Antoine Bellaigue
Jérémy Archer

Paquet d'onde gaussien dans un potentiel

1. Présentation

Le programme a pour but de produire une animation paramétrée présentant les comportements spatial et temporel d'un paquet d'onde gaussien en interaction avec différents potentiels dans le vide.

L'objectif ici est donc de résoudre l'équation de Schrodinger dépendante du temps :

$$i\hbar \frac{d\Psi(\vec{r},t)}{dt} = \frac{-\hbar^2}{2\mathrm{m}} \nabla^2 \Psi(\vec{r},t) + V(\vec{r},t)\,\Psi(\vec{r},t) \quad (1)$$

Pour cela, nous avons discrétisé le problème, la résolution se fait ensuite de manière itérative : la fonction $\Psi(\vec{r},t)$ nous permet de calculer la fonction $\Psi(\vec{r},t+dt)$.

$$\mathrm{i}\frac{\Psi_j^{n+1} - \Psi_j^{n}}{\Delta t} = \frac{-\Psi_{j+1}^{n} - 2\Psi_j^{n} + \Psi_{j-1}^{n}}{(\Delta x)^2} + V_j \Psi_j^{n} \quad (2)$$

$$\mathrm{i}\frac{\Psi_j^{n+1} - \Psi_j^{n}}{\Delta t} = H\,\Psi_j^{n+1} \quad (3)$$

$$(1 + i\Delta t H)\,\Psi_j^{n+1} = \Psi_j^{n} \quad (4)$$

$$\Psi(x, t+\Delta t) = e^{-i\Delta t H}\,\Psi(x,t) \quad (5)$$

En utilisant l'approximation de Caley on a :

$$e^{-i\Delta t H} = \frac{1 - \frac{i\Delta t H}{2}}{1 + \frac{i\Delta t H}{2}} \quad (6)$$

Le schéma numérique s'écrit alors :

$$operateur\_1\,\Psi_j^{n+1} = operateur\_2\,\Psi_j^{n} \quad (7)$$

où $operateur\_1 = 1 + \frac{i\Delta t H}{2}$ et $operateur\_2 = 1 - \frac{i\Delta t H}{2}$ (8)

Antoine Bellaigue
Jérémy Archer

Le problème revient à résoudre une équation du type :

A X=Y (9) où A est une matrice tridiagonale

La résolution se fait ensuite à l'aide de l'algorithme de Thomas.

On manipule ici des matrices tridiagonales, le problème peut donc être réécrit :

$$a_j \Psi_{j-1}^{n+1} + b_j \Psi_j^{n+1} + c_j \Psi_{j+1}^{n+1} = d_j \Psi_{j-1}^{n} + e_j \Psi_j^{n} + f_j \Psi_{j+1}^{n} \quad (10)$$

Où a_j, b_j, c_j, d_j, e_j et f_j sont les éléments des 3 diagonales centrales des matrices tridiagonales.

Nous avons appliqué en condition initiale un paquet d'onde gaussien:

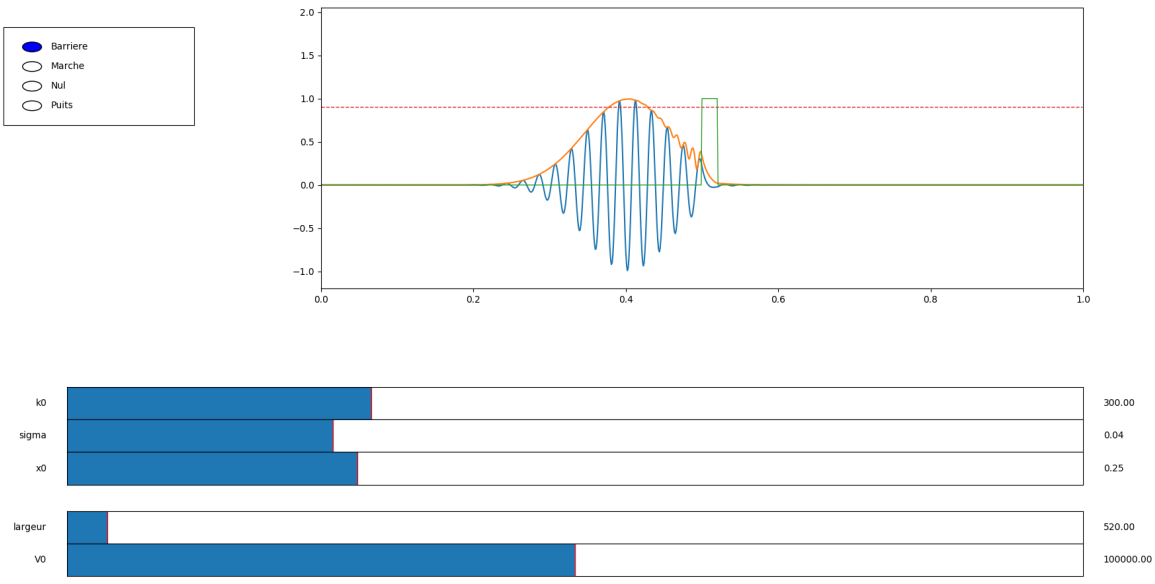
$$\Psi(x,0) = e^{jk_0 x} e^{\frac{-(x-x_0)^2}{4\sigma_0^2}} \quad (11)$$

