**Interface gráfica do usuário, Texto

Descrição gerada automaticamenteInterface gráfica do usuário, Texto

Descrição gerada automaticamente**

**IPRJ - Métodos Numéricos para Equações Diferenciais – 2024/2**

**Trabalhos 3 e 4**

**Prof. Grazione de Souza Boy**

**Nome da aluna: Júlya Matias de Alcantara**

**Matrícula: 2020.103.574-11**

**Nome do aluno: André Flávio das Chagas Barros**

**Matrícula: 2020.100.783-11**

Nova Friburgo – 09/12/2024

# Resumo

Este relatório apresenta a aplicação de métodos numéricos para resolver equações diferenciais parciais relacionadas a modelos físicos. No Trabalho 3, utilizamos o método de diferenças finitas com formulação totalmente implícita para simular o perfil de concentração ao longo do tempo e da posição em um sistema descrito por uma equação de difusão. No Trabalho 4, aplicamos o método de diferenças finitas com formulação explícita para resolver uma equação de advecção-difusão, analisando a evolução da concentração com variação de parâmetros e refinamento da malha. Os resultados foram avaliados por meio de gráficos e tabelas, destacando a influência de condições de contorno, estabilidade e precisão numérica.

# Trabalho 3

Para esta etapa do trabalho, foram adotados os valores iniciais apresentados na Tabela 1.

|  |  |
| --- | --- |
| **Parâmetros** | **Valores** |
| Comprimento do domínio (Lx) | 1m |
| Número de pontos espaciais (Nx) | 50 |
| Constante α | 0.01 |
| Número de pontos no tempo (Nt) | 100 |
| Condição de contorno em x=0 (CE) | 1 |
| Tempo total de simulação (T\_max) | 1s |
| Tolerância | m |
| Constante K | 0.1 |
| Passo de tempo (dt) | 0.01 |

Tabela 1 - Valores iniciais

Utilizando uma formulação totalmente implícita oriunda da aplicação de aproximações atrasada no tempo e centrada no espaço, é possível observar na figura 1 como a concentração se comporta em função de x e dependente do tempo.

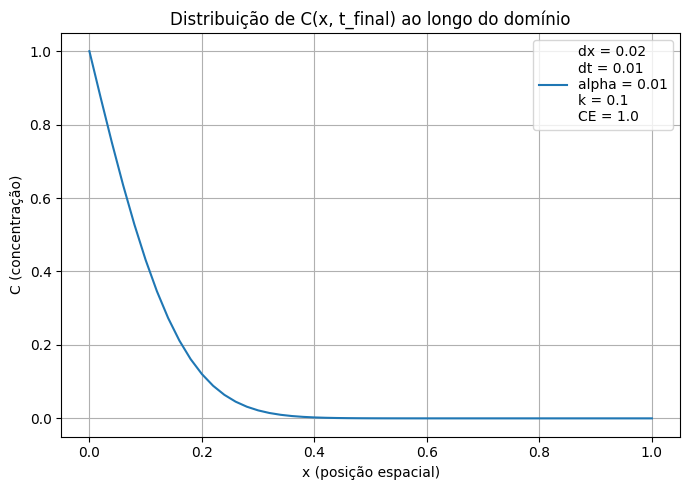


Figura 1 - Gráfico de Referência

Com o objetivo de estudar o impacto individual de cada parâmetro no comportamento do sistema, realizamos uma série de testes alterando os parâmetros um de cada vez, enquanto mantemos os demais constantes.

