UNIVERSITÉ GRENOBLE ALPES

RAPPORT DE PROJET NUMERIQUE

Electron relativiste

Théo Dubroca
Léo Reynaud

Professeur
Ismaïl Mourad

September 8, 2023

Contents

A Introduction

Depuis la mise au point de la Mécanique Quantique par l'école de Copenhague au début du XXe siècle, les chercheurs et chercheuses du monde entier s'acharnent à élaborer des modèles de plus en plus complexes pour décrire le comportement du monde de l'infiniment petit. Un des premiers grand pas vers la réussite de ce défi a été réalisé par Schrödinger qui mit en place l'équation d'onde d'une particule quelconque pour décrire l'évolution spatio-temporelle de cette ou ces particules :

$$i\hbar\frac{\partial\Psi(t,\vec{r})}{\partial t} = -\frac{\hbar^2}{2m}\nabla^2\Psi(t,\vec{r}) + V(t,\vec{r})\Psi(t,\vec{r}) \tag{1}$$

Bien qu'à la base de la Mécanique Quantique moderne cette équation ne tient pas compte d'une autre physique dont l'action est négligeable sur certains aspects du monde quantique mais néanmoins primordiale pour comprendre pleinement notre Univers : la physique de la Relativité Restreinte. Cette théorie élaborée par Albert Einstein en 1905 permet de comprendre l'évolution spatio-temporelle d'objets se déplaçant à des vitesses proches de celle de la lumière en décrivant l'espace et le temps non pas comme deux entités séparées mais comme une seule et même toile sur laquelle les objets rapides (très rapides) se meuvent et aux effets physiques particuliers bien décrit par la Relativité restreinte.

La théorie d'Einstein fut donc travaillée par d'autres pour obtenir une connexion entre physique quantique et Relativité Restreinte. Les résultats résultats partiellement satisfaisant furent l'équation de Klein-Gordon, qui fut reprise par Dirac pour donner naissance à l'équation qui porte aujourd'hui sont nom. La première s'énonce pour des particules dont le spin est nulle tandis que la seconde décrit le comportement d'une particule avec un spin $\pm\frac{1}{2}$, l'électron. Elle furent énoncées à partir de l'équation de Schrödinger en prenant cette fois-ci une énergie relativiste dans un espace-temps plat (dit de Minkovsky) obéissant à la relation :

$$E^2 = p^2m^2 + m^2c^4 \tag{2}$$

où p est la quantité de mouvement de la particule, m sa masse et c la vitesse de la lumière.

A partir de là en prenant les équivalents quantiques de l'énergie E et de l'impulsion p soit

$$E = i\hbar\frac{\partial\Psi(t,\vec{r})}{\partial t} \quad \text{et} \quad p^2 = \hbar^2\nabla^2 \tag{3}$$

l'on peut réécrire l'équation de Schrödinger telle que :

$$\nabla^2(\Psi(t,\vec{r})) - \frac{1}{c}\frac{\partial^2}{\partial t^2}(\Psi(t,\vec{r})) = \frac{m^2c^2}{\hbar^2}(\Psi(t,\vec{r})) \tag{4}$$

Cette équation n'est autre que l'équation de Klein-Gordon qui décrit en mécanique quantique relativiste l'évolution spatio-temporelle d'une particule sans spin avec une énergie relativiste. A partir de cette équation, Dirac proposa de la réécrire de manière à ne pas faire intervenir de dérivées secondes et donc l'équation de Klein-Gordon peut se mettre sous une forme factorisée en introduisant deux nouveaux opérateurs : $\hat{\alpha}$ et $\hat{\beta}$. Dirac obtint alors l'équation suivante :

$$\left(\beta mc^2 + \sum_{n=1}^{3}\alpha_n p_n\right)\Psi(t,\vec{x},\vec{y},\vec{z}) = i\hbar\frac{\partial\Psi(t,\vec{x},\vec{y},\vec{z})}{\partial t} \tag{5}$$

où n indice les coordonnées x,y,z. Pour tenir compte du spin Dirac utilisa les matrices de Pauli afin d'exprimer les α_n et β. Grâce à cela ces denières s'écrivent :

$$\alpha_n = \begin{pmatrix} 0 & \sigma_n \\ \sigma_n & 0 \end{pmatrix} \quad \text{et} \quad \beta = \begin{pmatrix} I_2 & 0 \\ 0 & -I_2 \end{pmatrix} \tag{6}$$

avec I_2 la matrice unité 2x2 et σ_n les matrices de Pauli.

