eman ta zabal zazu

Universidad del País Vasco Euskal Herriko Unibertsitatea

ZTF-FCT
Zientzia eta Teknologia Fakultatea
Facultad de Ciencia y Tecnología

Visualization of Classical and Quantum Scattering

Undergraduate Degree Project
Bachelor's Degree in Physics

Aitor Pérez Godoy

Directed by
Josu M. Igartua Aldamiz

Leioa, July, 2018

Contents

Introduction

The nature of this project is totally pedagogic. As I was finishing the career I thought about contributing my grain of sand in physics by helping those students who are starting now. Most of the concepts learned in physics are not very intuitive and in most of the cases some examples are needed to understand them. From my experience, visualizing them was the most difficult part. So, in order to avoid these problems I find it useful teaching them in a interactive way, in a way in which the students could participate in their learning process.

In order to achieve that goal it is going to be ilustrated how the classical and quantum scattering can be explained in a interactive way. Due to the social vocation of this project it is useful to program it in a open source language, as it is Python. Every code used in this project will be available on github. In this way, anybody who is interested in this topic will be able to enjoy it or even use it for other projects.

Firstly, it will be given a short introduction of the libraries used in the project. Then, following a chrnological order, and also intuitive, the first setp will be to explain the classical scattering and its computational implementation. Finally, the quantum scattering which is far more complex than the latter. Besides, some mathematical calcules and procedures will be added in the apendix so that there is no need of reading other books.

Chapter 1

Libraries and modules

One of the reasons of choosing Python as the programming language is that there are a lot of libraries with useful modules ready to use them. Below, it is ilustrated a summary of the ones used to program all the algorithms, graphics and videos in the project.

- **Numpy**: In some problems the necessity of storing the information in arrays and getting the information from them, or just changing the values of them, it is very common. In this cases, Python does not work correctly, but using numpy arrays this problem disappears. In almost all the programms there is an array to store the results and another to store the values in which we wanted to calculate the results.

- **Scipy**: This library has a lot of integration modules that are very useful for computational calculation. Also, there is an extension of this same library called scipy.specials that includes mathematical well known functions, such as, bessel functions, legendre functions, and a long et cetera.

- **Matplotlib**: In order to visualize the results in graphics, an easy way to do it is by using matplotlib which gives freedom to plot graphics.

- **Ipywidgets**: To make the graphics and videos interactive, ipywidgets works efficiently to include widgets on them. This has been the most important library used in the project as is the one which makes the difference between been interactive or not. The structure used in this cases is the one ilustrated in code (2.1).

- **Vpython**: All the mentioned libraries until this one are very common in daily computing with python. Perphaps, vpython is not as well known as the others, but it is equally useful. This is used to make displays or animations in 3D. The way it works is by drawing different kind of forms with very simple comands. An easy example would be to draw an sphere and a box with few code lines,

```
import vpython as vp
vp.sphere(pos=vp.vector(1,2,1),radius=0.5,
          color=vp.color.cyan)
vp.box(pos=vp.vector(0,0,0),lenght=1,width=2,
   height=2)
```

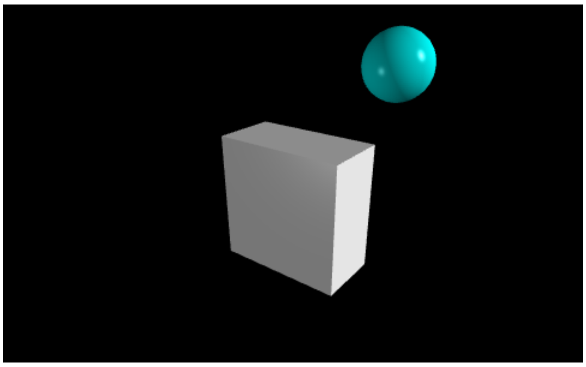


Figure 1.1: Vpython's 3D animation of a box and a sphere.

Chapter 2

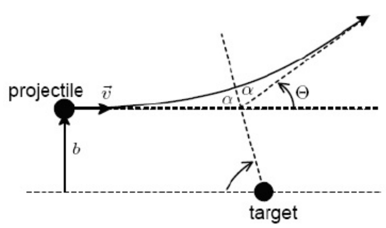
Classical scattering

This chapter reviews the classical theory describing the scattering of a beamed particle hitting a target particle. In order to make it interactive, for each explanation or graphic will be some changeable paramaters.

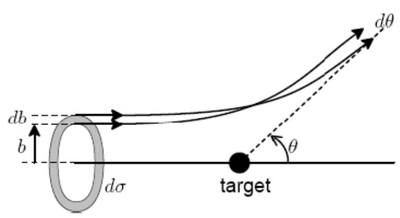
2.1 Deflection function and scattering angle

To explain what the deflection function is , the easiest way is to ilustrate it. In figure 2.1a it can be seen how a poyectile coming from the infinite approaches a target. Taking the incident direction to be horizontal, the initial vertical distance b from the target is called the impact parameter. Once the particle has passed through the interaction area, it is scattered to a deflection angle Θ at infinity.