* **Variação do número de pontos espaciais (Nx)**

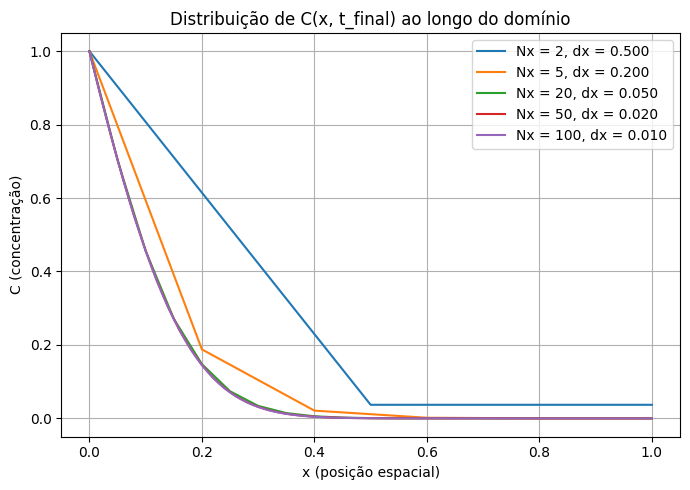


Figura 2 – Concentração X Posição (Variando Nx)

Podemos concluir que o número de pontos influencia fortemente na condição da curva. Portanto, temos que **quanto maior o número de pontos espaciais, melhor a aproximação**, o que é refletido na sua aparência no gráfico. Temos que ter em mente, porém, que mais pontos significa maior poder computacional necessário para processá-los. Logo, existe um ponto em que, a partir dele, deixa de ser vantajoso aumentar a quantidade, pois apenas o custo cresce sem grandes mudanças no resultado.

* **Variação da constante K**

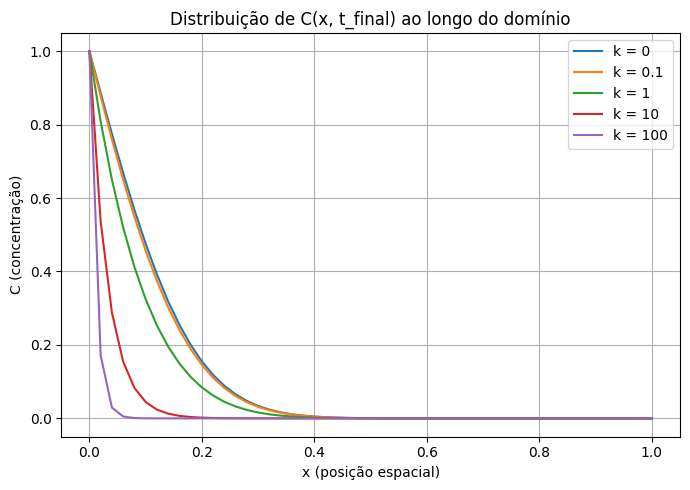


Figura 3 – Concentração X Posição (Variando K)

O valor K = 100 teve o melhor desempenho, resultando em uma boa convergência, enquanto K = 0 apresentou o pior desempenho e convergiu em uma maior distância, o que indica que quanto **maior** o seu valor, mais eficiente a convergência.

Quanto à aparência menos suave das curvas com maior K, isso deve à quantidade de pontos, que aparenta ser baixa demais para uma curva tão íngreme.

* **Variação da constante α**

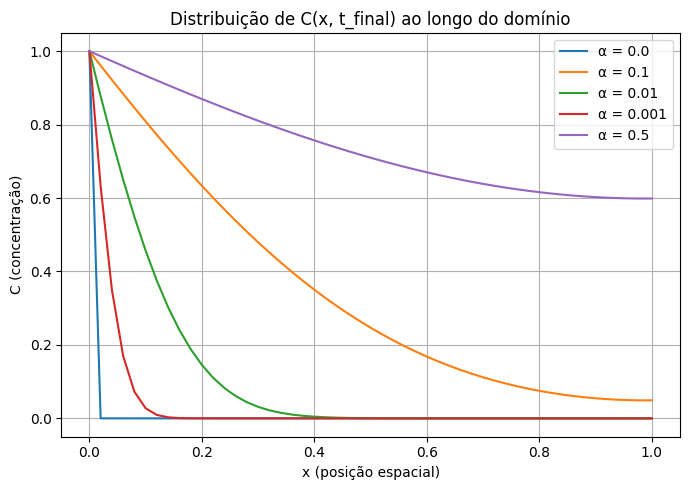


Figura 4 – Concentração X Posição (Variando α)

O valor α = 0.001 teve o melhor desempenho, resultando em uma boa convergência. Valores maiores que 0.01 tenderam a apresentar piores resultados, sem atingir a convergência, o que indica que quanto **menor** o seu valor, mais eficiente.

