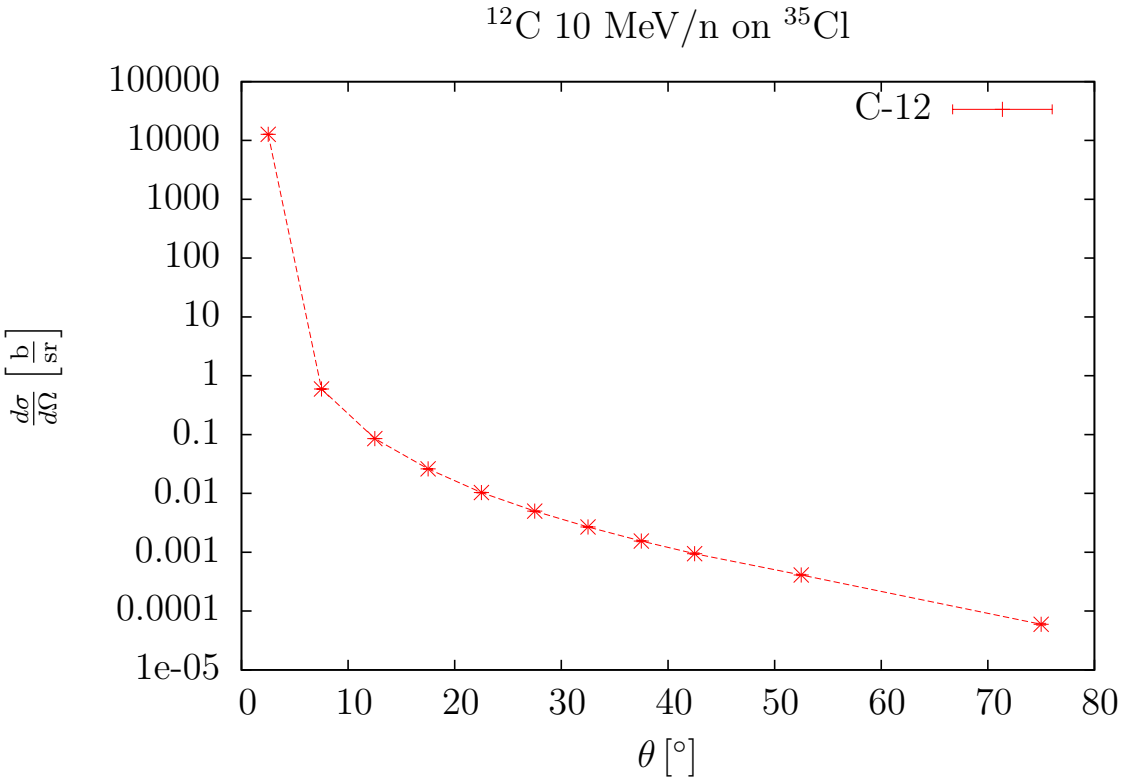
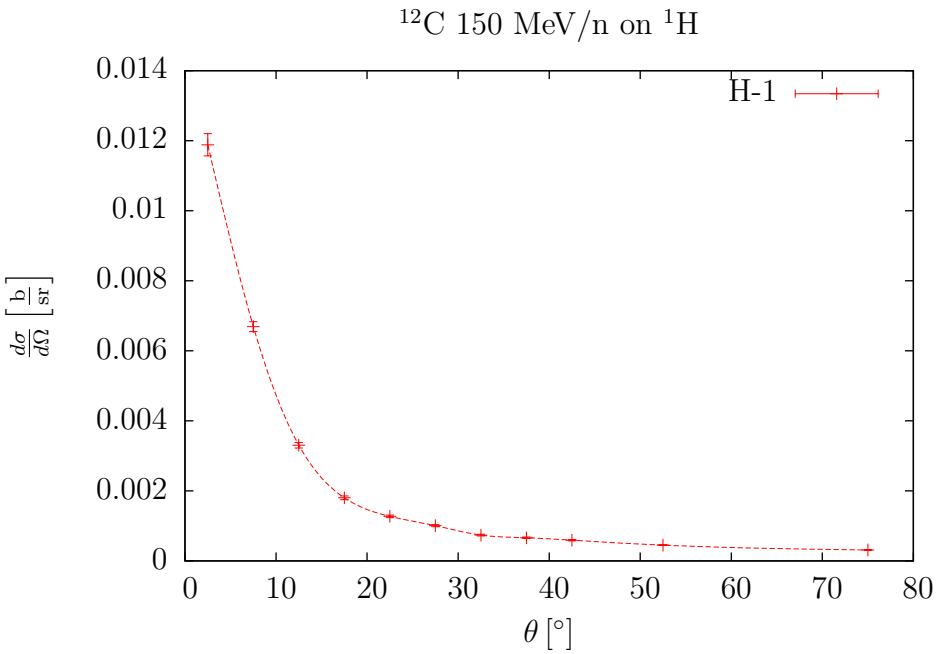
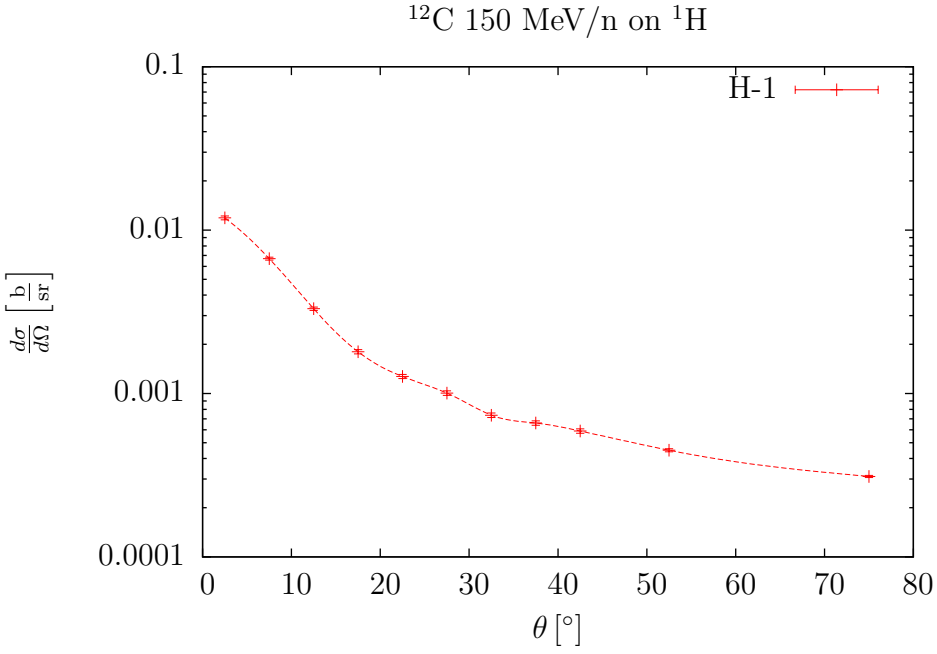


Figure 4.2: ^{12}C differential cross sections

In general, most of the differential cross section profiles show an exponential-like behavior. As it is obvious, a few words about the differential cross sections of the deflected primaries are given. One can see that the ^{12}C projectiles' differential cross section are high for low angles. For example in the case of a ^{35}Cl target, see Figure 4.2, there are two main characteristics. On the one hand, one can see that the value at an angle of $\theta = 2.5°$ is much higher than the next one at $\theta = 7.5°$, in this special case the difference is about four decimal powers. Results for other angles then only differ by a factor less than ten. Figure 4.3 shows the case of ^{1}H secondary production for a ^{1}H target. The small error bars may occur due to the fact, that the target is knocked out by the heavy primary and the secondary particle production by the projectile. As the target shows no characteristics of fragmentation in this case (see Section 2), this is not observed for the cases of other secondary particles. As one can see in Figure 4.4 `FLUKA` can give valuable data for a data set of fragmentation process simulations, whereby the quality of the results vary from fragment to fragment. ^{3}He and ^{4}He as lighter ions mostly show a meaningful statistical behavior, whereas in the mapped example, the fragments ^{6}Li and ^{7}Li have larger error bars.

