PROBLEM 2 - Solution of Diffusion-Reaction System

Gisela Martí Guerrero, ICC 2023

This program simulates a diffusion-reaction system using an implicit method. The system involves two chemical species, A and B, that diffuse and react with each other over time.

$$A + B \longrightarrow Prod$$

where $\frac{dc_A}{dt} = \frac{dc_B}{dt} = D\nabla^2 C_{A,B} - kC_A C_B$.

The temporal evolution of A and B concentrations is solved, with initial conditions $C_{A,B}(x,0) = C_0 \pm \delta_{A,B}(x)$ and periodic boundary conditions.

The decay exponent α is also etimated. This exponent defines the type of regime of the system (reactive control regime for $\alpha = 1$ and diffusive control regime for $\alpha < 1$. It follows the expression:

$$\overline{c_A}(t) = \langle C_A(x,t) \rangle_x = t^{-\alpha}$$

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.sparse import diags
from matplotlib.animation import FuncAnimation, PillowWriter
```

In [2]:
```
# Define parameters
L = 1.0                         # Length
dL = 100                        # Grid points
dx = L/dL                       # Subintervals
xrange = np.arange(0,L+dx,dx)
t = 10000                       # Total time
tp = 1000                       # Time points
dt = t/(tp-1)                   # Subintervals
trange = np.arange(0,t+dt,dt)
D = 1e-5                        # Diffusion coefficient
k = 1e-3                        # Velocity constant
animation_frames = 1000
animation_name = "Diffusion.gif"
```

A tridiagonal matrix is created with the following form:

$$\begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ -a & 1+2a & -a & \dots & 0 \\ 0 & -a & 1+2a & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where $a = \frac{D\Delta t}{\Delta x^2}$

In [3]:
```
# Create tridiagonal matrix with scipy
dim = dL+1
a = D * dt / dx**2
main_diagonal = 1 + 2 * a
off_diagonal = -a
matrix = diags([off_diagonal, main_diagonal, off_diagonal], [-1, 0, 1], shape=(dim, dim
# Set boundary conditions
matrix[0, 0] = 1
matrix[-1, -1] = 1
matrix[0, 1] = 0
matrix[-1, -2] = 0
```

Now the concentrations C_A and C_B are initialized. We set $C_A = C_B = 1$

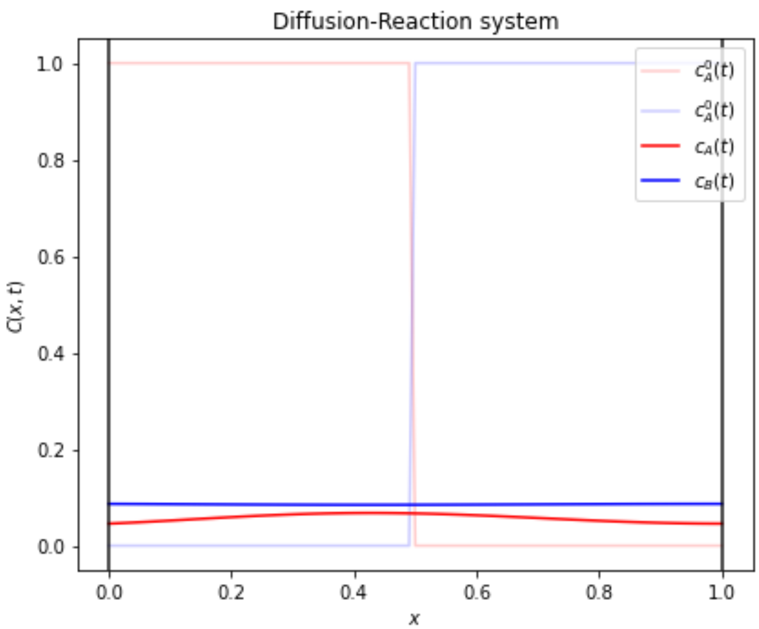
In [4]:
```
# Concentration arrays
conc = np.zeros(shape=(2, dL+1))
_, x_points = conc.shape
conc[0][0:int(x_points/2)] += 1
conc[1][int(x_points/2):] += 1
c_init = conc.copy()
```

We start the main integration loop. It enforces periodic boundary conditions by setting the first and last elements of each concentration array equal to the second-to-last element. The concentration profiles are saved at certain intervals, and calculated the α parameter based on the mean concentration of C_A and the current time. Then the concentrations of both substances are updated using a numerical integration.

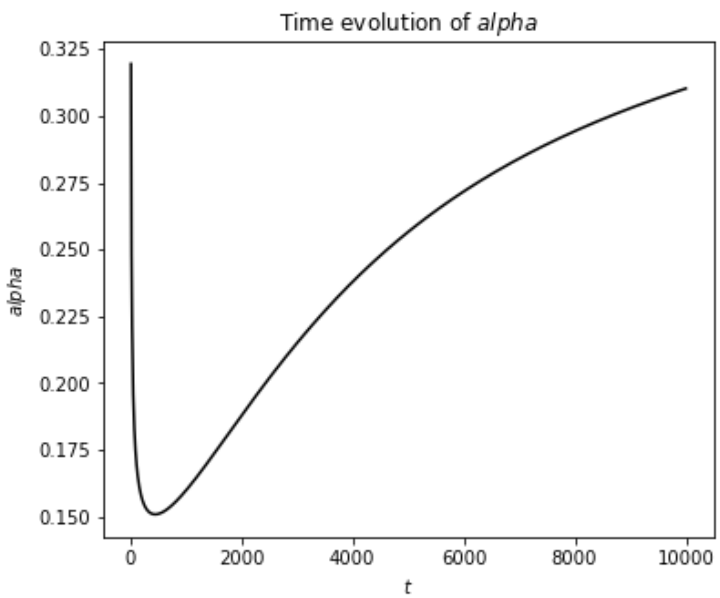
In [5]:
```
# Main integration loop
ct = [[],[]]
alpha = []