One could think that for each value of b corresponds an unique Θ. The truth is that in some cases different values of b end up with the same deflection angle. Usually, the value of b cannot be determined experimentally, so a relationship between b and Θ is needed, Θ(b), namely the deflection function. Thus, the expression for the deflection function (See A.1) is



(a)



(b)

Figure 2.1: Classical scattering structure of a particle coming from infinite and approaching a target [1].

$$\Theta(b) = \pi - 2\int_{r_{min}}^{\infty} \frac{b}{r^2\sqrt{1-\frac{V(r)}{E}-\frac{b^2}{r^2}}}\mathrm{d}r. \tag{2.1}$$

Generally, this integral can only be calculated numerically. Besides, in opposite to θ, the deflection angle cannot be directly observable. Nonetheless, they are related as

$$\theta = \Theta \text{ (if } \Theta > 0) \;\;\&\;\; \theta = |2n\pi + \Theta| \text{ for } n = Int\left|\frac{\pi - \Theta}{2\pi}\right| \text{ (if } \Theta < 0). \tag{2.2}$$

After introducing the theory, it may be interesting to see an example of how could be implemented in a interactive way. As it has already said, the deflection function needs to be calculated numerically. However, in the case of the Rutherford scattering, the solution is analitic. In fact, the result is the one described for the Kepler's orbits, from where appears the following relation between b and Θ [1]:

$$b = \frac{k}{2E}\cot(\frac{\Theta}{2}). \tag{2.3}$$

At the same time, other useful expressions for the eccentricity and r_{min} [1] are the following ones:

$$e = \sqrt{1+\frac{4E^2b^2}{k^2}} \quad \text{and} \quad r_{min} = \frac{2Eb^2}{k(\pm e - 1)}. \tag{2.4}$$

There are many ways to demonstrate that the implementation works correctly. One of them is comparing the analitic and numerical solutions of the Rutherford scattering. Moreover, some changeable parameters are introduced so that the user could play and interact with them. Using ipywidgets this task becomes very simple. An easy way to do it is by following this structure:

Listing 2.1: The structure of the implementation used for an interactive graphic of the deflection function.

```
def deflectionfunction(choose, arguments):

    # Implement the numerical and analitic results

    if choose=="Numerical":
        plt.plot(impact_parameter,deflection)
    elif choose=="Analitic":
        plt.plot(impact_parameter,deflection_theory
            ,color="C1")
```

```
    else:
        plt.plot(impact_parameter,deflection,"b+")
        plt.plot(impact_parameter,deflection_theory
            ,color="C1")
    plt.show()

choose=widgets.ToggleButtons(options=["Numerical","
    Analitic","both"])
b1=widgets.FloatSlider(min=0,max=2,step=0.01,value
    =0.01,description='impact parameter')
b2=widgets.FloatSlider(min=0,max=2,step=0.01,value
    =1,description='impact parameter')
q=widgets.interactive(deflectionfunction,choose=
    choose,b1=b1,b2=b2)
display(q)
```

It may be easier to save the results in numpy arrays. In this way, the values can be taken or changed for others. Furthermore, in order to calculate the numerical solutions, the scipy library is very efficient due to his integration modules. Then, matplotlib.pyplot, works really well plotting those results. The implementation code for solving and plotting Eq. (2.1) can be found in github [6]. Anyway, the result is illustrated in figure 2.2.

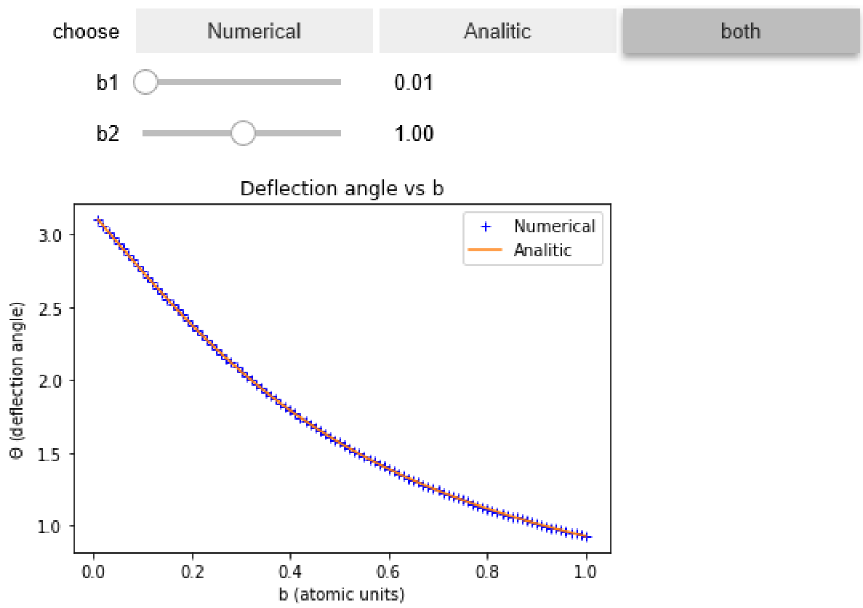


Figure 2.2: The analitical and numerical solutions of the deflection function for Coulomb's potential in terms of b, the impact parameter.