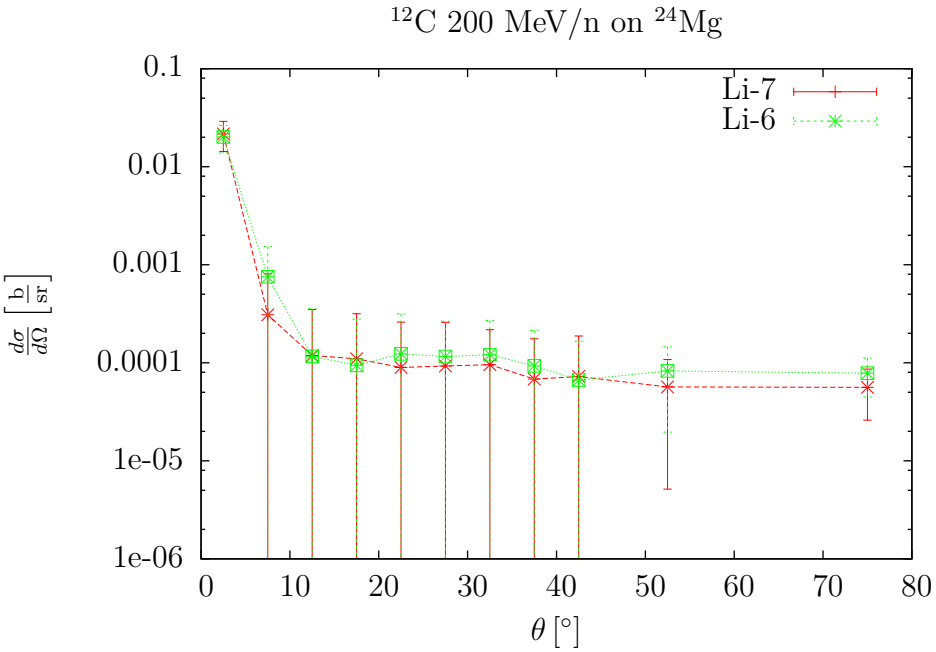


(a) linear scale

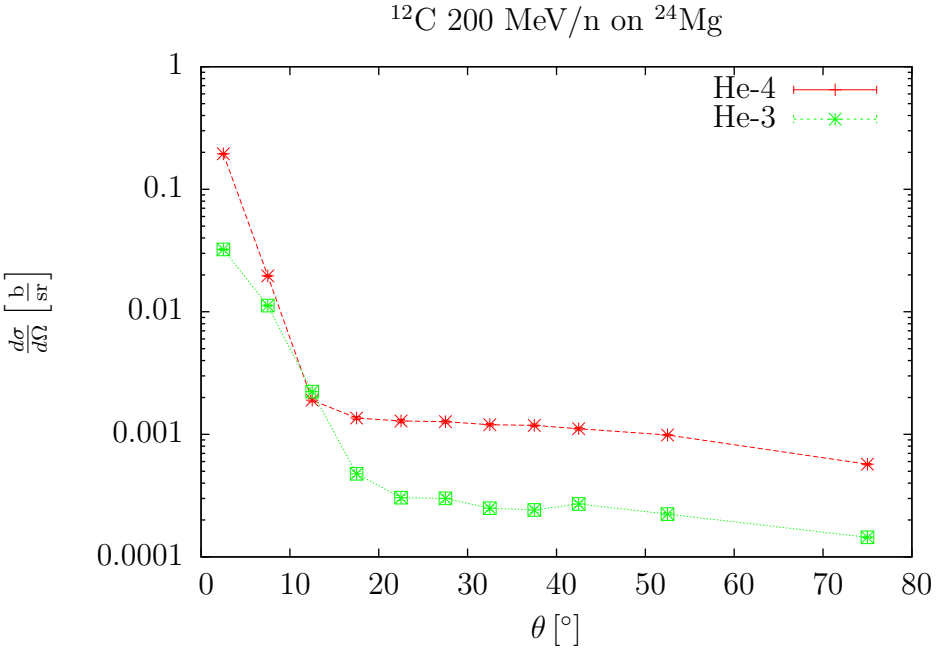


(b) logarithmic scale

Figure 4.3: ^{1}H differential cross sections.



(a) ^{6}Li and ^{7}Li differential cross sections on logarithmic scale



(b) ^{3}He and ^{4}He differential cross sections on logarithmic scale

Figure 4.4: Above, the data curves seem to show good agreement but contain very large errors. Below, the same energy and target were taken but different fragments – giving more precise results.

4.2. Double differential cross section

The simulations of double differential cross sections show in general a behavior, as follows: The center of the highest peaks in low angle areas is around the primary beam energy. For higher angles, it moves to smaller energy scales.

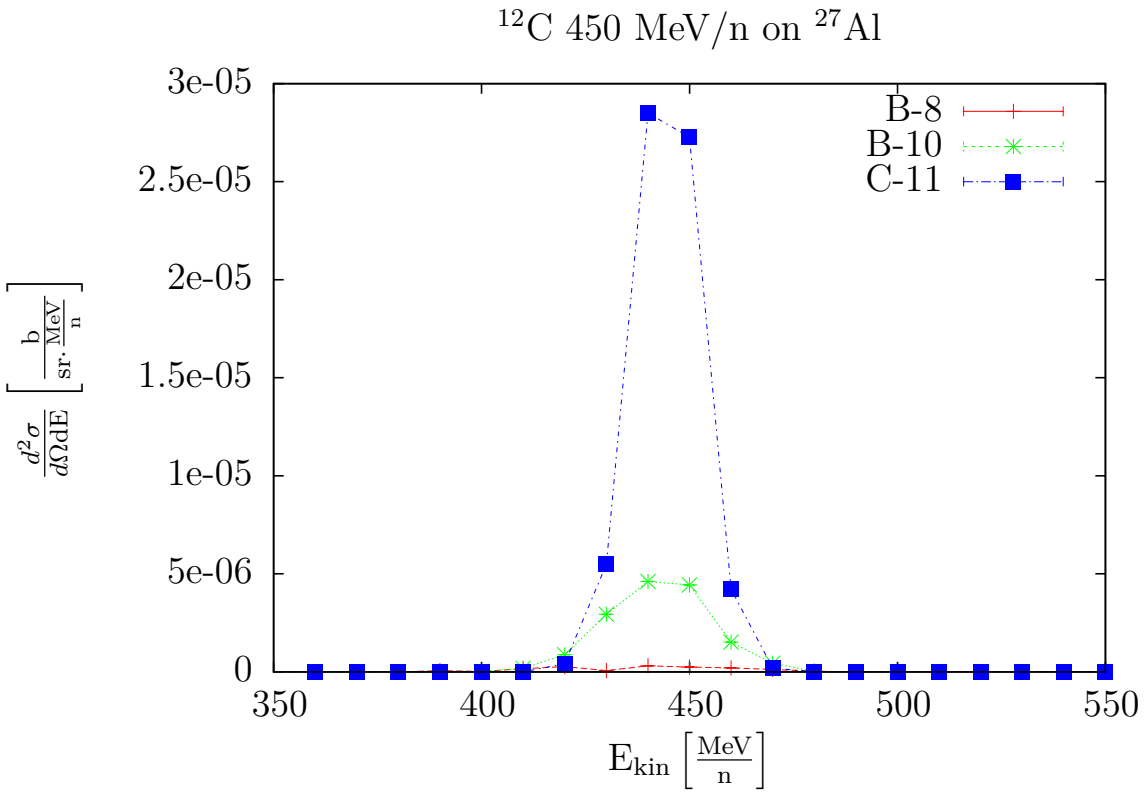
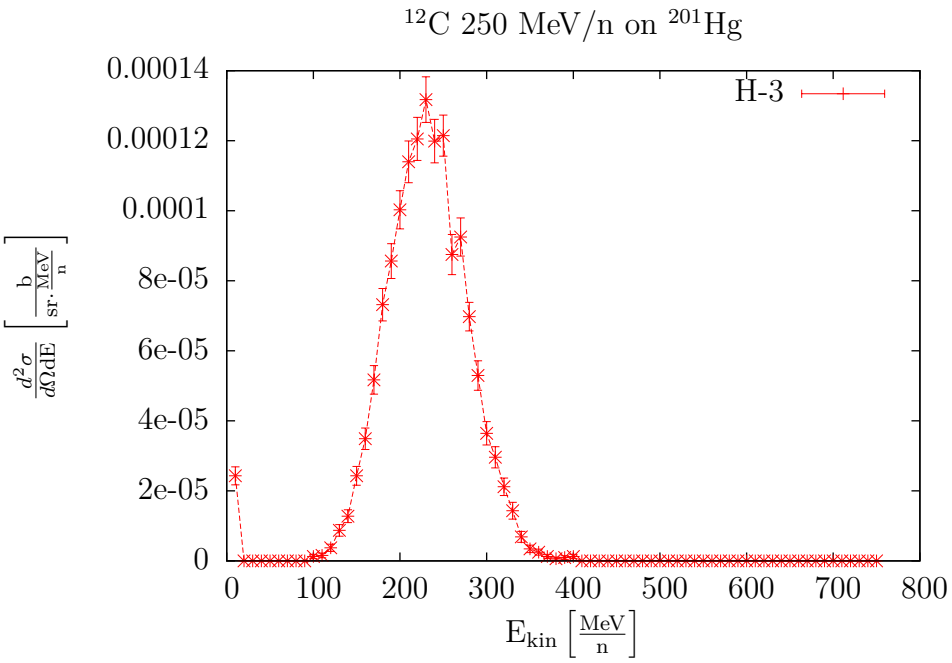
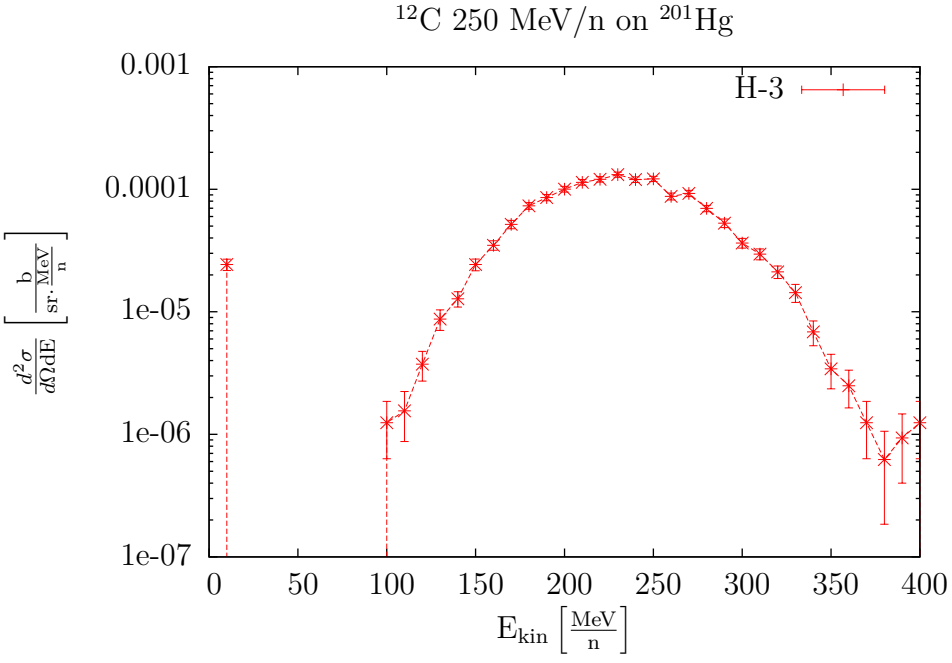


