Ecuación de difusión en una dimensión

Josué Juárez Morales

La ecuacion de difusión la encontramos en diferentes capos de la física. Si analizamos una función de densidad local de partículas ρ, ésta obedece una relación

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot \mathbf{j} \tag{1}$$

donde $\mathbf{j}$ es una densidad de flujo de corriente. También se conoce ésta interpretación como aproximación de Fick. En ésta describimos el desplazamiento natural de los sistemas de lugarres de mayor concentración a menor concentración.

$$\frac{\partial u}{\partial t} = D_0 \nabla^2 \rho \tag{2}$$

La ecuación de difusión en una dimension tiene la forma

$$\frac{\partial u}{\partial t} = \nu \frac{\partial^2 u}{\partial x^2}, \tag{3}$$

donde v es el coeficiente de difusión. En ésta ocación tenemos una segunda derivada en juego, y como tal debemos discretizarl de manera apropiada. Consideramos la expansión en series de Taylor para una variación hacia adelante $+\Delta x$ y otra para hacia atras $-\Delta x$ tal que

$$u_{i+1} = u_i + \left(\frac{\partial u}{\partial x}\right)_i \Delta x + \frac{1}{2}\left(\frac{\partial^2 u}{\partial x^2}\right)_i \Delta x^2 + \frac{1}{6}\left(\frac{\partial^3 u}{\partial x^3}\right)_i \Delta x^3 + O(\Delta x^4), \tag{4}$$

$$u_{i-1} = u_i - \left(\frac{\partial u}{\partial x}\right)_i \Delta x + \frac{1}{2}\left(\frac{\partial^2 u}{\partial x^2}\right)_i \Delta x^2 - \frac{1}{6}\left(\frac{\partial^3 u}{\partial x^3}\right)_i \Delta x^3 + O(\Delta x^4), \tag{5}$$

al sumar estas dos cantidades y despejar el símbolo de segunda derivada evaluada en el punto x_i obtenemos

$$\left(\frac{\partial^2 u}{\partial x^2}\right) = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} + O(\Delta x^2), \tag{6}$$

con esto podemos discretizar la ecuación de difusion en una dimensión

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \nu \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2}, \tag{7}$$

teniendo una condición inicial, la unica incognita en la ecuacion es u_i^{n+1}, despejando para esta incognita

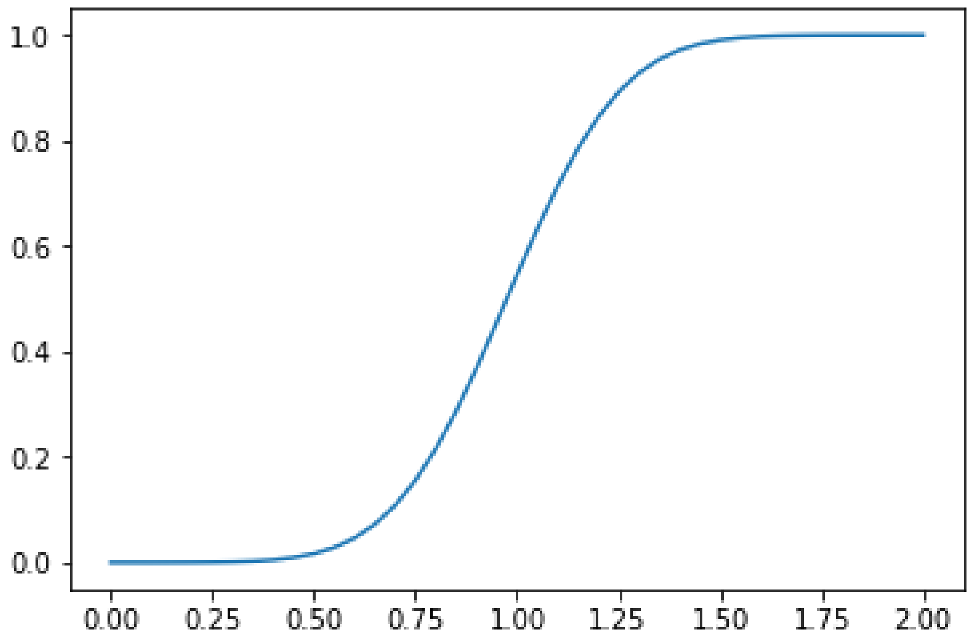
$$u_i^{n+1} = u_i^n + \frac{\nu \Delta t}{\Delta x^2}(u_{i+1}^n - 2u_i^n + u_{i-1}^n), \tag{8}$$

la ultima ecuación nos permite avanzar en el tiempo dada una condición inicial.

Realizamos un ejemplo con la función pulso.

[0]:
```python
import numpy as np #importa numpy.
import matplotlib.pyplot as plt #importa la herramienta para graficar
%matplotlib inline
#hace que las gráficas aparescan en la siguiente linea

def pulso(x0, x1, x): #define la función pulso
  if x < x0 or x > x1:
    return(0.0)
  else:
    return(1.0)

L = 2 #el tamaño de nuestro intervalo en x
nx = 41
dx = L/(nx-1) #la distancia que hay entre cada punto discretizado x (dx)
T = 1.0 #intervalo total de tiempo
nt = 2000 #número de veces que se discretiza la variable tiempo
c = 1.0 #velocidad de la onda (e.d.)
nu = 0.3 #viscocidad
sigmamax = 0.1
dt = dx*sigmamax/nu


u = np.linspace(0, L, nx) #np.linspace genera un vector con nx entradas que␣
↪contiene números igualmente espaciados en un intervalo (0,L)
x = np.linspace(0, L, nx) #generamos dos porque uno va a entrar a la funcion␣
↪pulso

for i in range(len(x)):
  u[i] = pulso(1.0, 2.0, x[i])


un = np.zeros(nx) #crea un vector temporal de tamaño nx con entradas ceros
for n in range(nt): #genera el loop nt veces
  un = u.copy() #copia los elementos de u al vector temporal un
  for i in range(1,nx-1): #el loop realiza las operaciones para calcular el␣
↪u^{n+1}_{i}, pero comienza con el elemento u[1] y no u[o] (se salta el primer␣
↪elemento)
    u[i] = un[i] + nu*dt*(un[i+1]-2.0*un[i] + un[i-1])/dx*dx

plt.plot(x,u)
```

[0]:
```
[<matplotlib.lines.Line2D at 0x7f08ceecc780>]
```


1.0
0.8
0.6
0.4
0.2
0.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00