Notre but dans ce projet est d'écrire un code en Python permettant de trouver l'évolution d'une fonction d'onde par l'équation de Dirac. Pour cela nous utiliserons une méthode visant à discrétiser les dérivées partielles. La méthode employée est celle de la méthode des différences finies avec un schéma implicite. Après plusieurs tentative et différent schéma utilisé nous nous somme rendus compte de la difficulté trop grande d'effectuer un tel code c'est pourquoi nous nous somme rabattus sur l'équation de Klein-Gordon toujours avec un schéma implicite et la méthode des différences finies. L'utilité du schéma implicite réside dans sa stabilité quelque soit la méthode de discrétisation employée.

B Difficultés rencontrées pour l'équation de Dirac

Nous avions donc pour but au départ de résoudre des équations simplifier de Dirac. Pour cela nous avons résolut l'équation en 2 dimension (une d'espace et une de temps) et sans potentiel ce qui nous donna un resultat de la forme suivante:

$$(c\sigma_x p_x + mc^2\beta)\psi = E\psi \tag{7}$$

avec

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \beta = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Donc

$$\left[c \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} p_x + mc^2 \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \right] \begin{pmatrix} \psi_A \\ \psi_B \end{pmatrix} = E \begin{pmatrix} \psi_A \\ \psi_B \end{pmatrix}$$

$$\left[c \begin{pmatrix} 0 & \psi_B \\ \psi_A & 0 \end{pmatrix} p_x + mc^2 \begin{pmatrix} \psi_A & 0 \\ 0 & -\psi_B \end{pmatrix} \right] = E \begin{pmatrix} \psi_A \\ \psi_B \end{pmatrix}$$

$$\left[\begin{pmatrix} mc^2\psi_A & cp_x\psi_B \\ cp_x\psi_A & -mc^2\psi_B \end{pmatrix} = E \begin{pmatrix} \psi_A \\ \psi_B \end{pmatrix} \right.$$

Ce qui finalement nous donne les deux équations suivantes :

$$f(x) = \left\{ \begin{array}{lcl} cp_x\psi_B + mc^2\psi_A & = & E\psi_A \\ cp_x\psi_A - mc^2\psi_B & = & E\psi_B \end{array} \right. \tag{8}$$

On se retrouve donc avec un couple d'équations et donc deux composantes pour ψ. Le but étant d'exprimer ψ_B en fonction de ψ_A puis de résoudre l'équation différentielle d'ordre deux ainsi obtenue.

$$\psi_B = \frac{cp_x}{mc^2 + E}\psi_A \tag{9}$$

$$\Rightarrow \frac{c^2 p_x^2}{mc^2 + E}\psi_A + (mc^2 - E)\psi_A = 0 \tag{10}$$

Avec $p_x = -i\hbar\frac{\partial}{\partial x}$

Nous pensions pouvoir réussir à résoudre cette équation différentielle mais après de nombreuses tentatives nous avons préféré abandonner car impossible d'avoir des résultats un minimum satisfaisant. Nous pensions que les problèmes viennent des conditions initiales, nous avons donc essayé avec plusieurs type d'ondes mais sans plus de résultats.

Nous avons donc pris la décision d'essayer de résoudre les équations de Klein-Gordon qui malgré tout sont un peu plus simple.

C Equation de Klein-Gordon

Dans notre étude nous nous concentrons uniquement sur la résolution de l'équation de Klein-Gordon à une dimension spatiale. Afin de résoudre l'équation nous avons tout d'abord cherché à comprendre comment les équations aux dérivées partielles pouvaient être numériquement résolues. Pour cela nous avons regardé des méthodes de résolution de ce qui se rapprochait le plus d'une équation telle que celle de Klein-Gordon : il y a par exemple la résolution numérique de l'équation de Schrödinger, de l'équation de la chaleur [**2**] faisant intervenir des dérivées secondes ou encore de l'équation des ondes. Nous nous sommes alors rendus compte que la méthode la plus généralement employées était celle des différences finies ce qui nous a permis de cibler un peu mieux nos recherches tant sur le point de vue numérique que mathématique.
En effet, la méthode mathématique employée pour les différences finies n'est autre que l'approximation des dérivées premières et secondes par un développement de Taylor puis par une discrétisation s'appuyant sur un maillage de l'espace et du temps. Comme le montre [**1**] nous pouvons approximer les dérivées secondes par les équations discrétisées suivantes :