Figure 4.5: Double differential cross sections of ^{8}B, ^{10}B and ^{11}C compared at $\theta = 2.5°$

This is because of the *classical* behavior of the secondary particles, some of them lose energy and are deflected in the 4π angular regions. As one can see in Figure 4.5, the double differential cross sections can show a similar behavior for different particles, but the values can differ clearly.



(a) linear scale



(b) logarithmic scale

Figure 4.6: ^{3}H double differential cross sections at $\theta = 7.5°$

Moreover the fragments show a behavior depending on the target. While targets of higher mass numbers give good statistics for a variety of fragments, e.g. ^{3}H in Figure 4.6 for a ^{201}Hg target, ^{1}H as a lighter ion only gives valuable data for ^{12}C and ^{1}H cross sections. As it can be seen in Figure 4.7, the simulation of ^{4}He fragments, that usually shows good statistics, contains large error bars. A reason for that can be found in the fact that in the case of a ^{1}H target only the projectile is fragmented and therefore the scattering probability of ^{4}He is relatively low. Another argument for this statistical behavior is the density of liquid hydrogen, which his small compared to the other targets (about 1–3 decimal powers) and additionally lowers the probability of ^{4}He production in the collision process between target and projectile.

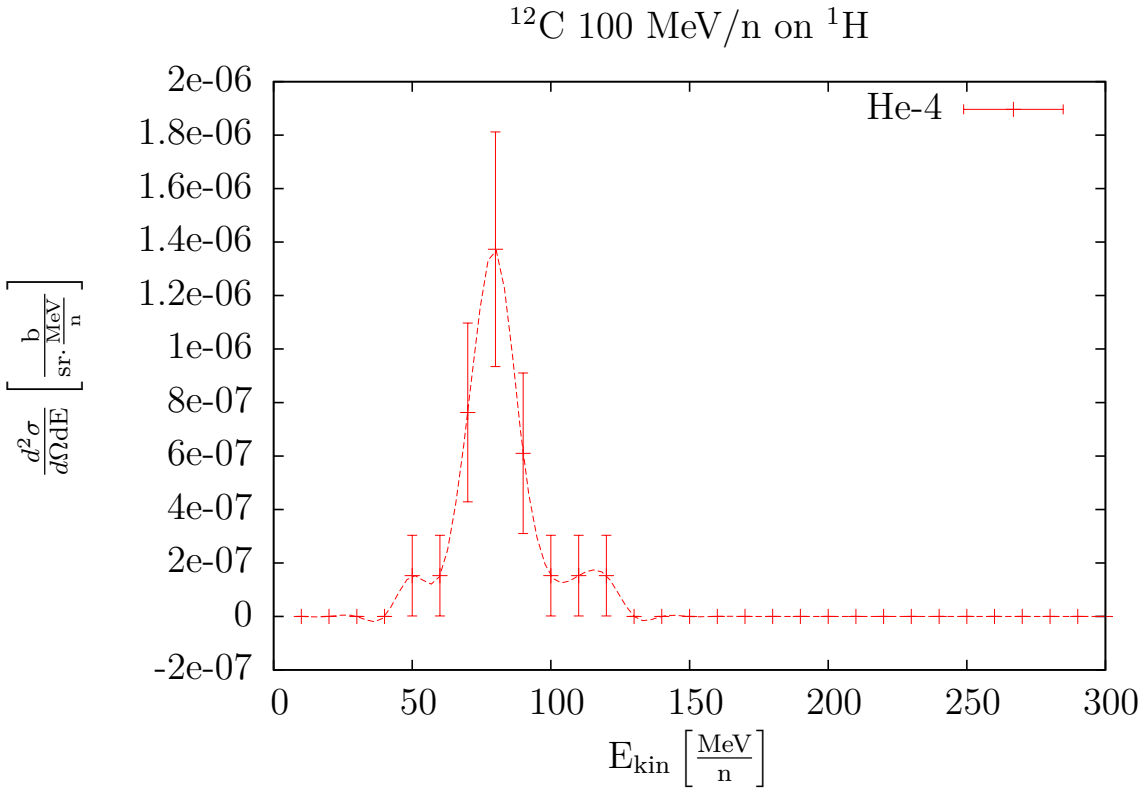


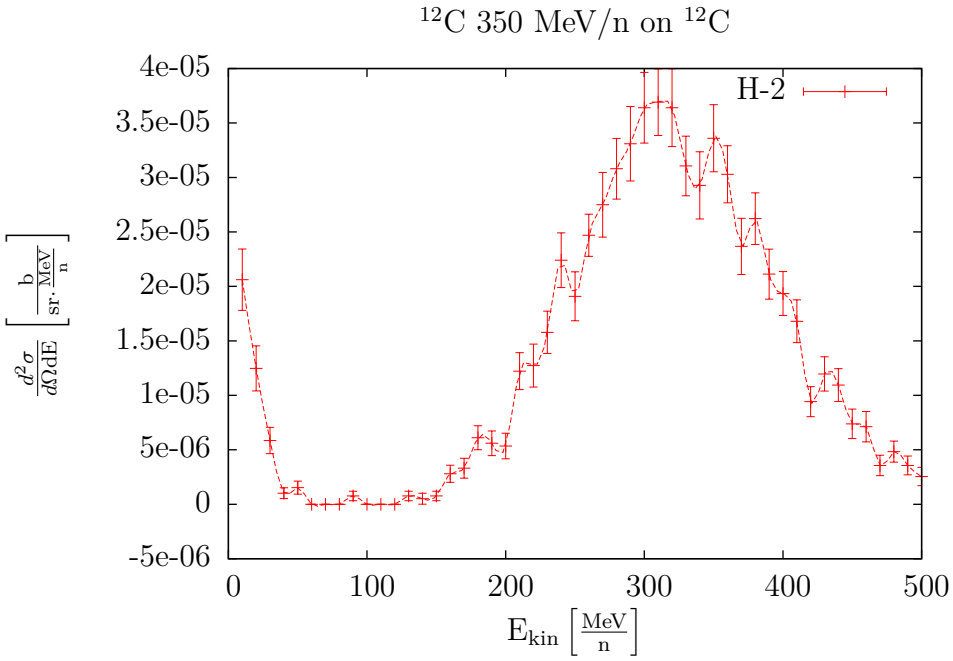
Figure 4.7: ^{4}He double differential cross sections at $\theta = 17.5°$.

In general, the highest values of double differential cross sections are found in the regions of low angles, as the behavior of the differential cross sections indicates. The more one approaches to $\theta = 90°$, the smaller the double differential cross sections get. In most of the cases, a *double peak* structure can be observed in the shape of the curve (see Table 4.2 and Figure 4.8). While the *main peak* – in low angle regions at the energy of the incident beam – gets smaller and moves to lower energies, there is another one of constant height at energies not much larger than 0 $\frac{\text{MeV}}{\text{n}}$.

