Universidad cooperativa de Colombia

TRABAJO DE AULA

Sistemas operativos

Avance de proyecto 1

Alumnos:

Antonio Ortega

Charlies rueda

Docente:

Diego Andrés restrepo leal

Santa marta, Magdalena

2 de noviembres 2023

Informe sobre el Código: Representación Gráfica de Polinomios de Hermite y Funciones de Onda

Introducción:

El código proporcionado tiene como objetivo principal calcular y visualizar los polinomios de Hermite y las funciones de onda asociadas para diferentes valores de n. Los polinomios de Hermite son soluciones de la ecuación diferencial de Hermite y tienen aplicaciones significativas en la mecánica cuántica, especialmente en el contexto de la oscilación cuántica y los osciladores armónicos cuánticos. Este informe analizará el código paso a paso y explicará las operaciones realizadas en cada sección.

Desarrollo:

1. **Importación de Librerías:** El código comienza importando las bibliotecas necesarias:
    - **sympy**: Para realizar cálculos simbólicos, incluyendo la manipulación de polinomios de Hermite y funciones de onda.
    - **matplotlib.pyplot**: Para crear visualizaciones gráficas.
    - **numpy**: Para manipulaciones numéricas y generación de puntos para las gráficas.
2. **Definición de Símbolos y Polinomios de Hermite:** Se definen los símbolos �x y �n, y se crea una expresión para los polinomios de Hermite usando la función **sp.hermite(n, x)**.
3. **Cálculo y Representación Gráfica de �0(�)$H0(x)$:** Se calcula el polinomio de Hermite para �=0n=0 (hermite_0) y se representa gráficamente utilizando la función **sp.plot()**. El resultado se muestra en la primera figura con el título "H0(x)".
4. **Transformación a Función de Onda y Representación Gráfica:** El polinomio de Hermite �0(�)$H0(x)$ se transforma en una función de onda �0(�)$\psi0(x)$ utilizando la fórmula asociada. Se crea una representación numérica de esta función de onda y se grafica en una segunda figura titulada "Función de onda para �=0n=0".
5. **Cálculo y Representación Gráfica de �1(�)$\psi1(x)$, �2(�)$\psi2(x)$ y �3(�)$\psi3(x)$:** Se calculan los polinomios de Hermite para �=1n=1, �=2n=2 y �=3n=3 respectivamente (hermite_1, hermite_2, hermite_3). Estos polinomios se transforman en funciones de onda �1(�)$\psi1(x)$, �2(�)$\psi2(x)$ y �3(�)$\psi3(x)$ utilizando la misma fórmula que se utilizó para �0(�)$\psi0(x)$. Estas funciones de onda se convierten en representaciones numéricas y se grafican en la tercera figura con el título "Funciones de onda para �=1,2,3n=1,2,3".

Solución Del Oscilador Armónico Cuántico Por Medio De Método de numerov

```
#include <stdio.h>

#include <math.h>


// Parámetros físicos

double hbar = 1.0;  // Constante reducida de Planck

double m = 1.0;     // Masa de la partícula

double omega = 1.0; // Frecuencia del oscilador armónico


// Tamaño del paso y número de puntos de la cuadrícula

double dx = 0.01;

int N = 1000;


// Función para calcular las funciones propias usando el método
de Numerov

void eigenfunction(double* y, double* x, int n) {

    double alpha = sqrt(m * omega / hbar);

    y[0] = 0.0;

    y[1] = 0.1;  // Valor inicial arbitrario

    double fn_minus_1 = 1.0 + (1.0 / 12.0) * alpha * alpha *
x[0] * x[0];


    FILE *file = fopen("archivocuantico.txt", "w");

    if (file == NULL) {

        printf("No se pudo abrir el archivo.\n");

        return;

    }
```

```
    for (int i = 1; i < N - 1; i++) {
        double fn = 1.0 + (1.0 / 12.0) * alpha * alpha * x[i] *
x[i];
        double fn_plus_1 = 1.0 + (1.0 / 12.0) * alpha * alpha *
x[i + 1] * x[i + 1];
        y[i + 1] = ((12.0 - 10.0 * fn) * y[i] - fn_minus_1 * y[i
- 1]) / fn_plus_1;
        fn_minus_1 = fn;

        fprintf(file, "%f %f\n", x[i], y[i]);
    }
    fclose(file);
}

int main() {
    double x[N];
    double y[N];
    // Inicialización de la cuadrícula
    for (int i = 0; i < N; i++) {
        x[i] = i * dx;
    }
```