Here it can be seen that the numerical and analitic solutions are equal. In this way, for any given potential it is assured that the calcules for each deflection function will be totally accurate. Besides, it is ilustrated how as b increasing the deflection function decreases, which is something completely intuitive. However, if the distance between the projectile and the target increases the interaction will be of shorter time.

After this brief introduction of the deflection function, the differences between two famous potentials are explained below. Those are, the previously mentioned Rutherford's potential, and the Thomson's potential, also known as the plum-pudding potential. Analizing these ones it is also useful to see the differences between Thomson's and Rutherford's atomic models which are caracterized by these potentials. In 1909 Ernest Rutherford directed the Geiger-Marsden experiment [7] which suggested that the Thomson's plum pudding model of the atom [8] was incorrect. This experiment was based on hitting a thin gold foil with beamed α particles. The plum-pudding potential is represented by

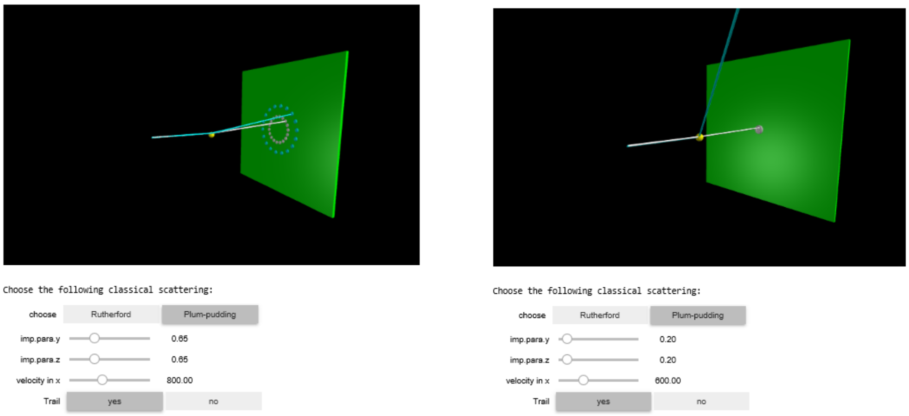
$$V(r) = \begin{cases} \frac{Z}{2R}(\frac{r^2}{R^2} - 3), & \text{if } \; r \leq R \;, \\ -\frac{Z}{r}, & \text{if } \; r > R \;. \end{cases} \tag{2.5}$$

In order to visualize these two interactions vpython works really efficiently as it has been already explained in code (1.1). Plotting them is similar to matplotlib and the structure used in code (2.1) could be copied. The plan is sending beamed particles against a target and see where they finish. To make this interactively the user will choose some starting conditions, such as, the potential, the impact parameters and the velocity, again using ipywidgets. On the other hand, the position of these particles is given by solving the following differential equation, better known as the Newton's second law of motion,

$$\vec{F} = m\vec{a}. \tag{2.6}$$

Knowing the starting position, velocity and acceleration values is enough to solve the Eq. (2.6) with Eulers method [9] or, as in our case, with Runge-kutta method [1]. As you may have noticed, due to dealing with central fields, to calculate the next steps values it is easier to work in spherical coordinates [2]. The code used for this purpose it can be found it in github [6].

In figure 2.3b, for Rutherford's potential at small distances it can be seen that the particles are defrected almost backward. Whereas, in plum-pudding potential the deflection decreases at small distances. This was the evidence that Rutherford saw in his experiment to understand that the plum-pudding model was totally incorrect. In essence, at distances near to radius of the



(a) (b)

Figure 2.3: The classical scattering of an electron coming from infinite and approaching a gold atom. Blue: Rutherford's atomic model ; White: Thomson's atomic model.

nucleus, the beamed particles interact heavily with it. Thus, the nucleus, as Rutherford said [7], is composed of neutrons and protons concentrated in the center of the atom.

Another result that can be seen in this animation is the rainbow scattering which is caracteristic in potentials of two areas as the plumm-pudding potential. With incresing b, firstly, the scattering angle also does. Nonetheless, as b further increases, the time the particle spends on the interaction area decreases, and therefore, the scattering angle θ. The maximum scattering angle reached by the beamed particles is called the rainbow angle.

Not only learning but, with these simulations there are given the tools for programming any orbital problem. In fact, Coulomb's potential and the gravitational potential are both inversely proportional to r^2, but with different constants. So, just changing some values the solar system could be easily programmed. As it has already said in the introduction, it is bound up in working on Python with a pedagogic objective to sharing the codes for other purposes. Anybody who is interested in this project or in programming theirs, will find it useful these programming codes [6].