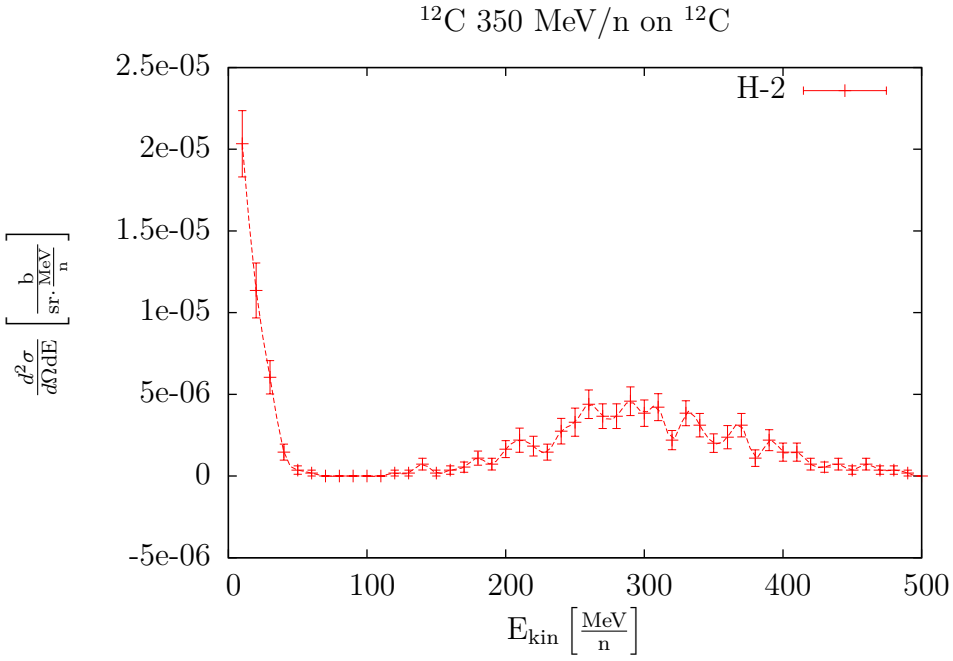
Angle [°]	$\mathbf{E}_1$ $\left[\frac{\text{MeV}}{\text{n}}\right]$	$\mathbf{E}_2$ $\left[\frac{\text{MeV}}{\text{n}}\right]$	$\frac{\mathbf{DDC}_1}{\mathbf{DDC}_2}$ [%]
2.5 ± 2.5	10	350	3.99
7.5 ± 2.5	10	340	13.09
12.5 ± 2.5	10	310	55.86
17.5 ± 2.5	10	290	444.00
22.5 ± 2.5	10	280	2263.36
27.5 ± 2.5	10	*	*
32.5 ± 2.5	10	*	*
37.5 ± 2.5	10	*	*
42.5 ± 2.5	10	*	*
52.5 ± 7.5	10	*	*
75 ± 15	10	*	*

Table 4.2.: ^{12}C directed on a ^{12}C target: ^{2}H double differential cross section behavior taken as example for the *double peak* structure; E_1 and E_2 are the energies for the two cross section main peaks, with $E_1 < E_2$.

Not often, but unfortunally sometimes it can happen, that results become *meaningless* due to statistics. In Figure 4.9 such a case is shown. The double differential



(a) $\theta = 12.5°$



(b) $\theta = 17.5°$

Figure 4.8: Comparison of double differential cross sections: The main peak decreases relatively to the first peak and is shifted to lower energies.

cross sections of three different particles (here ^{6}He, ^{9}Be and ^{10}C) are centered at the same point on average, but inserting error bars makes them lose their characteristics. Problems like this can appear because of modeling weaknesses or input of inappropriate parameters.

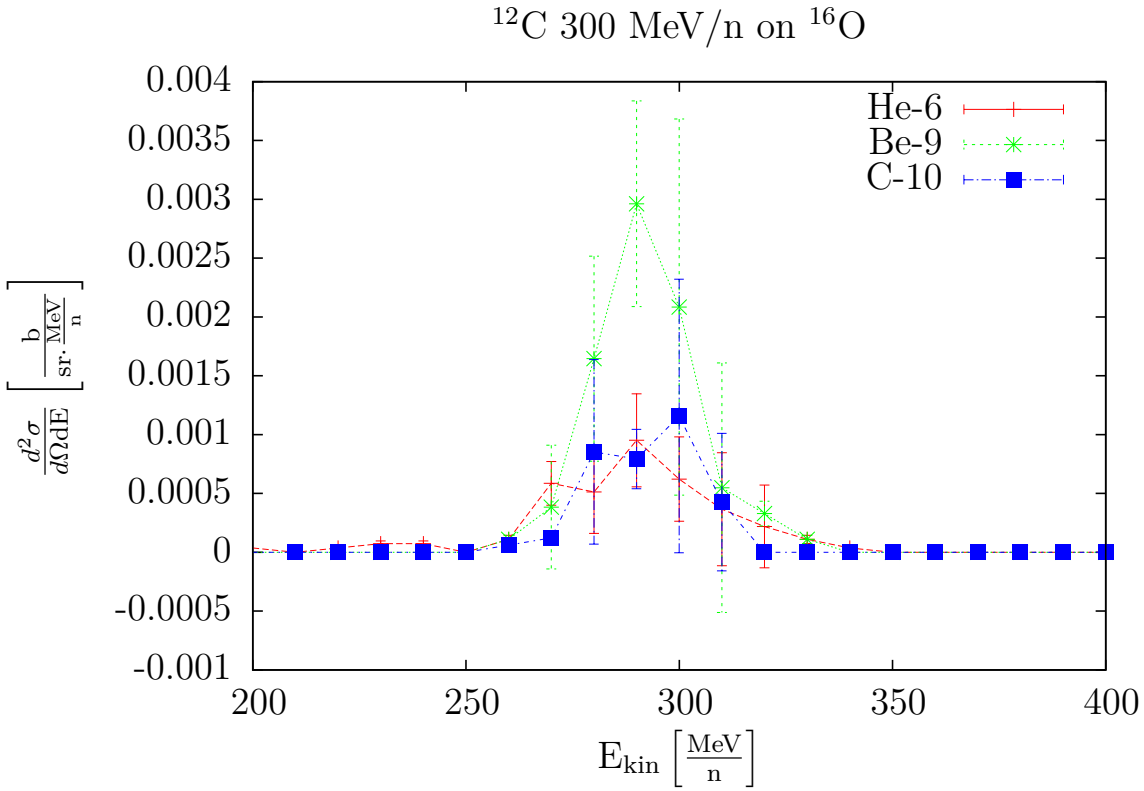


Figure 4.9: Double differential cross sections of ^{6}He, ^{9}Be and ^{10}C compared at $\theta = 2.5°$

5. Summary and conclusion

The present work started with an introduction into the field of ion beam therapy and other methods to treat cancerous tumors. While therapies using photons have led to respectable results but have also caused negative side effects, ion beam therapy has been developed in the last few decades and has shown much higher effectiveness. A short overview of the history is also given in the first chapter.