Este código resuelve la ecuación de Schrödinger para el oscilador armónico cuántico y calcula la función de onda correspondiente. Puedes ajustar el valor de la energía inicial Ey

Cálculo y representación en Python de las funciones propias:

Para calcular y representar las funciones propias en Python.

import sympy as sp

import matplotlib.pyplot as plt

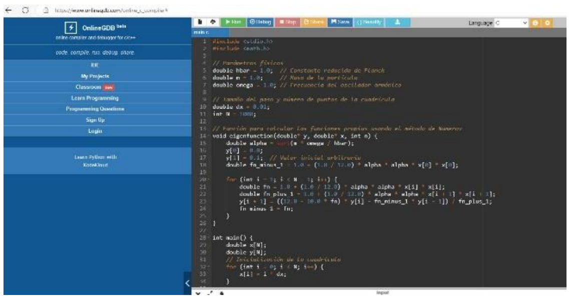
import numpy as np

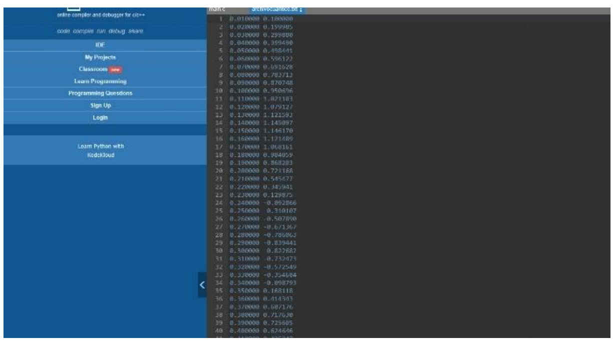
```
# Símbolos y función de Hermite
x, n = sp.symbols('x n')
hermite = sp.hermite(n, x)

# Cálculo de los primeros polinomios de Hermite
hermite_0 = hermite.subs(n, 0)

# Representación gráfica de H0(x)
sp.plot(hermite_0, (x, -5, 5), title="H0(x)")

# Transformación de Hermite a función de onda
psi_0 = (1/sp.sqrt(sp.factorial(0))) * sp.exp(-x**2/2) * hermite_0

# Crear una función numérica a partir de la función de onda
psi_0_numeric = sp.lambdify(x, psi_0, "numpy")

# Crear puntos para la gráfica
x_values = np.linspace(-5, 5, 100)
y_values = psi_0_numeric(x_values)

# Representación gráfica de la función de onda para n = 0
plt.figure()
plt.plot(x_values, y_values)
plt.title("Función de onda para n = 0")
plt.xlabel("x")
plt.ylabel("ψ(x)")
plt.grid()

# Cálculo de los siguientes polinomios de Hermite
```

```
hermite_1 = hermite.subs(n, 1)
hermite_2 = hermite.subs(n, 2)
hermite_3 = hermite.subs(n, 3)

# Transformación de Hermite a funciones de onda
psi_1 = (1/sp.sqrt(sp.factorial(1))) * sp.exp(-x**2/2) * hermite_1
psi_2 = (1/sp.sqrt(sp.factorial(2))) * sp.exp(-x**2/2) * hermite_2
psi_3 = (1/sp.sqrt(sp.factorial(3))) * sp.exp(-x**2/2) * hermite_3

# Crear funciones numéricas a partir de las funciones de onda
psi_1_numeric = sp.lambdify(x, psi_1, "numpy")
psi_2_numeric = sp.lambdify(x, psi_2, "numpy")
psi_3_numeric = sp.lambdify(x, psi_3, "numpy")

# Crear puntos para las gráficas
y_values_1 = psi_1_numeric(x_values)
y_values_2 = psi_2_numeric(x_values)
y_values_3 = psi_3_numeric(x_values)

# Representación gráfica de las funciones de onda para n = 1, 2, 3
plt.figure()
plt.plot(x_values, y_values_1, label="n = 1")
plt.plot(x_values, y_values_2, label="n = 2")
plt.plot(x_values, y_values_3, label="n = 3")
plt.title("Funciones de onda para n = 1, 2, 3")
plt.xlabel("x")
plt.ylabel("ψ(x)")
plt.legend()
plt.grid()
```