Temos também o valor α = 0 que despenca logo no início, o que indica que talvez tenha ocorrido algum erro e não seja uma boa opção a ser escolhida.

* **Variação da condição de contorno em x=0 (CE)**

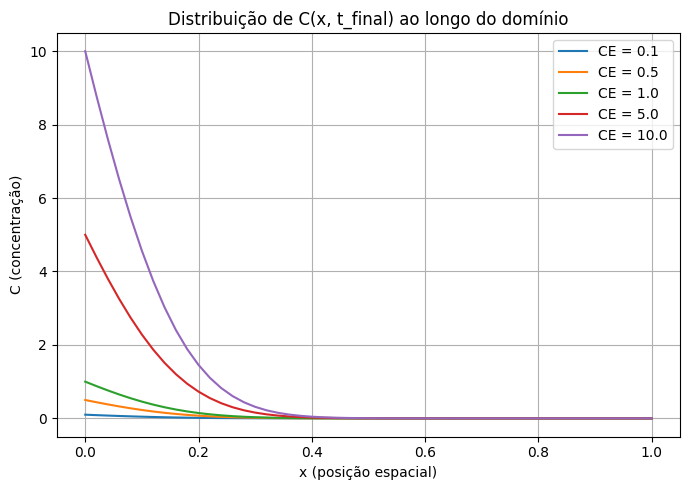


Figura 5 – Concentração X Posição (Variando CE)

Os valores de CE 1 tiveram o melhor desempenho e atingiram a convergência com maior rapidez, enquanto CE > 1 apresentaram pior desempenho, atingindo a convergência numa posição mais distante, o que indica que quanto **menor** o seu valor, mais eficiente a convergência.

* **Variação do passo de tempo (dt)**

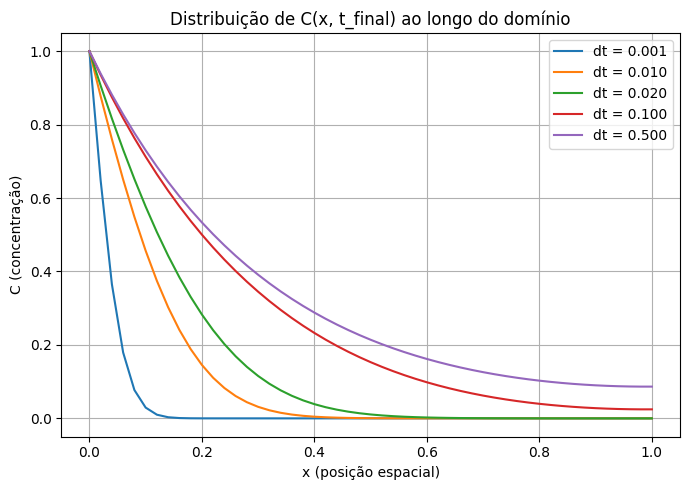


Figura 6 – Concentração X Posição (Variando dt)

Os valores de dt mais baixos tiveram o melhor desempenho, sendo dt = 0.001 o ponto de melhor eficiência, enquanto valores mais altos se mostraram mais distantes do valor final (zero) o que indica que quanto **menor** o seu valor, mais eficiente a convergência.

Script Utilizado (Trabalho 3)

import numpy as np

import matplotlib.pyplot as plt

titulo = 'variando\_final'

parametros = [

    {"label": 'Caso base', "Lx": 1.0, "alpha": 0.01, "k": 0.1, "CE": 1.0, "Nx": 50, "Nt": 100,"tolerance": 1e-6, "t\_max": 1.0}, # CASO BASE

]

plt.figure(figsize=(7, 5))

for params in parametros:

    Lx = params["Lx"]

    alpha = params["alpha"]

    k = params["k"]

    CE = params["CE"]

    Nx = params["Nx"]