2.2 Cross section

The reaction rates measured in scattering experiments, and the energy spectra and angular distributions of the reactions between the beam particles and the targets gives information about the dynamics of them. Imagine placing particle detectors where the beam particles are no more interacting

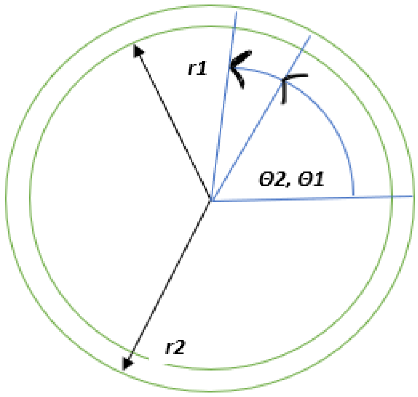


Figure 2.4: Projection of figure 2.1b where particles are beamed from positions ∈ (r1,r2) and angles ∈ $(\theta 1, \theta 2)$.

with the target, measuring the number of particles (dN) captured by the detector subtending a solid angle, dΩ. This number is proportional to the beam flux I. It is common to introduce the following expression,

$$\sigma(\Omega) = \frac{\text{number of particles scattered into } \theta \text{ and } \theta + d\theta}{\text{beamed particles per unit area } \cdot \text{ solid angle}} = \frac{1}{I}\frac{dN}{d\Omega} \tag{2.7}$$

as the cross section [4], which is the probability of a particle being scattered to the angle θ. Another expression related with this one is the differential cross section, $\frac{d\sigma}{d\Omega}$. As previouly done in the first section, in order to assure that the implementation works correctly, the analitic and numerical results are going to be compared.

Firstly, it will be calculated the last position of all the beamed particles that will have starting random positions $\in (r1, r2)$ and angles $\in (\theta_1, \theta_2)$, as it can be seen in figure 2.4. Then, the last step is to number how many of them are in the resulting solid angle, which is in terms of b, the impact parameter.

In order to make it interactive, this time, the user will choose the target's atomic number, apart from the already mentioned limits of the random starting positions, their energy and number of beamed particles. The figure 2.5 shows how for $r1 = r2$ the last position of the beamed particles coincides with the theoretical deflection function (the blue circle), which has been been calculated using Eq. (2.1) for r1, the impact parameter. Then, if the implementation works correctly, any calculation for other parameters should as well. The result given when $r1 \neq r2$ is ilustrated in figure 2.6. This time the particles are scattered to different positions as it was expected.

Once the cross section and the methods for calculating it with experimental results has been introduced, it is mentionable that the Rutherford's scat-

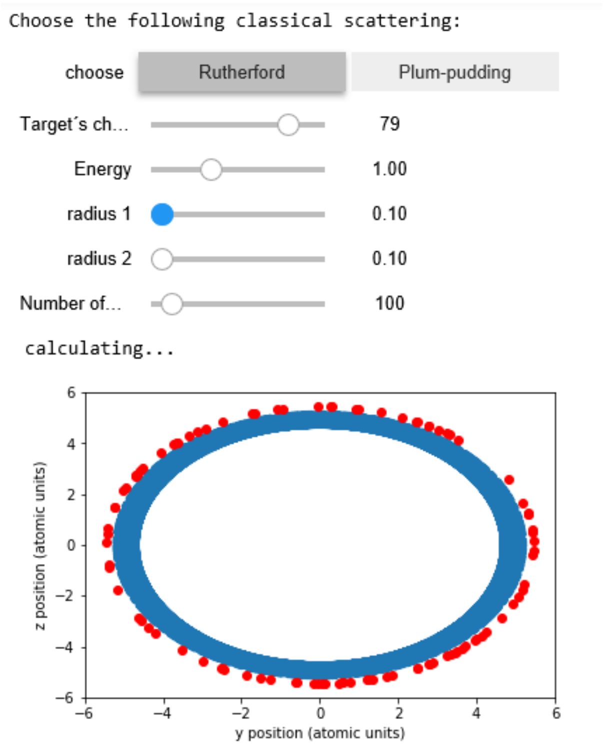


Figure 2.5: The last position of the beamed particles coming from the same starting position ($r1 = r2$), where the blue circle represents the analitic result of the particles coming from a starting position r1.

tering has already an analitic solution and that there is no neccesity of calculating it with computational methods. The Rutherfords cross section (See A.2) follows this equation:

$$\sigma(\theta) = \frac{k^2}{16E^2} \frac{1}{\sin^4 \frac{\theta}{2}}. \tag{2.8}$$

Anyway, having an adequate understanding on how it is calculated experimentally is totally neccesary. Nevertheless, it is true that in all the simulations programmed the projectiles where just electrons due to convenience in atomic units. In the Rutherfords experiment the particles used where α particles which mass is 7294 times bigger than the electron's and their charge 2 times bigger (but positive).

In addition, just taking into account the Coulomb's interaction between the projectile and the nucleus is enough to understand the whole concept. But it is true that for experimental results it should also be considered the interaction between the projectile and the electrons around the nucleus.

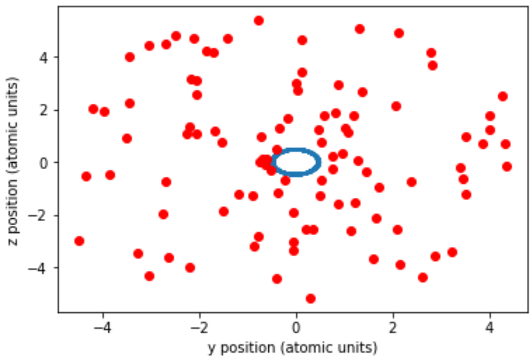


Figure 2.6: Last position of the beamed particles coming from different starting positions ($r1 \neq r2$). Also, the blue circle appears representing the analitic result of the particles coming from a starting position r1.