for i,ti in enumerate(trange):

    # Periodic conditions
    conc[0][0] = conc[0][-1] = conc[0][-2]
    conc[1][0] = conc[1][-1] = conc[1][-2]

    # Save frames
    if i%int(tp/animation_frames) == 0 :
        ct[0].append(conc[0].copy())
        ct[1].append(conc[1].copy())

        ai = -np.log(conc[0].mean()) / np.log(ti)
        alpha.append(ai)

    # Update concentrations
    conc[0] = np.linalg.inv(matrix.toarray())@conc[0] - dt*k*conc[0]*conc[1]
    conc[1] = np.linalg.inv(matrix.toarray())@conc[1] - dt*k*conc[1]*conc[1]

ct = np.array(ct,dtype=object)
alpha = np.array(alpha)
```

```
C:\Users\gisee\AppData\Local\Temp\ipykernel_38196\1643991874.py:16: RuntimeWarning: di
vide by zero encountered in log
  ai = -np.log(conc[0].mean()) / np.log(ti)
```

In [6]:
```python
# Start animation
def GIF(frame):
    """Function that creates a frame for the GIF."""
    ax.clear()
    cA0, = ax.plot(xrange,c_init[0],c="red",alpha=0.2,label="$c_A^0(t)$")
    cB0, = ax.plot(xrange,c_init[1],c="blue",alpha=0.2,label="$c_A^0(t)$")
    cAt, = ax.plot(xrange,ct[0][frame],c="red",label="$c_A(t)$")
    cBt, = ax.plot(xrange,ct[1][frame],c="blue",label="$c_B(t)$")
    wall1 = ax.axvline(L,ymin=0,c="k",alpha=0.9)
    wall2 = ax.axvline(0,ymin=0,c="k",alpha=0.9)
    ax.set_xlabel("$x$")
    ax.set_ylabel("$C(x,t)$")
    ax.set_title("Diffusion-Reaction system")
    ax.legend(loc="upper right")
    return cA0,cB0,cAt,cBt,wall1,wall2

fig,ax = plt.subplots(figsize=(6,5))
animation = FuncAnimation(fig,GIF,frames=animation_frames,interval=20,blit=True,repeat=
animation.save(animation_name,dpi=120,writer=PillowWriter(fps=25))
fig.tight_layout()
plt.show()
```



In [7]:
```
# Time evolution of alpha
alpha_t = trange[::int(tp/animation_frames)]
plt.figure(figsize=(6,5))
plt.plot(alpha_t[1:],alpha[1:],c="black")
plt.title('Time evolution of $alpha$')
plt.xlabel('$t$')
#plt.xlim([0,alpha_t[-1]])
plt.ylabel('$alpha$')
plt.show()
```



It can be seen how both species diffuse between each other and both concentrations decrease. Playing with the diffusion constant (D) and reaction velocity constant (k) one can see which one is controlling the system.

In the case displayed in this program, we can see how α first rapidly decreases, and then it starts slowly rising again, which indicates a transition in the regime of the system. Initially, the system is in a reactive control regime, since the of C_A decreases quickly with time, which suggests that chemical reactions play a dominant role in controlling the behavior of the system during this phase. As α starts to increase, the system transitions toward a diffusive control regime, where the concentration of C_A is expected to decrease more slowly with time, and diffusion processes become more influential compared to chemical reactions.