    Nt = params["Nt"]

    t\_max = params["t\_max"]

    dt = t\_max / Nt

    dx = Lx / Nx

    label = params["label"]

    s = alpha \* dt / dx\*\*2

    tolerance = params["tolerance"]

    max\_iterations = 1000

    C = np.zeros(Nx + 1)

    C\_new = np.zeros(Nx + 1)

    errors = []

    for n in range(Nt):

        iteration = 0

        max\_error = float('inf')

        while max\_error > tolerance and iteration < max\_iterations:

            max\_error = 0.0

            for i in range(1, Nx):

                old\_value = C\_new[i]

                C\_new[i] = (C[i] + s \* (C\_new[i - 1] + C\_new[i + 1])) / (1 + (2 \* s) + (k \* dt))

                max\_error = max(max\_error, abs(C\_new[i] - old\_value))

            C\_new[0] = CE

            C\_new[-1] = C\_new[-2]

            iteration += 1

        errors.append(max\_error)

        C[:] = C\_new[:]

    x = np.linspace(0, Lx, Nx + 1)

    plt.plot(x, C, label=label)

plt.title("Distribuição de C(x, t\_final) ao longo do domínio")

plt.ylabel("C (concentração)")

plt.xlabel("x (posição espacial)")

# plt.xlim(0, 0.5)

plt.grid(True)

plt.legend()

plt.tight\_layout()

plt.savefig(f"plots/T3/{titulo}.png", bbox\_inches='tight')

plt.show()

# Trabalho 4

Para esta etapa do trabalho, foram adotados os valores iniciais apresentados na Tabela 2.

|  |  |
| --- | --- |
| **Parâmetros** | **Valores** |
| Comprimento do domínio (Lx) | 1m |
| Número de pontos espaciais (Nx) | 100 |
| Constante α | 0.01 |
| Constante u | 0.1 |
| Condição de contorno em x=0 (CE) | 1 |
| Tempo total de simulação (T\_max) | 20s |
| dt | 90% da restrição dada |

Tabela 2 - Valores iniciais

Utilizando uma formulação explícita oriunda da aplicação de aproximações avançada no tempo, centrada no espaço para a derivada segunda e recuada no espaço para a derivada primeira espacial. Para o passo de tempo, foi utilizada a restrição . É possível observar na figura 7 como a concentração se comporta em função de x e dependente do tempo.

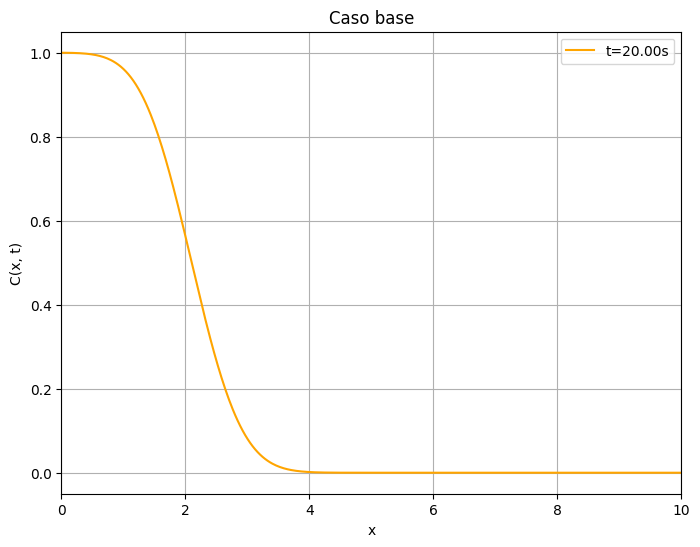


Figura 7 - Gráfico de Referência

Com o objetivo de estudar o impacto individual de cada parâmetro no comportamento do sistema, realizamos uma série de testes alterando os parâmetros um de cada vez, enquanto mantemos os demais constantes.