Chapter 3

Quantum scattering

Due to quantum phenomena, the quantum scattering at the atomic scale is more appropiate to describe the scattering of a particle beamed with velocity near the speed of light. Between the classical and quantum scattering theories there is a main important difference, dealing with matter waves that obey the Schrödinger equation. Some procedures may be explained, but in general, the calcules will be ilustrated in the Apendix B. Anyway, the theory explained below are just some examples programmed with Python to see how can be implemented different quantum scattering problems using diverse libraries in a interactive way.

3.1 Scattering amplitude

To understand the bases, the quantum scattering from a central field is ilustrated in the following lines. The latter, can be understood in figure 3.1.

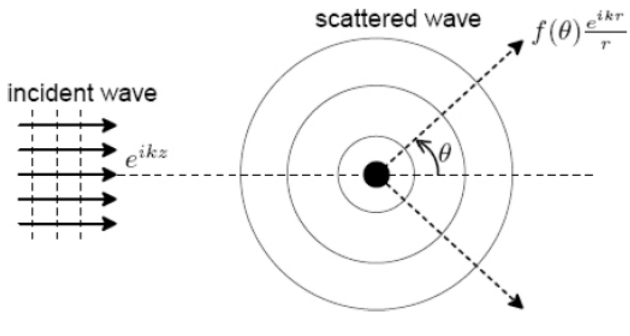


Figure 3.1: The Quantum scattering from a central field [1].

Thus, for $r \to \infty$ the whole wave function can be written as a superposition of the incident and the scattered wave,

$$\psi = \psi_{in} + \psi_{sc} \quad .$$

For scattering with $E > 0$ and $V = 0$ the most usual solution are plane waves [4]. So,

$$\psi \to e^{ikz} + f(\theta)\frac{e^{ikr}}{r} \quad ,$$

where $k = \sqrt{2mE}/\hbar$ is the wave vector, $E > 0$ the energy, and m the mass of the particle. Furthermore, the factor $f(\theta)$ is called the scattering amplitude. In this calcules the wave number k is the same for both parts due to considering the scattering to be elastic. Anyway, the scattering amplitude is the one describing the cross section (See B.1). In fact, it is represented by the following expression:

$$\sigma(\Omega) = |f(\theta)|^2 \quad . \tag{3.1}$$

Making a comparison with classical scattering, scattering amplitude is to quantum scattering as the deflection function is to classical scattering.

3.2 Partial waves

Decomposing the incident waves into partial waves is an useful and effective method to calculate the scattering amplitude and, hence, the quantum cross section. For this purpose it may be useful to start by transforming the problem into spherical coordinates. Thus, the wave function [1] can be written as

$$\psi(r, \theta, \gamma) = \sum_l R_l(r) Y_{lm}(\theta, \gamma). \tag{3.2}$$

To simplify this function the first step is to separate its variables. On one hand, the angular solution is given by $P_l(\cos\theta)$, the Legendre polynomials, where it is noticed that the resulting wave function is independet of γ because of freedom to choose $\vec{k}$ parallel to the z axis. So, the incident plane wave e^{ikz} can be written as

$$e^{ikr\cos(\theta)} = \sum_{l=0}^{\infty} (2l+1) i^l j_l(kr) P_l(\cos\theta), \tag{3.3}$$

where $j_l(kr)$ is the spherical Bessel function of the first kind. Later there is an example of how would be calculated the phase shifts, and for convinience, in order to see the different applications of Python's modules, the case for a nonzero potential is ilustrated below, where the solutions for the wave function are:

$$\psi = \sum_{l=0}^{\infty} (2l+1) i^l R_l(k,r) P_l(\cos\theta). \tag{3.4}$$

The $R_l(kr)$ satisfies the radial Schrödinger equation, and for $V = 0$, $j_l(kr)$. The latter has an asymptotic behaviour and that is the reason why the following expression for the scattering amplitude fixes well, hence,

$$f(\theta) = \frac{1}{2ik} \sum_{l=0}^{\infty} (2l+1) i^l P_l(\cos\theta)(e^{2i\delta_l} - 1) e^{-il\pi/2} \frac{e^{ikr}}{r}, \tag{3.5}$$

where using $e^{-il\pi/2} = (-i^l)$,

$$f(\theta) = \frac{1}{k} \sum_{l=0}^{\infty} (2l+1) e^{i\delta_l} \sin\delta_l P_l(\cos\theta). \tag{3.6}$$

Finally, going back to Eq. (3.1), the cross section is

$$\sigma(\theta) = \frac{1}{k^2} \mid \sum_{l=0}^{\infty} (2l+1) e^{i\delta_l} \sin\delta_l P_l(\cos\theta) \mid^2 . \tag{3.7}$$

Integrating this equation, the total cross section results to have the following expression:

$$\sigma_t = \frac{4\pi}{k^2} \sum_{l=0}^{\infty} (2l+1) \sin^2\delta_l \ . \tag{3.8}$$

