Aula 5 – Introdução a `matplotlib`: Visualização de Campos 2D

EMC410235 - Programação Científica para Engenharia e Ciência Térmicas

Prof. Rafael F. L. de Cerqueira

2025.2

Equação de Potencial e Propriedades da Equação de Laplace

A partir das hipóteses feitas, obtemos a equação do potencial de velocidade:

$$\boxed{\nabla^2 \phi = 0}$$

A velocidade é dada como o gradiente do potencial:

$$\vec{u} = \vec{\nabla}\phi = \left(\frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y}\right)$$

Propriedades importantes da equação de Laplace:

- É uma equação linear e homogênea.
- Suas soluções são funções harmônicas.
- **Superposição:** se ϕ_1 e ϕ_2 são soluções, então $\phi = \phi_1 + \phi_2$ também é.

Isso permite construir escoamentos complexos pela soma de potenciais elementares (ex: escoamento uniforme + fonte).

Exemplo: Fonte Pontual (Escoamento Radial)

Uma **fonte** libera um volume Q por unidade de tempo, uniformemente em todas as direções.

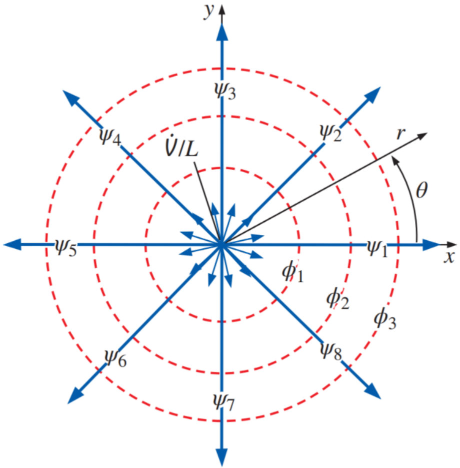
$$\frac{Q}{L} = 2\pi r u_r \quad \rightarrow \quad u_r = \frac{Q/L}{2\pi r}$$

Assim,

$$u_r = \frac{\partial \phi}{\partial r} = \frac{\dot{V}/L}{2\pi r} \qquad u_\theta = \frac{1}{r}\frac{\partial \phi}{\partial \theta} = 0$$

Desse modo, obtém-se:

$$\phi = \frac{Q/L}{2\pi r} \ln r$$



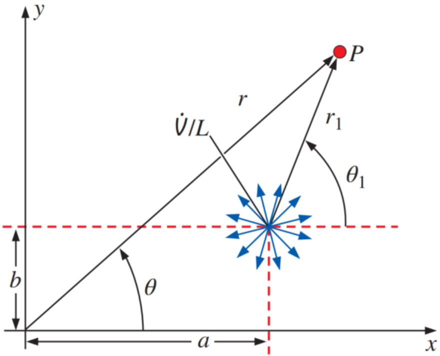
Fonte Deslocada (transladação para (a, b))

Podemos transladar a fonte para um ponto (a, b).
Dessa forma,

Potencial de velocidade:

$$\phi(x, y) = \frac{Q/L}{2\pi} \ln \sqrt{(x-a)^2 + (y-b)^2}$$

Notem que a **velocidade ainda aponta na direção radial a partir de** (a, b)**.**



Vamos plotar as diferentes informações contidas nessa função potencial?

Antes de começarmos: geração da malha 2D

Para visualizar os campos (potencial, função corrente ou velocidades), precisamos discretizar o domínio com uma malha 2D.

Exemplo de criação de malha com NumPy:

```
x_min = -2.0 # [m]
x_max = 2.0  # [m]
y_min = -2.0 # [m]
y_max = 2.0  # [m]

N_I = 50
N_J = 50

xx = np.linspace(x_min, x_max, num=N_I)
yy = np.linspace(y_min, y_max, num=N_J)
X, Y = np.meshgrid(xx, yy)
```

Função de potencial para uma fonte pontual

A seguir, definimos uma função que calcula o campo de potencial $\phi(x, y)$ gerado por uma fonte localizada em (a, b), com intensidade K. Para esse caso, $K = (Q/L)/2\pi$.

```
def phi_line_source(X, Y, K=0.0 , a=0.0, b=0.0):
    r2 = np.sqrt((X - a) ** 2 + (Y - b) ** 2)
    phi = K * np.log(r2)

    return phi
```