* **Variação do número de pontos espaciais (Nx)**

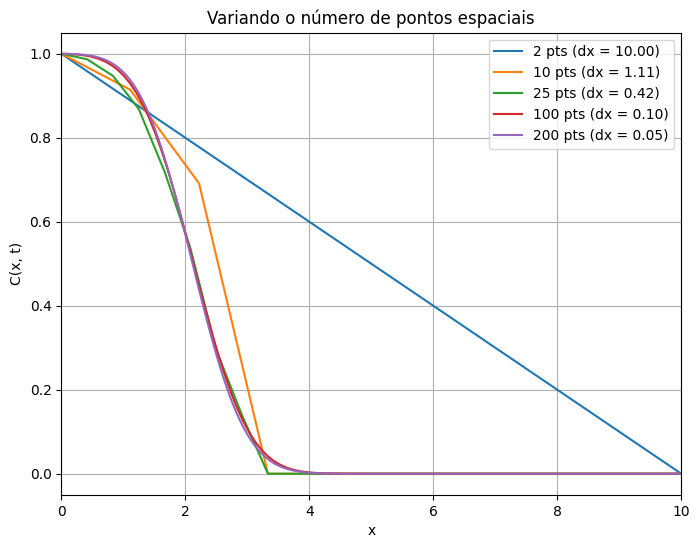


Figura 8 – Concentração X Posição (Variando Nx)

Aqui podemos aplicar a mesma lógica utilizada no Trabalho 3, que nos leva a concluir que **quanto maior o número de pontos espaciais, melhor a aproximação.** Novamente, porém, precisamos nos atentar ao poder computacional necessário para calcular os pontos.

* **Variação da constante α**

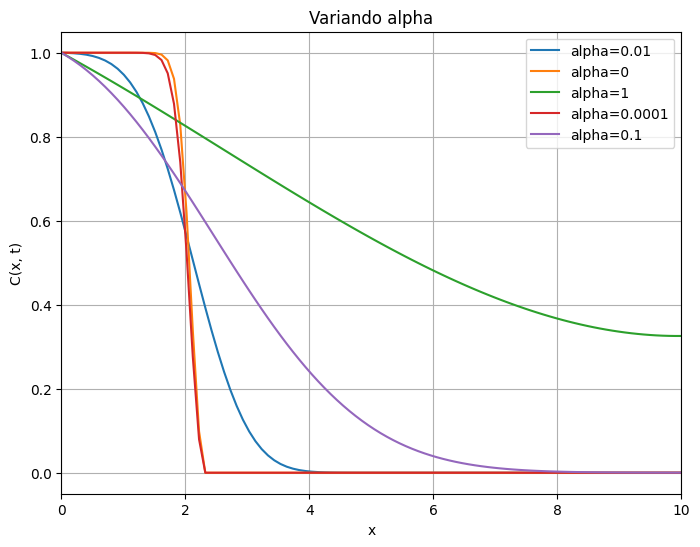


Figura 9 – Concentração X Posição (Variando α)

Também semelhantemente ao Trabalho 3, vemos que **valores menores de α convergem mais rápido**. Vale ressaltar que, quanto mais próximo de 0, mais a equação se aproxima de uma função de advecção (caracterizada por α = 0), como vemos no valor α = 0.0001.

* **Variação da constante u**

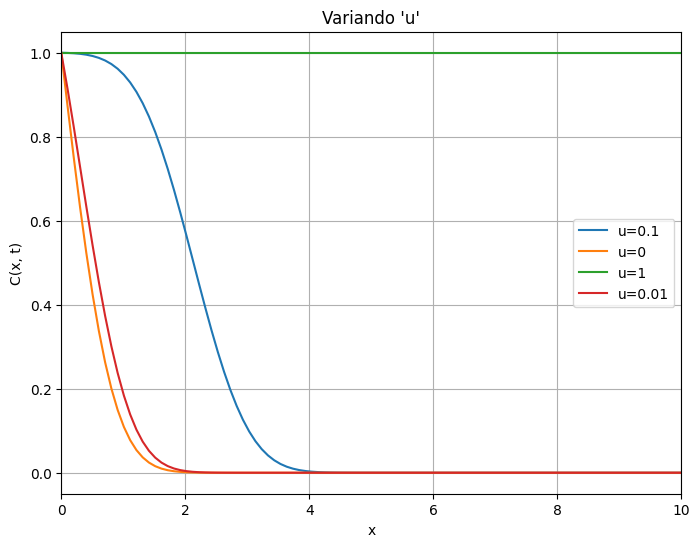


Figura 10 – Concentração X Posição (Variando u)