In order to outline a mathematical approach, the Bethe-Bloch formula, that describes the energy loss of heavy ions, was derived and explained in the next section.

The concept of differential, double differential and total cross sections was then introduced. The cross section is an effective surface that gives information about the scattering probability of a particle. The differential and double differential cross sections are physical quantities that also take into account that particles are deflected into different angular regions in various energy spectra.

A short derivation of the Coulomb / Rutherford scattering differential cross section was then presented. This – as a cumulated process – is another contribution to the heavy ion scattering, but relatively small compared to the stopping power in total.

Moreover the materials and methods of this work were introduced. In detail, the principles of Monte Carlo simulation and thereby also the generation of random numbers were explained, the `FLUKA` package was described and the actual input of

the simulations for this work was illustrated.

In the following chapter, the results of this work were presented in a compact style. One could see that there are certain characteristics and patterns in the behavior of differential and double differential cross sections.

It is appropriate to say that FLUKA delivers valuable data for certain setups with regards to ion beam therapy: Depending on input parameters, such as density, target element or the size of data bins, one can get comparatively precise results in terms of low statistical errors. For example, one could see, that certain targets cause production of certain particles or in cases of low mass targets there were less secondaries measurable.

However, for the further use of the data in therapy planning systems, it is considered advisable to compare with the results of simulations using other packages, such as PHITS and GEANT4. In the future it obviously also will be required to gather more experimental data – if then available – and re-estimate the precision of Monte Carlo simulated data, in order to produce a more accurate outcome. Furthermore, it will be necessary to develop the simulation models in order to reduce the gap between numerical calculations and real experiments. The outcome of this work can be taken as an improvement for the modelling of simulations, as it shows, that the results are not sufficiently precise. In the future, there might be opportunities to use numerical computations of differential and double differential cross sections as input for treatment planning systems.

Bibliography

[1] T.T. Böhlen, F. Cerutti, Fassò A. Chin, M.P.W., A. Ferrari, P.G. Ortega, A. Mairani, P.R. Sala, G. Smirnov, and V. Vlachoudis. The FLUKA Code: Developments and Challenges for High Energy and Medical Applications. *Nuclear Data Sheets*, 120:211–214, 2014.

[2] Ferrari, A. and Sala, P. R. and Fassò, A. and Ranft, J. *FLUKA: A multi-particle transport code (program version 2005).* CERN, Geneva, 2005.

[3] Wilson, Robert R. Radiological Use of Fast Protons, jun 1946.

[4] D. Schardt, T. Elsässer, and D. Schulz-Ertner. Heavy-ion tumor therapy: Physical and radiobiological benefits. *Rev. Mod. Phys.*, 82:383–425, Feb 2010.

[5] C. Thiel and R. Böhm. Fortgeschrittenen Praktikum Physik - Seminar.

[6] Wagner, A. and Böhm, R. Energiemessung & Teilchenidentifikation (Szintillatoren, Čerenkov), 2006.

[7] W. Demtröder. *Experimentalphysik 3: Atome, Moleküle und Festkörper.* Fourth edition, 2009.

[8] Gross, R. Physik IV – Atome, Moleküle, Wärmestatistik, 2003.

[9] Winkler, A. Seminarvortrag: Wirkungsquerschnitte und Formfaktoren, jun 2007.

[10] Ratz, M. Chapter II - Elemente der Streutheorie, 2011.

[11] A. C. Kraan. Range verification methods in particle therapy: underlying physics and Monte Carlo modelling. *Frontiers in Oncology*, 5(150), 2015.

[12] D. Schardt, I. Schall, H. Geissel, H. Irnich, G. Kraft, A. Magel, M. F. Mohar, G. Münzenberg, F. Nickel, C. Scheidenberger, W. Schwab, and L. Sihver. Nuclear fragmentation of high-energy heavy-ion beams in water.

[13] R. Serber. Nuclear Reactions at High Energies. *Phys. Rev.*, 72:1114–1115, Dec 1947.

[14] GEANT4. *Physics Reference Manual*. CERN, 10.2 edition, dec 2015.

[15] Sihver, L. and Giacomelli, M. and Ota, S. and Skvarc, J. and Yasuda, N. and Ilic, R. and Kodaira, S. Projectile fragment emission angles in fragmentation reactions of light heavy ions in the energy region; 200 MeV/nucleon: Experimental study. *Radiation Measurements*, 48:73 – 81, 2013.

[16] S. Agostinelli, J. Allison, K. Amako, J. Apostolakis, and H. Araujo et. al. Geant4—a simulation toolkit. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 506(3):250 – 303, 2003.

[17] J. Allison et. al. Geant4 Developments and Applications. *IEEE Transactions on Nuclear Science*, 53(1):270–278, 2006.

[18] Anderson, H.L. Metropolis, Monte Carlo and the MANIAC. *Los Alamos Science*, (14):96 – 108, 1986.

[19] S. Weinzierl. Introduction to Monte Carlo methods. *NIKHEF*, jun 2000.

[20] H. G. Katzgraber. Random Numbers in Scientific Computing: An Introduction. *eprit arXiv*, 1005.4117, may 2010. lecture at the second international summer school "Modern Computation Science", 9-20 August 2010, Oldenburg (Germany).

[21] D. Thiebaut. A Monte-Carlo Calculation of Pi.

[22] A. Fassò, A. Ferrari, S. Roesler, J. Ranft, P. R. Sala, G. Battistoni, M. Campanella, F. Cerutti, L. De Biaggi, E. Gadioli, M. V. Garzelli, F. Ballarini, A. Ottolenghi, D. Scannicchio, M. Carboni, M. Pelliccioni, R. Villari, V. Andersen, A. Empl, K. Lee, L. Pinsky, T. N. Wilson, and N. Zapp. The FLUKA code: present applications and future developments. *ArXiv Physics e-prints*, June 2003.

[23] A. Fasso', A. Ferrari, and J. Ranft. Fluka: a multi-particle transport code (Program version 2014).

[24] M. Cavinato, E. Fabrici, E. Gadioli, and E. Galbiati. Monte Carlo calculations using the Boltzmann Master Equation theory of nuclear reactions. *Physics Letters B*, 382:1–5, 1996.

[25] M. Cavinato, E. Fabrici, E. Gadioli, and E. Risi. Ejectile angular distributions in Boltzmann master equation theory of nuclear reactions. *Physics Letters B*, 405:219–223, 1997.

[26] K. Niita, S. Chiba, T. Maruyama, T. Maruyama, H. Takada, T. Fukahori, Y. Nakahara, and A. Iwamoto. Analysis of the (N,xN') reactions by quantum molecular dynamics plus statistical decay model. *Physical Review C*, 52:2620–2635, November 1995.

[27] E. Lehmann, Rajeev K. Puri, Amand Faessler, G. Batko, and S.W. Huang.

Consequences of a covariant description of heavy-ion reactions at intermediate energies. *Physical Review C.*, apr 1995.

[28] A. Ferrari and P. Sala. The Physics of High Energy Reactions, apr/may 1996.

[29] A. Capella, U. Sukhatme, C.-I. Tan, and J. Tran Thanh Van. Dual parton model. *Physics Reports*, 236(4):225 – 329, 1994.

[30] V. Vlachoudis. FLAIR: A Powerful But User Friendly Graphical Interface For FLUKA. 2009.

[31] T. Sato, K. Niita, N. Matsuda, S. Hashimoto, Y. Iwamoto, S. Noda, T. Ogawa, H. Iwase, H. Nakashima, T. Fukahori, K. Okumura, T. Kai, S. Chiba, T. Furuta, and L. Sihver. PHITS: Particle and Heavy Ion Transport Code System, Version 2.23. 2010.