Soit le maillage du temps et de l'espace :

$$T = tdt \quad x = pdx \tag{11}$$

Et enfin pour la discrétisation des dérivées secondes :

$$\begin{cases} \frac{\partial^2 \Psi(x,t)}{\partial t^2} = \frac{\Psi^p_{t+1} - 2\Psi^p_t + \Psi^p_{t-1}}{dt^2} \\ \frac{\partial^2 \Psi(x,t)}{\partial x^2} = \frac{\Psi^{p+1}_t - 2\Psi^p_t + \Psi^{p-1}_t}{dx^2} \end{cases} \tag{12}$$

Ce qui nous permet d'écrire l'équation de Klein-Gordon discrétisée telle que :

$$\Psi^p_{t+1} = -c^2 \frac{dt^2}{dx^2}(\Psi^{p+1}_t - 2\Psi^p_t + \Psi^{p-1}_t) - \frac{m^2 c^4 dt^2}{\hbar^2}\Psi^p_t + \Psi^p_{t-1} - 2\Psi^p_t \tag{13}$$

Comme nous avons exprimé la valeur à gauche de la fonction d'onde soit Ψ^p_{t+1} nous avons écris un schéma numérique dit implicite. Une fois le schéma obtenu, il nous faut l'implémenter en Python ce qui requiert certaines définitions dont les conditions aux limites et initiales.
De plus l'équation de Klein-Gordon est une équations aux dérivées partielles de type hyperbolique (cf [**7**]) nous devons utiliser le théorème de Cauchy-Kowalevskaïa [**6**][**8**] et pour le respecter nous pouvons utiliser un paquet d'onde gaussien définit dans [**4**] comme conditions initiales :

$$\Psi(t_0, x) = e^{ik_0 x - \omega t} e^{\frac{(x-x_0)^2}{4\sigma_0^2}} \tag{14}$$

La valeur x_0 pour laquelle on trouvera le centre de la gaussienne sera supérieure à 0 afin de bien observer l'entièreté de la fonction d'onde après une application à l'équation de Klein-Gordon. Le code obtenu est donné en Annexe I. Après avoir exécuter le code nous obtenons les graphiques suivants :

On observe sur ces graphes que la fonction d'onde ne semble pas connaître de variation dans le temps hormis à la fin de l'itération en temps ce qui s'avère faux lorsque l'on affiche les matrices associée à chaque fonction tracée sur les graphes. Le problème d'affichage vient que le nombre d'itération en temps, de pas, est trop faible pour permettre de voir les variations tellement celles-ci sont faibles. En essayant de mettre des pas de temps plus élevé nous obtenons que les valeurs calculées dépassent les capacités de Python et donc celui-ci affiche des valeurs "NaN" ce qui ne nous rend aucun graphe.