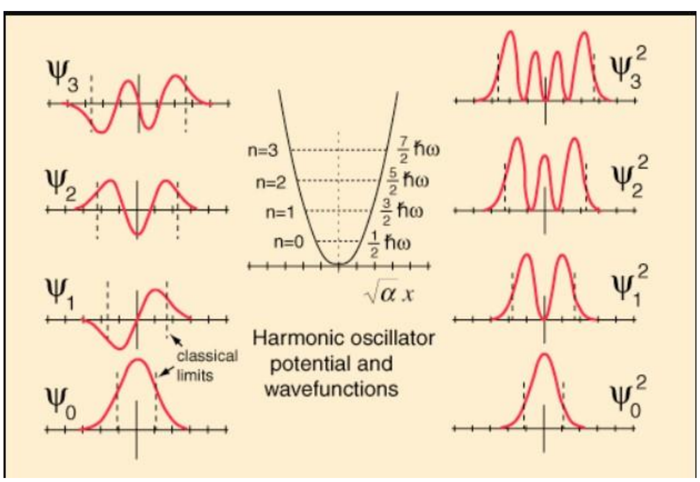
Resultados esperados

Resultados esperados en el lenguaje

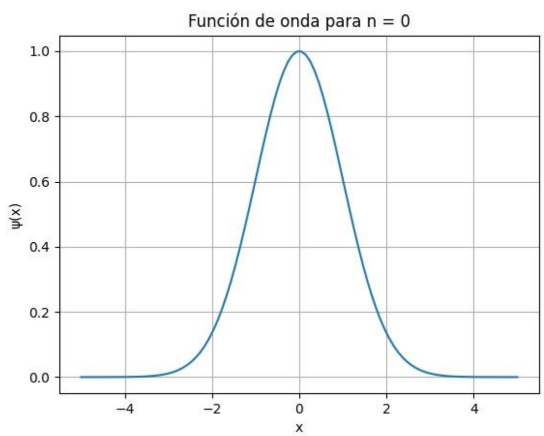


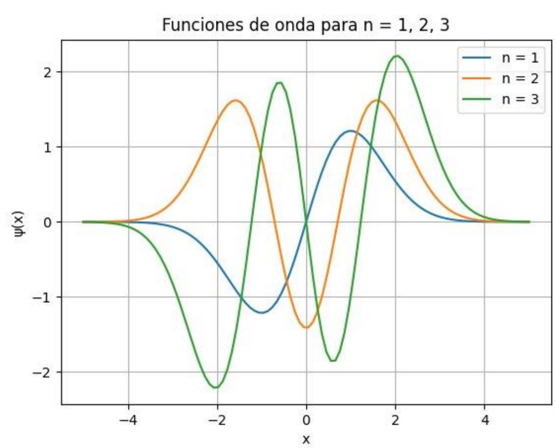


Ejemplo de graficas



Resultados esperados: cuando 0,1,2,3





Conclusión:

El código proporcionado demuestra cómo calcular los polinomios de Hermite y las funciones de onda asociadas para valores específicos de �n. Las visualizaciones gráficas ayudan a comprender mejor el comportamiento de estas funciones en el intervalo de $[-5,5][-5,5]$. Este código puede ser útil para estudiantes y profesionales que deseen explorar y comprender los polinomios de Hermite y sus aplicaciones en la física cuántica.