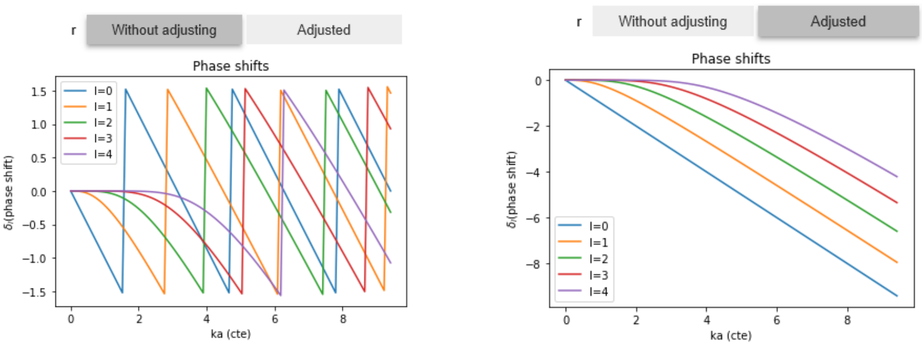
3.3 Phase shifts calculation

Now, the last step would be to calculate the phase shifts. An interactive example is introduced below to understand this problem, specifically the phase sifts in a scattering from a 1D potential. This is the potential used for this purpose:

$$V(r) = \begin{cases} \infty, & \text{if } \ r \leq a \ , \\ 0, & \text{if } \ r > a \ . \end{cases} \tag{3.9}$$

On one hand, the wave function vanishes inside the sphere. On the other hand, outside $r > a$, the radial wave function is a solution for this case. Imposing continuity conditions at $r = a$ the following equation can be calculated:

$$\tan\delta_l = \frac{j_l(ka)}{n_l(ka)}. \tag{3.10}$$



(a) (b)

Figure 3.2: The phase shifts calculation from Eq. (3.10) in terms of the energy (ka). (a): without adjusting; (b): adjusted.

The figure 3.2a illustrates the behaviour of Eq. (3.10). For convinience, it is useful to adjust this plot because in Eq. (3.8) $\sin^2(\delta)$ does not difference between positive or negative phase sifts. Thus, in figure 3.2b it can be seen clearer at small energies that the leader phase shift would be the one given by $l = 0$. Hence, the lower angular momentum. Nevertheless, as the energy increases, higher momentum angular waves start to contribute to the total cross section.

One of the implicit objectives of this project is to understand the differences between classical and quantum scattering. The latter can be seen by plotting the differential cross section for a hard sphere. Indeed, the result for the quantum scattering has already been discused in Eq. (3.7). For convinience, so that the calcules become easier, there are only taking into account the contributions of some angular momentums. For example, based on figure 3.2b, for low energies, just taking the contribution of the lower angular momentum would be a good approximation. Thus, the differential cross section for $ka = 0.01$ would be represented by

$$\sigma(\theta) = \frac{\sin^2 \delta_0}{k^2}. \tag{3.11}$$

However, for $ka = 1.0$ it is neccesary to include minimum one more angular momentum, the one given by the p-wave. So, the differential cross section becomes into

$$\sigma(\theta) = \frac{1}{k^2}[\sin^2 \delta_0 + 6\cos(\delta_0 - \delta_1)\sin\delta_0 \sin\delta_1 \cos\theta + 9\sin^2\delta_1 \cos^2\theta], \tag{3.12}$$

where, $P_0(cos\theta) = 1$ and $P_1(cos\theta) = cos\theta$ have been used. Also, the phase shifts have been calculated from Eq. (3.10). On the other hand, the classical result is given by

$$\sigma(\theta) = \frac{a^2}{4}. \tag{3.13}$$

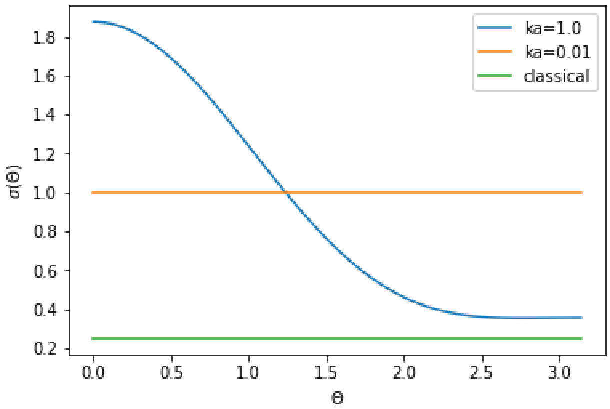


Figure 3.3: The differential cross section calculated for a hard sphere potential. On one hand, the quantum model for two different energies. And then, for the classical model which does not have energy dependence.

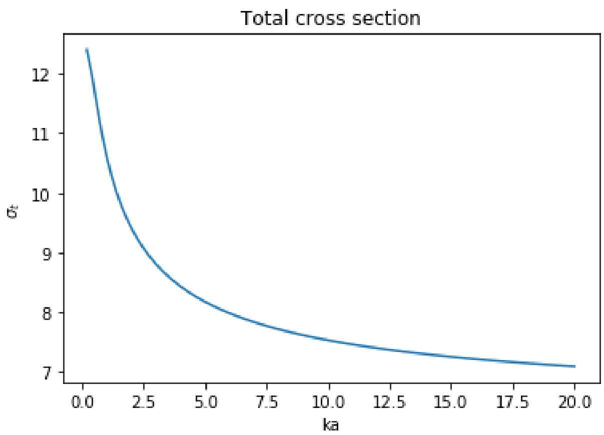


Figure 3.4: The total quantum cross section for a hard sphere potential in terms of different energies.