[32] Koji Niita, Tatsuhiko Sato, Hiroshi Iwase, Hiroyuki Nose, Hiroshi Nakashima, and Lembit Sihver. PHITS—a particle and heavy ion transport code system. *Radiation Measurements*, 41(9–10):1080 – 1090, 2006. Space Radiation Transport, Shielding, and Risk Assessment Models.

[33] University of Wisconsin – Madison Department of Computer Sciences. HTCondor logo. Website.

[34] Center for High Throughput Computing, University of Wisconsin–Madison. *HTCondor Manual*, 8.5.6 edition, jul 2016.

[35] Bassler, N. Parallelizing FLUKA with CONDOR, 2009.

[36] D. Ellmeyer, A. Hirtl, and L. Sihver. Projektarbeit: Monte Carlo simulation of cross sections using FLUKA. 5 2015.

[37] J. Dudouet, D. Juliani, M. Labalme, D. Cussol, J. C. Angélique, B. Braunn, J. Colin, C. Finck, J. M. Fontbonne, H. Guérin, P. Henriquet, J. Krimmer, M. Rousseau, M. G. Saint-Laurent, and S. Salvador. Double-differential fragmentation cross-section measurements of 95 MeV/nucleon ^{12}C beams on thin targets for hadron therapy. *Physical Review C*, 88(2):024606, August 2013.

A. Appendix

A.1. Differential cross sections

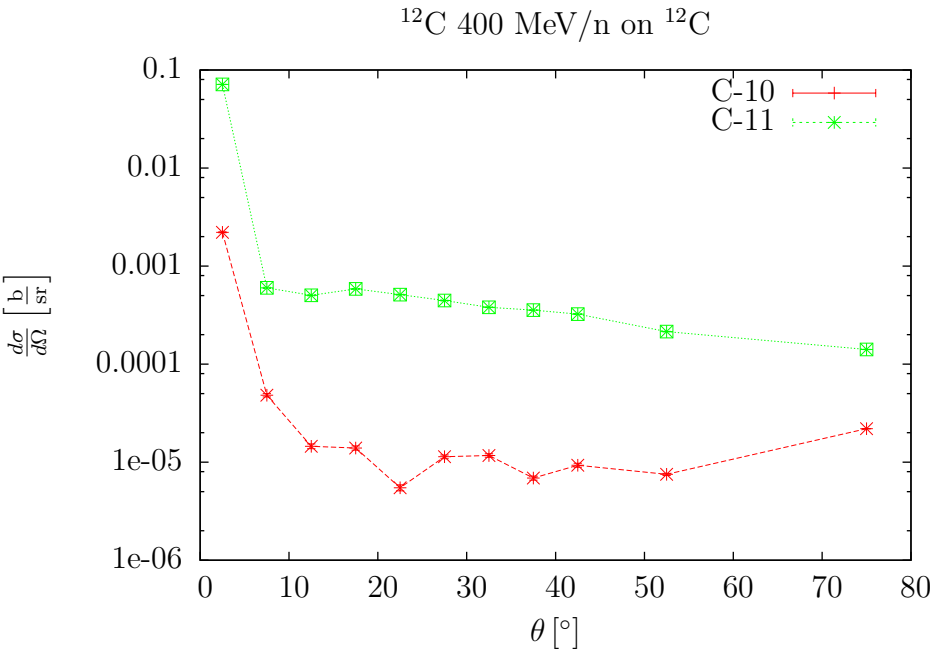
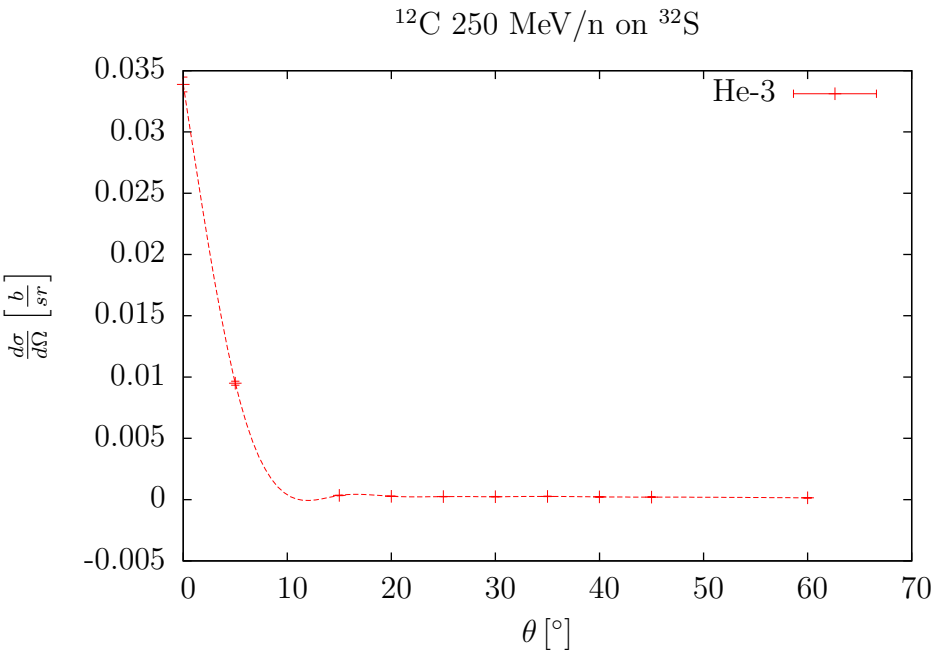
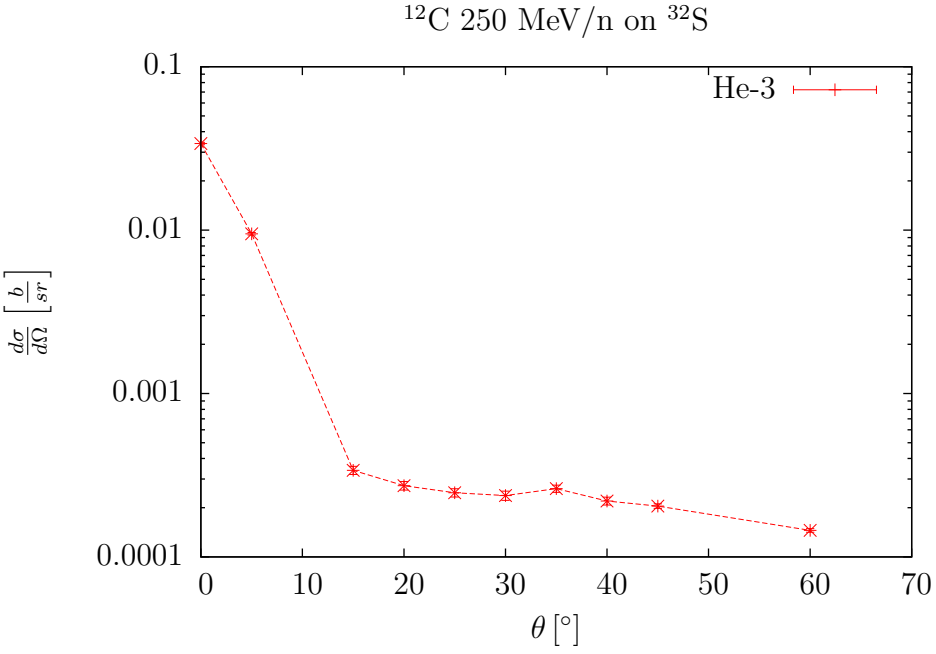