2. Mode d'emploi

Lorsque le programme est lancé, une interface s'ouvre. Par défaut, le paquet d'onde gaussien se propage en direction d'une barrière de potentiel. Il est possible de modifier les caractéristiques du paquet d'onde et du potentiel en utilisant les slider en dessous du graphique. On peut ainsi modifier k_0 le vecteur d'onde, σ_0 l'écart type, x_0 la position initiale, la largeur du potentiel, V_0 la hauteur de la barrière et de la marche.

A l'aide de boutons radio sous la gauche du graphique, on peut choisir le type de potentiel que l'on souhaite appliquer parmi les suivants : puits de potentiel harmonique, barrière de potentiel, marche de potentiel, puits de potentiel. Afin d'actualiser le nouveau potentiel, il est nécessaire de modifier un des 5 paramètres sous le graphique.

La courbe en trait plein orange correspond au module de la fonction d'onde tandis que la courbe en trait plein bleu représente la partie réelle de la fonction d'onde. La droite horizontale en pointillé orange correspond à l'énergie cinétique du paquet d'onde.



Interface interactive du programme

3. Fonctionnement du programme

Nous avons utilisé les librairies suivantes pour créer notre programme :

- numpy : pour manipuler des matrices, des nombres complexes et des fonctions mathématiques (exp,sin,..).
- matplotlib.pyplot : permet de réaliser des tracés.
- Matplotlib.animation : permet de réaliser des animations.
- matplotlib.widgets : création de curseurs et de boutons radio.

Dans un premier temps, on définit et on fixe les variables de base pour modéliser notre système : les pas de temps et d'espace, les valeurs initiales de la fonction d'onde ainsi que du potentiel.

Ensuite, nous allons construire la matrice de l'hamiltonien, qui est la somme de l'opérateur cinétique T et du potentiel V. Dans la suite, seul V sera modifié, T ne changera pas. Nous mettons en place différentes fonctions pour modéliser les différents potentiels souhaités (ainsi que leurs tracés sur le graphique final). L'opérateur H initial est donc construit.

Puis, on créé une fonction permettant de générer le paquet d'ondes gaussien définit à partir de sa position initial x_0 , le vecteur d'onde k_0 , ainsi qu'un écart-type σ_0 . On en profite pour définir des fonctions agissant sur ce paquet d'ondes, pour calculer sa norme et sa partie réelle.

Il faut maintenant créer les opérateurs qui entreront en jeu dans la résolution de l'algorithme. On les note *operateur_1* et *operateur_2* (voir Présentation). Comme ce sont deux matrices tridiagonales, on extrait leurs diagonales dans des variables, ce qui diminuera les temps de calculs lorsque l'on agira dessus. On créé également une fonction de résolution AX = Y et une fonction effectuant un produit matriciel A*X où A est une matrice tridiagonale et X un vecteur, qui interviendront dans l'algorithme.

Enfin, il s'agit de résoudre l'équation de Schrödinger et de tracer le paquet d'ondes (et le potentiel) dans le temps. On définit dans un premier temps les curseurs et les boutons radio qui permettront à l'utilisateur de pouvoir modifier les variables du système. On créé une fonction **update** qui va mettre à jour les variables de la fonction d'onde et du potentiel après modification par l'utilisateur. L'hamiltonien et le paquet d'ondes au temps zéro sont ainsi mis à jour, les opérateurs intervenant dans l'algorithme de résolution sont donc recalculés. De la même manière, on créé une fonction **potfunc** lié aux boutons radio qui va modifier la forme du potentiel V dans l'hamiltonien. Dans un second temps, on définit la fonction **animate** qui est le siège de l'algorithme de résolution de notre système. On y calcule de manière récursive notre fonction d'onde à chaque pas de temps puis on la trace (module et partie réelle). On conclut avec la ligne de commande *anim.FuncAnimation* qui permet le tracé à chaque pas de temps du paquet d'ondes et donc son animation.