Furthermore, these expresions are all ilustrated in figure 3.3. In the case of the lower energy, the differential cross section is isotropic as it has been calculated only with the s-wave. However, the contributions of other angular momentums would be extremely small and it ends up being a good approximation. This does not happen with higher energies, with $ka = 1.0$ for example. In this case, there is a maximum at forward angles. Moreover, something curious happens comparing it with the classical result, which is constant no matter what energy does it have. At backward angles they coincide almost with the same value due to the decreasement of de Broglie wavelenght with the increasement of the energy [5].

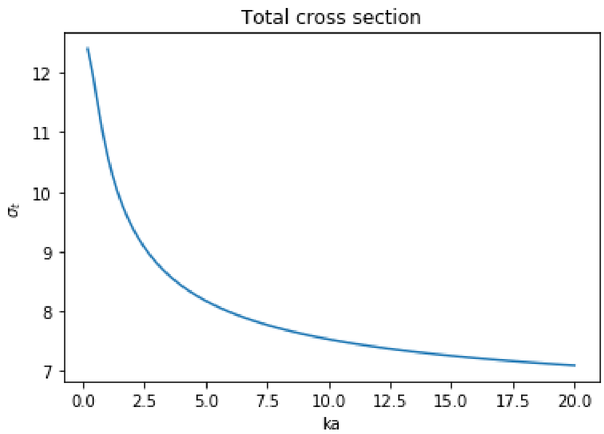


Figure 3.5: The total quantum cross section for a hard sphere potential in terms of different energies.

In figure 3.5 it can be seen numerically calculated the Eq (3.8). As it was expected, a maximum appears at $ka = 0$, which corresponds to the contribution of the s-wave. Other energies have other angular momentum's contributions and they do not have isotropic behaviours. So, the bigger the energy the smaller total cross section we get.

Chapter 4

Conclusions

In this project we have seen how they can be implemented different physical problems in a interactive way. We could divide the conclusions in four parts:

- Together with my comrade Ander Arbide, we have updated both projects on github. We feel the responsability of sharing with other people due to programming on an open source language, in addition to the pedagogic behaviour that has followed the whole project. In this way, anyone who is interested in learning about it or just in using the codes for other purposes (s)he is totally invited to do it.

- Programming in Python has issued the job because of its libraries, specially in programming 3D animations, where vpython is very intuitive. Also, thanks to the giant community supporting Python, finding answers to almost all the problems or doubts on programming has been really easy.

- Being interactive or not influences a lot on students. Given them the tools for studying in class or even at home facilitates the teacher's role. In fact, after any explanation in class there is always a visualization of the concept teached. So, in this way, having just a display to show the animations or graphics we would save a lot of time trying to ilustrate them.

- The chapters done in this project could be interesting to show them in the first course in physics or in a science week in order to beautify physics to those who are not very related with it. In our case specifically, could be appropiate to use them in General Physics or in Modern Physics as an introduction for the classical and quantum models. Pherhaps, in Classical Mechanics teaching central forces could also be possible.

To sum up, we let the door open to other students who want to contribute by improving this projects or begining anothers. Nowadays, trying to get better information on internet helps to fan the flames of curiosity in young students who are thinking about studying physics.

Appendix A

Classical scattering

A.1 Derivation of the deflection function

The deflection angle can be written as

$$\Theta = \pi - 2\alpha. \tag{A.1}$$

Although, it is easier to work with the polar angle α in terms of the radial vector. Firstly,

$$\dot{\theta} = \frac{L}{mr^2} \quad \text{and} \quad \dot{r} = \sqrt{\frac{2}{m}(E - V(r) - \frac{L^2}{2mr^2})}\ , \tag{A.2}$$

where θ is the angle from the target's horizontal line to the particle's outgoing direction (figure 2.1b) and V(r) is the interaction potential. Moreover, using the next relation,

$$\partial\theta = \frac{\partial\theta}{\partial r}\partial r = \frac{\partial\theta/\partial t}{\partial r/\partial t}\partial r = \frac{\dot{\theta}}{\dot{r}}\partial r \tag{A.3}$$

time would be eliminated from Eq. (A.2). Then, using Eq. (A.2) into Eq. (A.3) and integrating both sides, it is obtained the following:

$$\theta(r) - \theta_0 = \int_{r_0}^{r} \frac{L/mr^2}{\sqrt{\frac{2}{m}(E - V(r) - \frac{L^2}{2mr^2})}}\mathrm{d}r. \tag{A.4}$$

In this case , α would be described by the radial vector from r_{min} to ∞. Thus,

$$\alpha = \int_{r_0}^{r} \frac{L/mr^2}{\sqrt{\frac{2}{m}(E - V(r) - \frac{L^2}{2mr^2})}}\mathrm{d}r. \tag{A.5}$$