Figure A.1: ^{10}C and ^{11}C differential cross sections on logarithmic scale

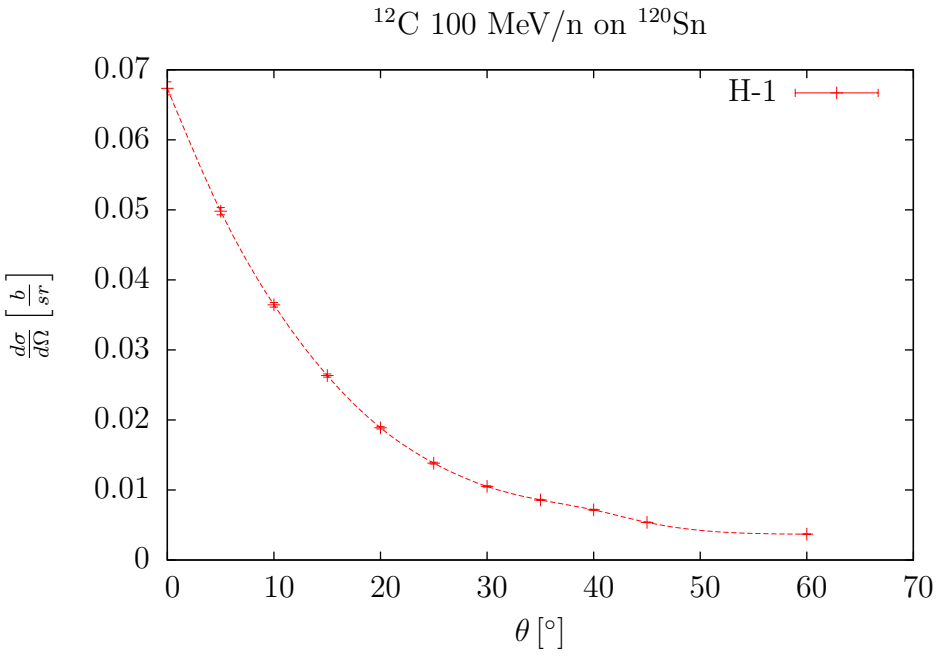


(a) linear scale

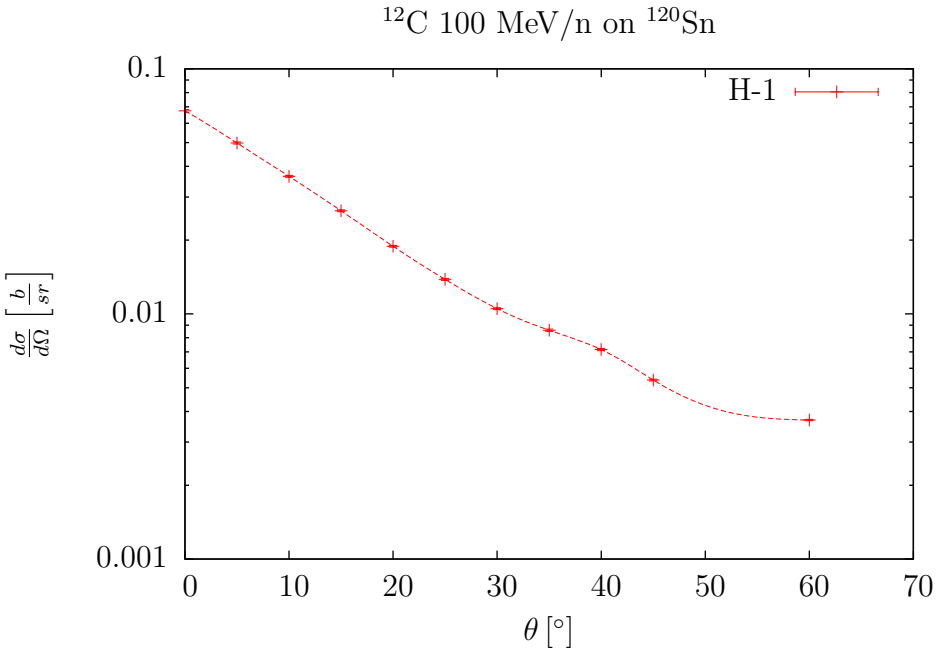


(b) logarithmic scale

Figure A.2: ^{3}He differential cross sections



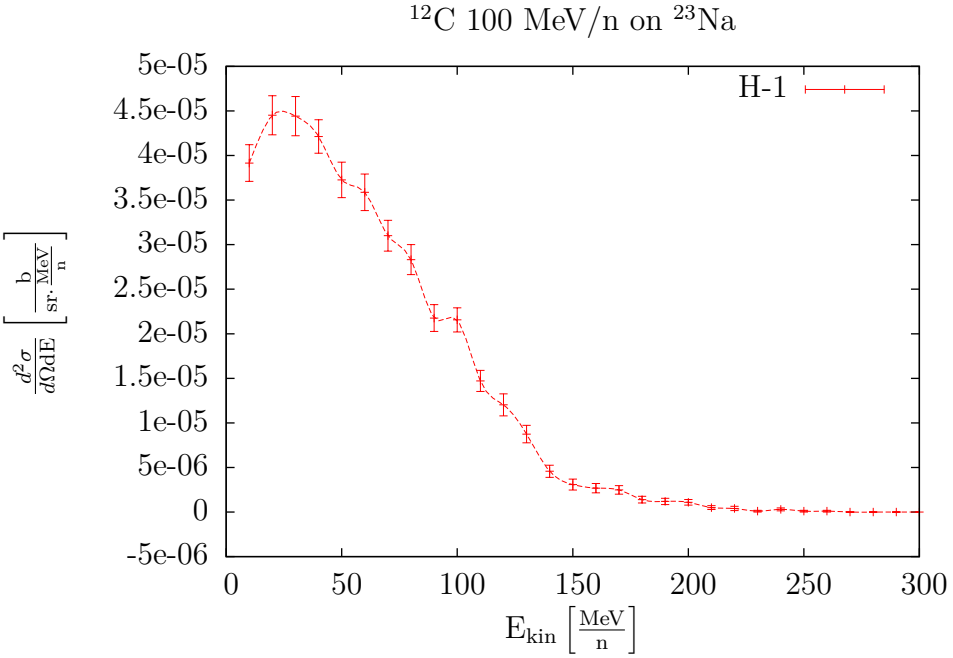
(a) linear scale



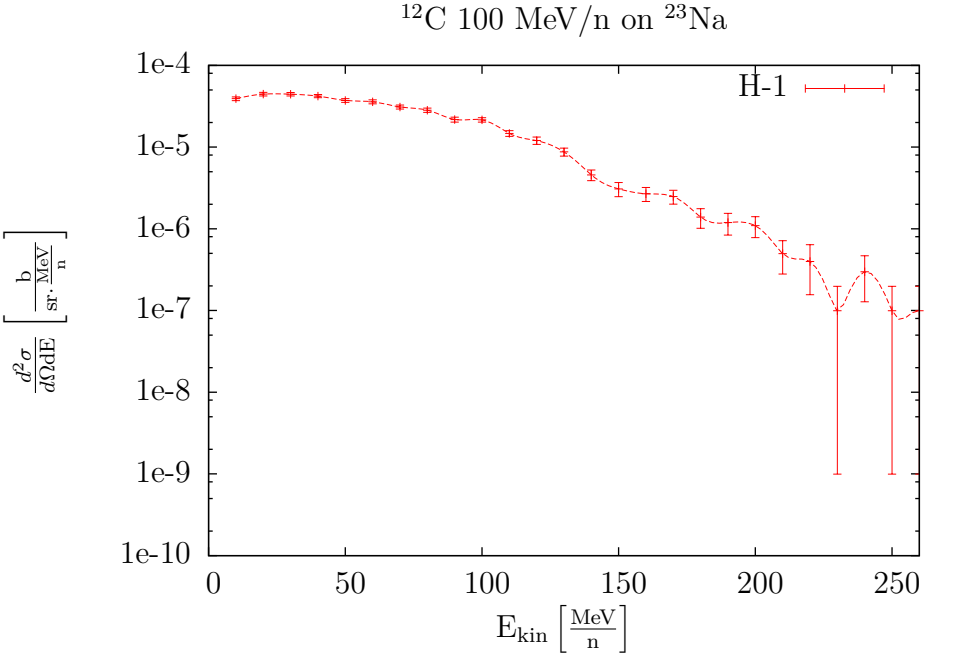
(b) logarithmic scale

Figure A.3: ^{1}H differential cross sections

A.2. Double differential cross sections

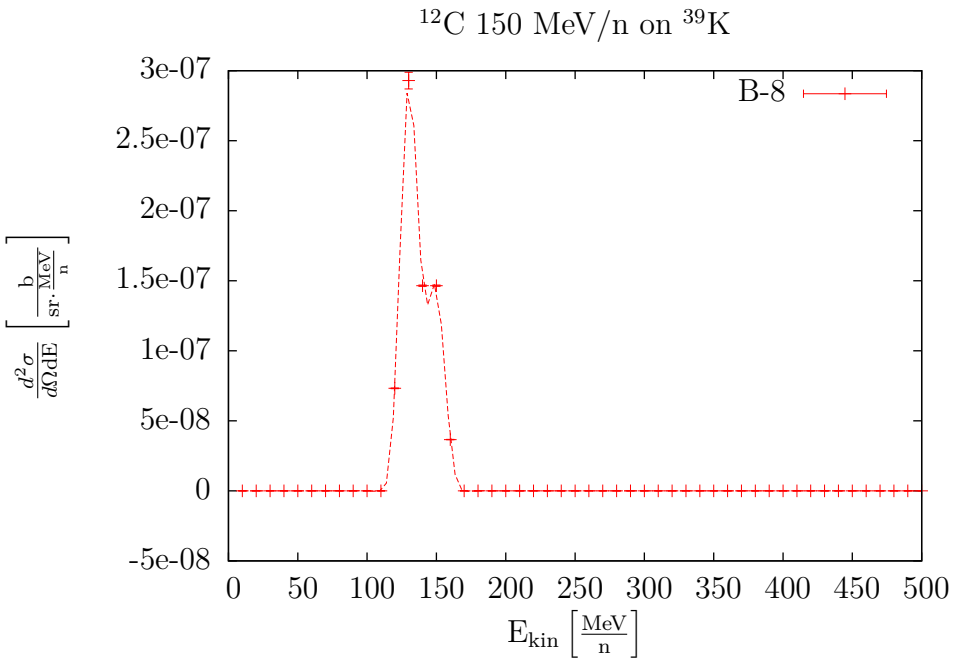