Cálculo e visualização do potencial

Chamamos a função `phi_line_source()` para calcular o campo escalar e utilizamos `plt.contourf()` para visualizá-lo como mapa de contorno preenchido.

```
a = 0.0 # posicao x da fonte
b = 0.0 # posicao y da fonte
Q = 1.0 # vazao volumetrica [m3/s]
L = 1.0 # comprimento [m]

phi_1 = phi_line_source(X, Y, a, b, Q, L)
#cont = plt.contour(X, Y, phi_1, levels=50, cmap='viridis')
cont = plt.contourf(X, Y, phi_1, levels=50, cmap='viridis')
plt.colorbar(cont, label='phi [m2/s]')
plt.xlabel('x [m]')
plt.ylabel('y [m]')
plt.axis(équal')
plt.grid(True)
plt.show()
```

Gerando uma animação frame a frame

Criamos uma animação movendo a posição a da fonte ao longo do eixo x, e salvamos cada frame como imagem numerada sequencialmente:

```
for i, a in enumerate(np.linspace(0, 1, num=50)):
  phi_1 = phi_line_source(X, Y, a, b, Q, L)

  cont = plt.contour(X, Y, phi_1, levels=50, cmap='viridis')
  plt.colorbar(cont, label='phi [m2/s]')
  plt.xlabel('x [m]')
  plt.ylabel('y [m]')
  plt.axis(équal')
  plt.grid(True)

  plt.savefig(féxemplo_animacao_{i:06d}.png')
  plt.close()
```

Essas imagens podem ser usadas com o `ffmpeg` para gerar um GIF ou vídeo. Exemplo no WSL: `ffmpeg -y -framerate 10 -i exemplo_animacao_%06d.png -c:v libx264 -pix_fmt yuv420p animacao.mp4`

Visualização alternativa com `pcolormesh`

Além de `contour` e `contourf`, também podemos usar `pcolormesh`, que plota diretamente a matriz de valores como um mapa de cores.

```
phi_1 = phi_line_source(X, Y, a, b, Q, L)

plt.figure(figsize=(6, 5))
pcm = plt.pcolormesh(X, Y, phi_1, shading=áuto', cmap='viridis')
plt.colorbar(pcm, label='phi [m2/s]')
plt.xlabel('x [m]')
plt.ylabel('y [m]')
plt.title('Campo de potencial com pcolormesh')
plt.axis(équal')
plt.grid(True)
plt.show()
```

Vantagens: mais rápido para malhas uniformes, útil para visualizações densas.

Obtendo o campo de velocidades a partir de $\phi(x, y)$

Sabemos que o vetor velocidade é o gradiente do potencial:

$$u = \frac{\partial \phi}{\partial x} \approx \frac{\phi_e - \phi_w}{2\Delta x} \qquad v = \frac{\partial \phi}{\partial y} \approx \frac{\phi_n - \phi_s}{2\Delta y}$$

Podemos calcular essas derivadas numericamente usando diferenças centrais.

Exemplo: componente u da velocidade:

```
def calculate_U_velocity(X, Y, phi):
    dx = X[0,1] - X[0,0]
    phi_e = np.roll(phi, -1, axis=1)
    phi_w = np.roll(phi, 1, axis=1)
    return (phi_e - phi_w) / (2 * dx)
```

Função análoga pode ser usada para calcular $v = \partial\phi/\partial y$, com diferenças na direção vertical.
Notem que estamos usando a função `np.roll()`

Visualização do campo de velocidades com quiver

Com as componentes de velocidade U e V obtidas a partir do potencial ϕ, podemos visualizá-las como vetores.