Also, in order to simplify Eq. (A.5) in terms of the impact parameter b, we use the definition of the angular momentum and the kinetic energy

$$v = \sqrt{2E/m}\ ,\ L = mvb\ . \tag{A.6}$$

Thus, the expression for the deflection function is

$$\Theta(b) = \pi - 2\int_{r_{min}}^{\infty} \frac{b}{r^2\sqrt{1-\frac{V(r)}{E}-\frac{b^2}{r^2}}}\mathrm{d}r. \tag{A.7}$$

A.2 Rutherford cross section

In figure 2.1b we can easily see that the area of the particles entering with an impact parameter $\in (b, b+db)$ is represented by the following expression:

$$d\sigma = 2\pi b\ db. \tag{A.8}$$

At the same time, because scattering in central fields is independent of the azimutal angle, we can conclude that the solid angle is $d\Omega = 2\pi sin\theta d\theta$. Besides, the differential cross section is the division of these last two expressions. Hence,

$$\sigma(\theta) = \frac{1}{\sin\theta}\left|\frac{db}{d\theta}\right|. \tag{A.9}$$

Also, as we have mentioned in chapter 1 they may be more than one impact parameter contributing to the same deflection angle. To take all of them into account we end up with the following equation:

$$\sigma(\theta) = \frac{1}{\sin\theta}\sum_i b_i \left|\frac{db}{d\Theta}\right|_i. \tag{A.10}$$

Now, explicitly for the rutherford scattering we already know the expression for $b(\Theta)$, the one seen in Eq. (2.3). Deriving it we have

$$\frac{db}{d\Theta} = -\frac{k}{4E\sin^2(\Theta/2)}.$$

Then, using it in Eq. (A.10) we get

$$\sigma(\theta) = \frac{k^2}{8E^2}\frac{\cot(\theta/2)}{\sin^2(\theta/2)\sin(\theta)}.$$

Finally, using trigonometric relations we get the Rutherford cross section:

$$\sigma(\theta) = \frac{k^2}{16E^2}\frac{1}{\sin^4\frac{\theta}{2}}. \tag{A.11}$$

Appendix B

Quantum scattering

B.1 The scattering amplitude

The probability of finding a particle inside a close region with S surface is defined as

$$P = \int_V \Psi^* \Psi dr. \tag{B.1}$$

And so, the probability flux it is how the probability changes in time. Thus,

$$\frac{dP}{dt} = \frac{d}{dt} \int_V \Psi^* \Psi dr. \tag{B.2}$$

Besides, for a free-particle system following the Schrodinger equation [7] we have that

$$\frac{-\hbar^2}{2m} \nabla^2 \Psi = i\hbar \frac{d\Psi^*}{dt}. \tag{B.3}$$

Using Eq. (B.3) in Eq. (B.2) we obtain the following equation,

$$\frac{dP}{dt} = \frac{-i\hbar}{2m} \int_V (\Psi \nabla^2 \Psi^* - \Psi^* \nabla^2 \Psi) dr. \tag{B.4}$$

As we want to have a result given for a S surface we must use Green theorem [3] to obtain it. Thus,

$$\frac{dP}{dt} = \frac{-i\hbar}{2m} \int_S (\Psi \nabla \Psi^* - \Psi^* \nabla \Psi) dS. \tag{B.5}$$

Consequently,

$$S = \frac{-i\hbar}{2m} (\Psi \nabla \Psi^* - \Psi^* \nabla \Psi). \tag{B.6}$$

Now we can calculate the incident and outgoing flux considering z the direction of the wave vector. Using the incident and scattered waves in Eq. (B.6) we get the following result:

$$I_{in} = \frac{\hbar k}{m} \quad and \quad I_{out} = \frac{\hbar k}{m}\frac{|f(\theta)|^2}{r^2}. \tag{B.7}$$

Furthermore, supposing that we have a detector at r and angle θ the number of particles finishing there are

$$dN = I_{out}dS = I_{out}r^2 d\Omega = I_{in}\,|f(\theta)|^2\,d\Omega,$$

which remembering Eq. (2.7) the quantum cross section depends only on the scattering amplitude, hence,

$$\sigma = |f(\theta)|^2\,. \tag{B.8}$$

Bibliography

[1] Jay Wang, Computational Modeling and Visualization of Physical Systems with Python (2015).

[2] Juan M. Aguirregabiria, Mekanika klasikoa (2015).

[3] Tom Rother, Greens Functions in Classical Physics (2017).

[4] B Povh. Particles and nuclei. Springer, 2008.

[5] M. Spivak, *Calculus*, 3ª ed., Reverté, Barcelona, 2012.

[6] Link to the project: Visualization of Classical and Quantum Scattering.

[7] E. Rutherford, Nuclear Constitution of Atoms (1920).

[8] Hentschel, Klaus. (2009). Atomic Models, J.J. Thomson's Plum Pudding Model. 10.1007/978-3-540-70626-7_9.

[9] Griffiths D.F., Higham D.J. (2010) Eulers Method. In: Numerical Methods for Ordinary Differential Equations. Springer Undergraduate Mathematics Series. Springer, London.