Analisando o gráfico, vemos que **valores menores de u convergem mais rápido**. Dessa vez, quanto mais próximo de 0, mais a equação se aproxima de uma função de difusão (caracterizada por u = 0), como vemos nos valores de u 0.01.

* **Variação da condição de contorno em x=0 (CE)**

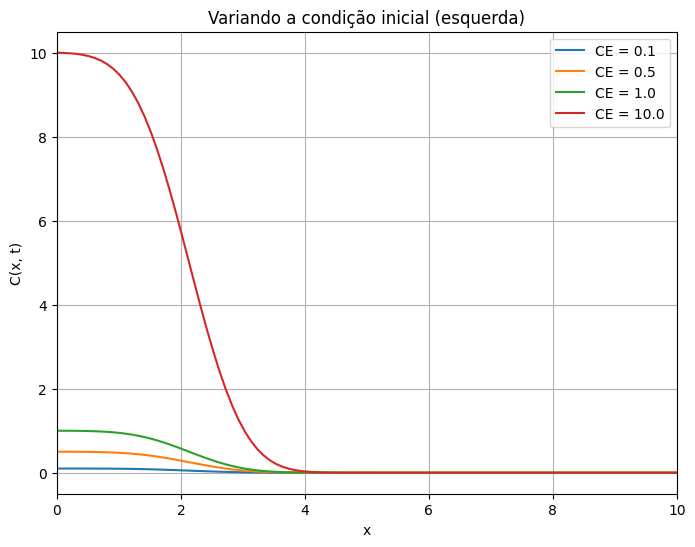


Figura 11 – Concentração X Posição (Variando CE)

Novamente, chegamos a conclusões semelhantes às do trabalho anterior, em que os valores de CE 1 tiveram o melhor desempenho e atingiram a convergência com maior rapidez, enquanto CE > 1 apresentaram pior desempenho, atingindo a convergência numa posição mais distante, o que indica que quanto **menor** o seu valor, mais eficiente a convergência.

* **Variação do passo de tempo (dt)**

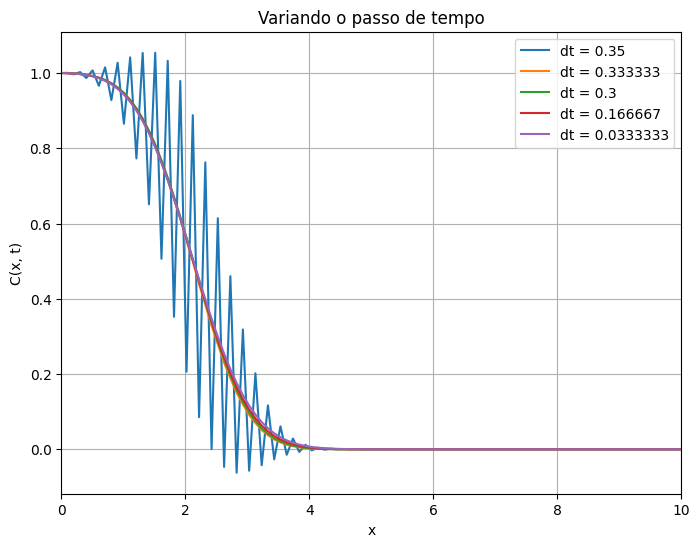


Figura 12 – Concentração X Posição (Variando dt)

Considerando a restrição dada e utilizando os nossos dados listados na Tabela 2, foi encontrado para o nosso caso que dt não poderia passar de 0.3333. Ao testar essa restrição na prática, foi possível confirmar a sua validade, já que o valor 0.35 (apenas 5% acima do limite) mostra uma oscilação anormal na representação gráfica.