Cálculo de u **e** v**:**

```
U = calculate_U_velocity(X, Y, phi_1)
V = calculate_V_velocity(Y, Y, phi_1)

U[:, :2] = np.nan
U[:, -2:] = np.nan
V[:2, :] = np.nan
V[-2:, :] = np.nan
```

Visualização com quiver:

```
plt.close()
plt.figure(figsize=(6,
    5))
plt.quiver(X, Y, U, V)
plt.xlabel('x [m]')
plt.ylabel('y [m]')
plt.axis(équal')
plt.grid(True)
plt.show()
```

Para evitar artefatos nas bordas, substituímos as regiões de contorno por NaN.

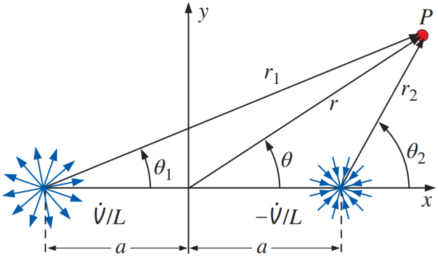
Superposição: Fonte + Sumidouro = Dipolo

Graças à linearidade da equação de Laplace, podemos somar soluções conhecidas para construir novos escoamentos.

Exemplo: uma fonte em $(a, 0)$ e um sumidouro em $(-a, 0)$, ambos com intensidade Q/L.

- **Potencial total:**

$$\phi(x, y) = \phi_{\text{fonte}}(x, y; a, 0) + \phi_{\text{sumidouro}}(x, y; -a, 0)$$



Esse escoamento é chamado **dipolo**.

Dipolo como superposição de duas fontes opostas

Também podemos construir um dipolo somando duas fontes com intensidades opostas:

- Fonte 1: localizada em $x = -a$, com intensidade $+K$
- Fonte 2: localizada em $x = +a$, com intensidade $-K$

Por conveniência, definimos K como a intensidade do dipolo: $K = a(Q/L)/\pi$

```
K = 1.0
a = 0.5
L = 1.0
Q = L * np.pi * K / a
phi_1 = phi_line_source(X, Y, -a, 0.0, Q, L) # fonte positiva
phi_2 = phi_line_source(X, Y, a, 0.0, -Q, L) # fonte negativa

phi_total = phi_1 + phi_2
```

A resultante é exatamente o campo de um dipolo, com linhas de corrente saindo de um ponto e entrando no outro.

Visualização combinada: contour + quiver

Podemos sobrepor o campo de contornos do potencial com o campo vetorial de velocidades para obter uma visualização completa do escoamento:

```
cont = plt.contour(X, Y, phi_total, levels=50, cmap='jet')
plt.quiver(X, Y, U, V)
plt.colorbar(cont, label='phi [m2/s]')
plt.xlabel('x [m]')
plt.ylabel('y [m]')
plt.axis(équal')
plt.grid(True)
plt.show()
```

Resultado: vemos tanto as linhas equipotenciais quanto a direção e magnitude do campo de velocidades.

Escoamento uniforme com ângulo α

Um escoamento uniforme pode ser representado por uma velocidade constante $\vec{U}$ inclinada de um ângulo α em relação ao eixo x.

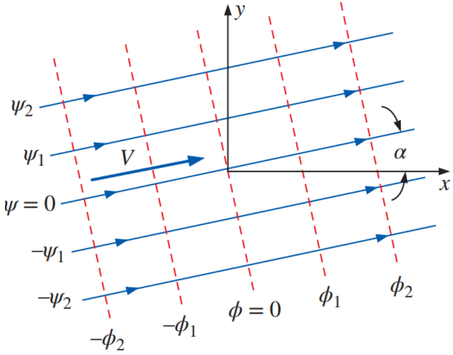
Componente da velocidade:

$$\vec{u} = (U\cos\alpha,\ U\sin\alpha)$$

Função potencial:

$$\phi(x, y) = U(x\cos\alpha + y\sin\alpha)$$

Esse escoamento é frequentemente usado em combinação com fontes, sumidouros ou cilindros via superposição.