(a) linear scale

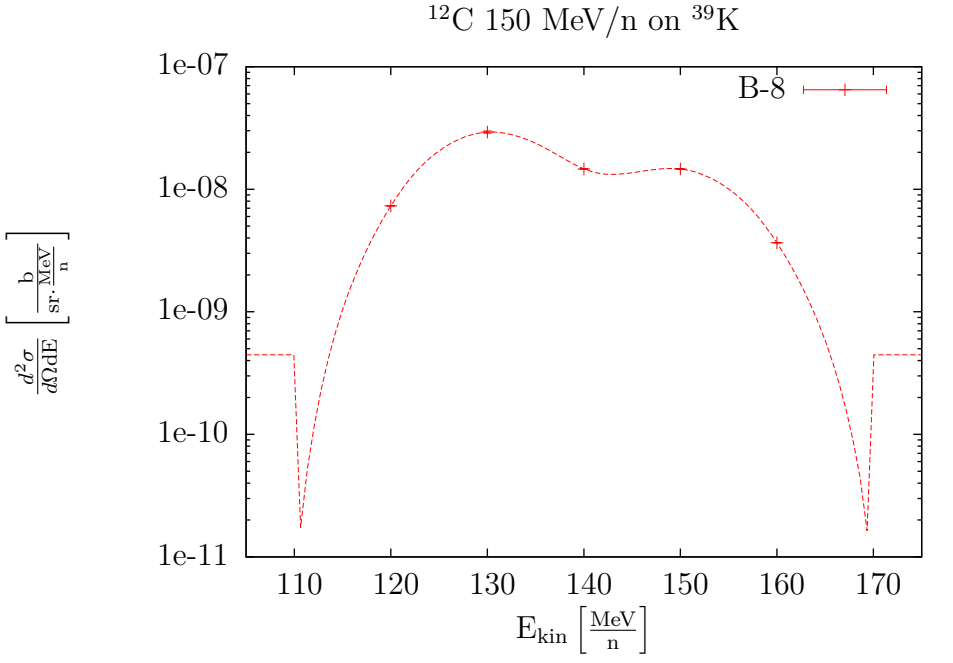


(b) logarithmic scale

Figure A.4: ^{1}H double differential cross sections at $\theta = 27.5°$.

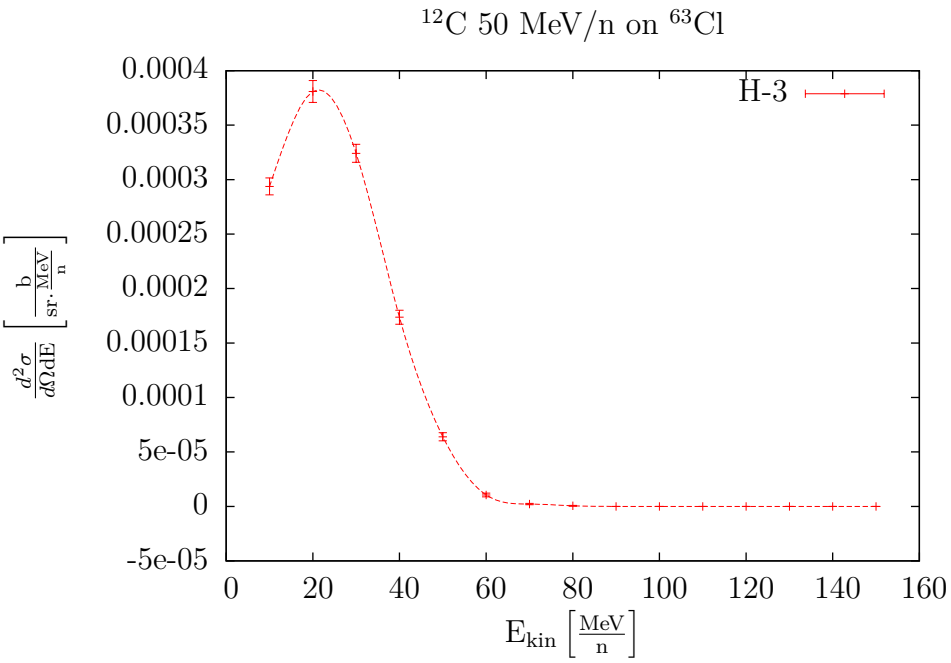


(a) linear scale

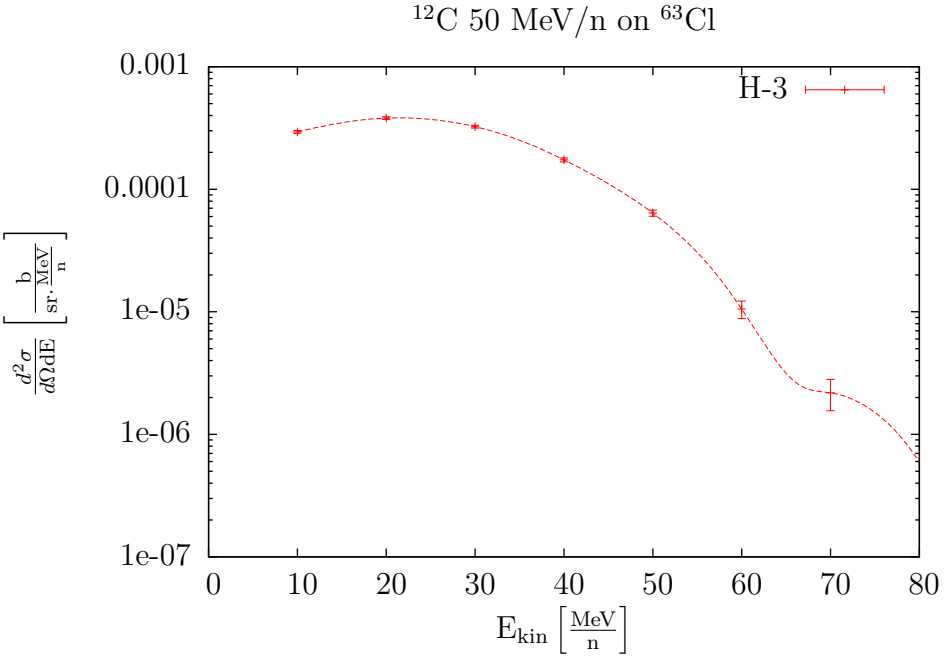


(b) logarithmic scale

Figure A.5: ^{8}B double differential cross sections at $\theta = 2.5°$.



(a) linear scale



(b) logarithmic scale

Figure A.6: ^{3}H double differential cross sections at $\theta = 17.5°$.