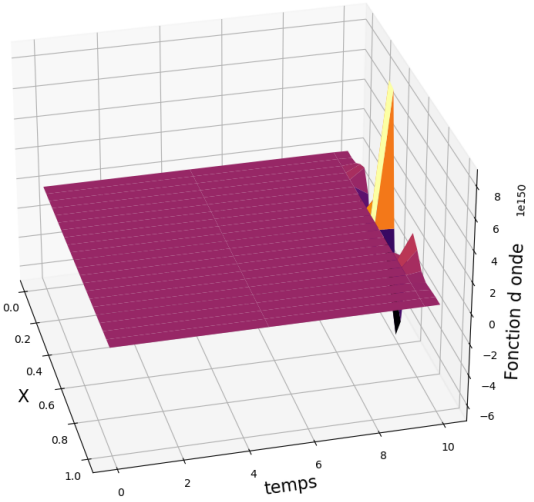

Figure 1: Partie réelle de ψ en fonction du temps

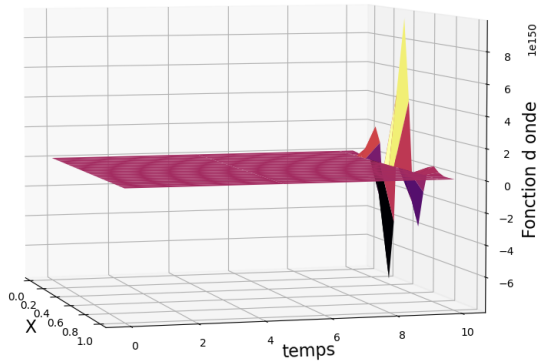

Figure 2: Partie imaginaire de ψ en fonction du temps

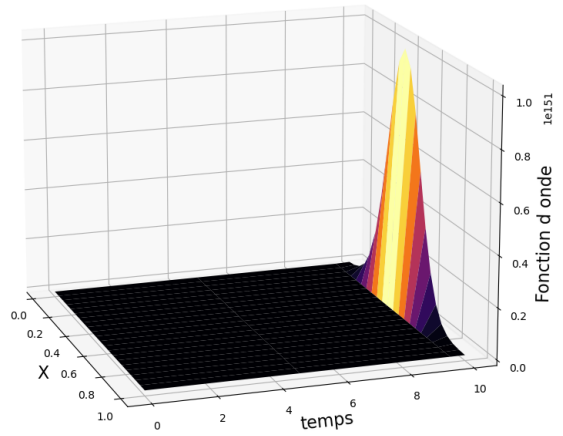

Figure 3: Module de ψ en fonction du temps

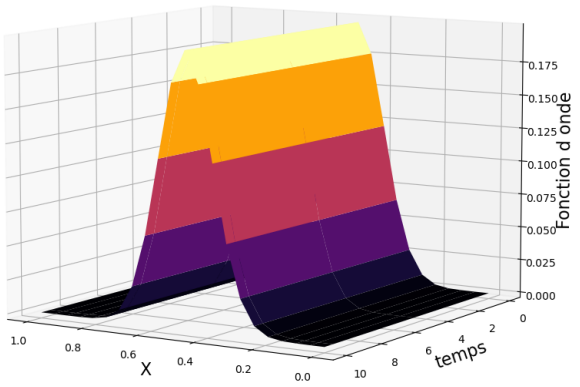

Figure 4: Probabilité de présence de ψ

D Conclusion

Nous avons eu beaucoup de problème lors de l'exécution de ce code. En effet l'impossibilité de normer notre partie imaginaire et réelle entre chaque pas de temps a fait que ces valeurs ont grimpé en flèche et ont vite été trop grandes pour Python. Pour contrer ces problèmes différents types de discrétisations et de schémas auraient pu être utilisés. En effet notre méthode ne conserve pas la norme. Une des méthodes que nous aurions pu employée est la méthode spectrale qui consiste en une décomposition de la fonction d'onde dans l'espace de Fourier et de travailler à intégrer l'équation de Klein-Gordon dans cet espace pour ensuite repasser dans l'espace réel. C'est une méthode semble-t-il plus efficace et demandant moins d'effort de calcul. Nous aurions pu aussi utiliser un schéma de Cranck-Nicholson mélangeant les schémas explicites et implicites de la méthode aux différences finies.

Dominique Lefebvre (2016) "La méthode aux différences finies"
http://www.tangentex.com/DiffFinies.htm

Dominique Lefebvre (2016) "L'équation de la chaleur"
http://www.tangentex.com/EquationChaleur.htm#Par4

Marc Buffat (2008) "Approximation des dérivées par différences finies"
https://perso.univ-lyon1.fr/marc.buffat/COURS/COURSDF_HTML/node16.html#SECTION005240000000000000000

F. Legrand (2015) "Paquet d'onde d'une particule matérielle"
https://www.f-legrand.fr/scidoc/docimg/sciphys/quantique/paquet/paquet.html

Raphaèle Herbin (2011) "Analyse numérique des équations aux dérivées partielles". Engineering school. Marseille cel-00637008
https://cel.archives-ouvertes.fr/cel-00637008/document

Université de Lyon. "Equations aux dérivées partielles et leurs approximations".
http://math.univ-lyon1.fr/homes-www/lagoutiere/poly.pdf

REZZOUG Imad (2017/2018). University Of Oum El Bouaghi. "Méthodes numériques pour les équations différentielles partielles"
https://www.academia.edu/34966901/M%C3%A9thodes_num%C3%A9riques_pour_les_%C3%A9quations_diff%C3%A9rentielles_partielles_pdf

ikipedia. "Théorème de Cauchy-Kowalevsky"
https://fr.wikipedia.org/wiki/Th%C3%A9or%C3%A8me_de_Cauchy-Kowalevski