Exemplo: escoamento uniforme com ângulo α

Função do potencial:

```
def compute_phi_uniform_flow(X, Y, V=0.0, alpha=0.0):
  return V * (X * np.cos(alpha) + Y * np.sin(alpha))
```

Definição:

```
phi_3 = phi_uniform_flow(X, Y, V=1.0,
     alpha=np.pi/6)
U = calculate_U_velocity(X, Y, phi_3)
V = calculate_V_velocity(Y, Y, phi_3)

U[:,:2]=np.nan; U[:,-2:]=np.nan
V[:2,:]=np.nan; V[-2:,:]=np.nan
```

Visualização:

```
cont = plt.contour(X, Y, phi_3,
     levels=50, cmap='jet')
plt.quiver(X, Y, U, V)
plt.colorbar(cont, label='phi [m2/s]'
     )
plt.xlabel('x [m]')
plt.ylabel('y [m]')
plt.axis(équal')
plt.grid(True)
plt.show()
```

Exemplo: Escoamento ao redor de um cilindro

Fonte localizada em $x = -a$:

$$\phi_1(x, y) = \frac{Q/L}{2\pi} \ln \sqrt{(-2a)^2 + (y - b)^2}$$

Sumidouro localizado em $x = a$:

$$\phi_2(x, y) = \frac{Q/L}{2\pi} \ln \sqrt{(y - b)^2}$$

Escoamento Uniforme Horizontal ($\alpha = 0$):

$$\phi_3(x, y) = Ux$$

All Together Now!

```
phi_1 = phi_line_source(X, Y, K= U_inf * (a**2), a=-a, b=0.0)
phi_2 = phi_line_source(X, Y, K=-U_inf * (a**2), a=a, b=0.0)
phi_3 = phi_uniform_flow(X, Y, U_inf, alpha)

phi = phi_1 + phi_2 + phi_3
```

Visualização com `streamplot` e máscara circular

Aplicamos uma máscara circular de raio *a* para simular, por exemplo, um obstáculo como um cilindro. Em seguida, visualizamos o campo com `streamplot`.

Cálculo do campo e aplicação da máscara:

```
phi = phi_1 + phi_2 + phi_3
U = calculate_U_velocity(X, Y, phi)
V = calculate_V_velocity(Y, Y, phi)

R = np.sqrt(X**2 + Y**2)
phi[R < a] = np.nan
U[R < a] = np.nan;  V[R < a] = np.
    nan

U[:, :2] = np.nan; U[:, -2:] = np.
    nan
V[:2, :] = np.nan; V[-2:, :] = np.
    nan
```

Visualização com `streamplot`:

```
MAG_VEL = np.sqrt(U**2 + V**2)

plt.streamplot(
X, Y, U, V,
color=MAG_VEL,
linewidth=1,
cmap='viridis'
)
plt.xlabel('x [m]')
plt.ylabel('y [m]')
plt.axis(équal')
plt.grid(True)
plt.show()
```

Interpretação física: escoamento uniforme sobre um cilindro

- O escoamento gerado combina:
  - Um **escoamento uniforme** com direção definida por um ângulo α;
  - Uma **fonte** e um **sumidouro** simétricos — formando um dipolo;
  - Uma **máscara circular** aplicada ao campo resultante — simulando a presença de um obstáculo (cilindro).
- Esse tipo de construção representa uma aproximação clássica do:

Escoamento potencial ao redor de um cilindro circular

- As linhas de corrente contornam a região onde o potencial foi anulado (a área sólida), formando padrões característicos de separação e simetria.

Exercício para Casa — Escoamento Uniforme sobre um Cilindro

Até agora utilizamos a **função potencial** ϕ para modelar e visualizar escoamentos.

Para este exercício, você deverá modificar o código fornecido para trabalhar com a **função corrente** ψ.

Tarefas:

1. Implementar as expressões de $\psi(x, y)$ para:
   - Fonte e sumidouro;
   - Escoamento uniforme com ângulo α.
2. Plotar o campo da função corrente $\psi(x, y)$ para o escoamento uniforme sobre um cilindro.
3. Calcular o campo de pressão com base na velocidade:

$$p = \frac{1}{2}\rho U^2 + p_{\text{ref}}$$

4. Considerando $\psi = 0$ sobre o cilindro, plotar os campos de velocidade e pressão ao redor do cilindro.
5. **Extra (opcional):** plotar a distribuição de pressão na superfície do cilindro em função do ângulo θ.