Prueba Corta 5

July 1, 2021

[1]:
```
import numpy as np
```

1 Solución Analítica

1. $V_A = 0$
2. $F_E = N_A \pi r^2$
3. $F_S = N_A \pi r^2 + \delta\left(N_A \pi r^2\right)$
4. $V_G = 0$
5. $V_C = (-R_A)\pi r^2 \delta z$

$$0 = N_A \pi r^2 - \left(N_A \pi r^2 + \delta\left(N_A \pi r^2\right)\right) - (-R_A)\pi r^2 \delta z$$
$$\delta\left(N_A\right) = -(-R_A)\delta z$$
$$\frac{dN_A}{dz} + \kappa C_A = 0$$

$$N_A = -D_{AB}\frac{dC_A}{dz} + \nu C_A$$

Se asume que gobierna la difución, entonces $\nu \approx 0$.

$$\frac{d^2 C_A}{dz^2} = \frac{\kappa C_A}{D_{AB}}$$

Suponiendo $C_A = C_1 \exp\left(mz\right)$:

$$m^2 = \frac{\kappa}{D_{AB}}$$
$$m = \pm\sqrt{\frac{\kappa}{D_{AB}}}$$

$$C_A = C_1 \exp\left(z\sqrt{\frac{\kappa}{D_{AB}}}\right) + C_2 \exp\left(-z\sqrt{\frac{\kappa}{D_{AB}}}\right)$$
$$C_A = A\sinh\left(z\sqrt{\frac{\kappa}{D_{AB}}}\right) + B\cosh\left(z\sqrt{\frac{\kappa}{D_{AB}}}\right)$$

$$B = C_{A0}$$

$$A = \frac{\alpha C_{A0} - C_{A0}\cosh\left(L\sqrt{\frac{\kappa}{D_{AB}}}\right)}{\sinh\left(L\sqrt{\frac{\kappa}{D_{AB}}}\right)}$$

$$C_A = \frac{\alpha C_{A0} - C_{A0}\cosh\left(L\sqrt{\frac{\kappa}{D_{AB}}}\right)}{\sinh\left(L\sqrt{\frac{\kappa}{D_{AB}}}\right)}\sinh\left(z\sqrt{\frac{\kappa}{D_{AB}}}\right) + C_{A0}\cosh\left(z\sqrt{\frac{\kappa}{D_{AB}}}\right)$$

2 Planteamiento para la solución numérica

Sea $\lambda = dC_A/dz$:

$$\frac{d\lambda}{dz} = \frac{\kappa C_A}{D_{AB}}$$
$$\frac{dC_A}{dz} = \lambda$$

[2]:
```
# Parámetros
k=0.0014
C0=856.08
alpha=0.9
CL=C0*alpha
L=18.05
Dab=0.0028098
h=0.0001
```

[3]:
```
def Analítica(z):
    return (CL-C0*np.cosh(L*np.sqrt(k/Dab)))/(np.sinh(L*np.sqrt(k/Dab)))*np.
↪sinh(z*np.sqrt(k/Dab))+C0*np.cosh(z*np.sqrt(k/Dab))
```

[4]:
```
Analítica(4.33)
```

[4]: 40.33108674771938

[5]:
```
def Derivada(f,h,t):
    x1=f(t-h)
    x2=f(t+h)
    dfdt=(x2-x1)/(2*h)
    return dfdt
```

[6]:
```
u0=Derivada(Analítica,h,0)
u0
```

[6]: -604.2802160686733

[7]:
```python
f = lambda C: k*C/Dab # ODE
```

[8]:
```python
def RK4(f,zf):
    '''
    Esta función corresponde a una función que soluciona una EDO por RK4␣
↪iterando sobre un y = np.arange(y0,yf,h)

    Parámetros
    ------------------------
    f: función al lado derecho de la EDO
    zf: valor de la variable independiente para el cual se desea encontrar el␣
↪valor de C.
    ------------------------
    C: valor de C en zf.
    '''
    u=u0
    C=C0
    z=0
    while z<=zf:
        m1=h*u
        k1=h*f(C)
        m2=h*(u+k1/2)
        k2=h*f(C+m1/2)
        m3=h*(u+k2/2)
        k3=h*f(C+m2/2)
        m4=h*(u+k3)
        k4=h*f(C+m3)
        u+=(k1+2*k2+2*k3+k4)/6
        C+=(m1+2*m2+2*m3+m4)/6
        z+=h
    return C
```

[9]:
```python
RK4(f,4.33)
```

[9]: 40.33107921194848