E Annexe

```
import matplotlib.pyplot as plt
import numpy as np
from scipy.integrate import odeint
from mpl_toolkits.mplot3d import Axes3D
import mpl_toolkits.mplot3d
from math import *
import cmath
#Equation de Schr dinger avec correction relativiste par la m thode des diff rences finie

#D finition des constantes
c=1 #3.0*10**8 #vitesse de la lumi re en m.s-1
c2=c**2
c4=c2**2
x0=0#centre de la gaussienne
sigma0=0.5
i=complex(0,1)
m=9.1*10**-31 #la masse de l' lectron  en kg
m2=m**2
w=30

hbar = 1.1*10**-34 #constante de Planck r duite en J.s
hbar2=hbar**2
```

```
#D finition du domaine spatio-temporel et son maillage
Lx=1 #la distance en m
tf=10 #le temps total en s

Nx=200 #le nombre de pas effectu  dans l'espace
Nt=22 #le nombre de pas effectu  dans le temps

dx=Lx/(Nx-1) #la longueur d'une maille en espace
dt=tf/(Nt-1) #la longueur d'une maille en temps
dt2=dt**2
dx2=dx**2
x0=10
k0=1 #float(input('Saisir une valeur pour le vecteur donde k0'))

if k0 < Lx/2*sigma0**2:
    raise Exception("Etalement du paquet d'onde donc refaire tourner le programme en prenan


#Conditions initiales r pondants au th. de Cauchy-Kowalesky, paquet d'onde gaussien
def psi (t,x):
    psi = cmath.exp(i*k0*x-w*t)*exp(-(x-x0)**2/4*sigma0**2)
    return psi

psi1=np.zeros((Nt,Nx)) #Tableau nous donnant la partie r el de psi
psi2=np.zeros((Nt,Nx)) #Tableau nous donnant la partie imaginaire de psi
psi3=np.zeros((Nt,Nx)) #Tableau nous donnant le module de psi
psi4=np.zeros((Nt,Nx)) #Tableau nous donnant la probabilit  de pr sence, module de psi au

#conditions aux limites
a=0
for x in range (0,Nx):
    psi0=psi(0,x)
    # dpsi0=dpsi(0,x)
    psi1[0][x]=psi0.real
    psi2[0][x]=psi0.imag
    psi3[0][x]=abs(psi0)
    a=a+(psi3[0][x])**2



for x in range (0,Nx):
    psi4[0][x]=(psi3[0][x])**2/a


    #Euler implicit
b=0
for t in range (0,Nt-1):
    PSI0=psi0
    PSI=psi1
    PSI2=psi2
    PSI
    for p in range (0,Nx-1):
        PSI[t+1][p]=-c2*(dt2/dx2)*(PSI[t][p+1]-2*PSI[t][p]+PSI[t][p-1])-(m2*c4*dt2/hbar2)*P
        psi1[t+1][p]=PSI[t+1][p]

        PSI2[t+1][p]=-c2*(dt2/dx2)*(PSI2[t][p+1]-2*PSI2[t][p]+PSI2[t][p-1])-(m2*c4*dt2/hbar
        psi2[t+1][p]=PSI2[t+1][p]


        psi3[t+1][p]=sqrt((PSI[t+1][p])**2+psi2[t+1][p]**2)
```

```
            b=b+(psi3[t+1][p])**2
        for p in range (0,Nx-1):
            psi4[t+1][p]=((psi3[t+1][p])**2/b)

print(psi4)

# Figure1
x_array=np.linspace(0.0,1,Nx)
t_array=np.linspace(0.0,10,Nt)
X,T=np.meshgrid(x_array,t_array)

#Ploting the real part of psi
fig=plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.set_xlabel('X', fontsize = 16)
ax.set_ylabel('temps', fontsize = 16)
ax.set_zlabel('Fonction d onde', fontsize = 16)
ax.view_init(elev=15, azim = 120)

p = ax.plot_surface(X,T,psi1,cstride=1,linewidth=0,cmap='inferno')

# Figure2
x_array=np.linspace(0.0,1,Nx)
t_array=np.linspace(0.0,10,Nt)
X,T=np.meshgrid(x_array,t_array)


fig=plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.set_xlabel('X', fontsize = 16)
ax.set_ylabel('temps', fontsize = 16)
ax.set_zlabel('Fonction d onde', fontsize = 16)
ax.view_init(elev=15, azim = 120)

p = ax.plot_surface(X,T,psi2,cstride=1,linewidth=0,cmap='inferno')


# Figure3
x_array=np.linspace(0.0,1,Nx)
t_array=np.linspace(0.0,10,Nt)
X,T=np.meshgrid(x_array,t_array)


fig=plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.set_xlabel('X', fontsize = 16)
ax.set_ylabel('temps', fontsize = 16)
ax.set_zlabel('Fonction d onde', fontsize = 16)
ax.view_init(elev=15, azim = 120)

p = ax.plot_surface(X,T,psi3,cstride=1,linewidth=0,cmap='inferno')
# Figure4
x_array=np.linspace(0.0,1,Nx)
t_array=np.linspace(0.0,10,Nt)
X,T=np.meshgrid(x_array,t_array)


fig=plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.set_xlabel('X', fontsize = 16)
ax.set_ylabel('temps', fontsize = 16)
ax.set_zlabel('Fonction d onde', fontsize = 16)
```

```
ax.view_init(elev=15, azim = 120)

p = ax.plot_surface(X,T,psi4,cstride=1,linewidth=0,cmap='inferno')
```