Quanto aos valores que satisfazem a essa restrição, como 0.3333, 0.30, 0.16667, 0.0333 (respectivamente, 100%, 90%, 50% e 10% do limite), sua diferença não causou um impacto muito grande, porém ainda vemos uma tendência previsível e já constatada anteriormente: **quanto menor o passo de tempo, mais eficiente o resultado**.

* **Advecção e Difusão**

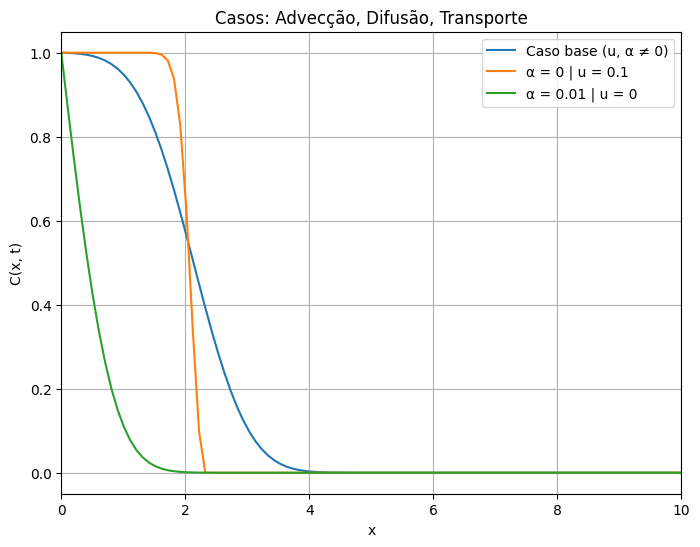


Figura 13 – Concentração X Posição

Foi feita também a comparação do caso base, que é considerado uma equação de transporte (valores de α e de u diferentes de zero) em relação às suas versões de advecção, caracterizado por α = 0, e também de difusão, em que o valor da constante u é zero.

Script Utilizado (Trabalho 4)

import numpy as np

import matplotlib.pyplot as plt

C = []

dados = [

    {

        'label': 'Caso base (u, α ≠ 0)',

        'Lx': 10.0,

        'Nx': 100,

        'alpha': 0.01,

        'u': 0.1,

        'CE': 1,

        'T\_max': 20,

    },

    {

        'label': 'α = 0 | u = 0.1',

        'Lx': 10.0,

        'Nx': 100,

        'alpha': 0,

        'u': 0.1,

        'CE': 1,

        'T\_max': 20,

    },

    {

        'label': 'α = 0.01 | u = 0',

        'Lx': 10.0,

        'Nx': 100,

        'alpha': 0.01,

        'u': 0,

        'CE': 1,

        'T\_max': 20,

    },

]

plt.figure(figsize=(8, 6))

plt.grid()

for it in dados:

    dx = it['Lx'] / (it['Nx'] - 1)

    dt = 1 / (2 \* it['alpha'] / dx\*\*2 + it['u'] / dx) \* 0.9

    Nt = int(it['T\_max'] / dt)

    x = np.linspace(0, it['Lx'], it['Nx'])

    C = np.zeros((it['Nx'], Nt + 1))

    C[:, 0] = 0

    C[0, :] = it['CE']

    for n in range(Nt):

        for i in range(1, it['Nx'] - 1):

            C[i, n + 1] = (dt \* it['alpha'] / dx\*\*2 \* (C[i + 1, n] - 2 \* C[i, n] + C[i - 1, n]) - dt \* it['u'] / dx \* (C[i, n] - C[i - 1, n]) + C[i, n])

        C[-1, n + 1] = C[-2, n + 1]

    plt.plot(x, C[:, Nt], label=f'{it["label"]}')

    plt.xlabel('x')

    plt.xlim(0, 10)

    plt.ylabel('C(x, t)')

    plt.title("Casos: Advecção, Difusão, Transporte")

    plt.legend()

plt.savefig(f"plots/T4/{'variando\_coefs'}.png", bbox\_inches='tight')