Introdução à Otimização Numérica com SciPy: Métodos univariados, multivariados e com restrições

EMC410235 - Programação Científica para Engenharia e Ciência Térmicas

Prof. Rafael F. L. de Cerqueira

2025.2

Objetivos da Aula

- Apresentar os fundamentos da otimização numérica.
- Discutir a classificação dos métodos:
  - Univariados vs. multivariados
  - Com e sem restrições
  - Baseados em gradiente vs. "gradient-free"
- Deduzir os métodos de Steepest Descent e Newton para problemas sem restrições.
- Comentar brevemente métodos modernos como CG, BFGS, Newton-CG.
- Mostrar exemplos práticos e aplicações em engenharia.

O que é Otimização?

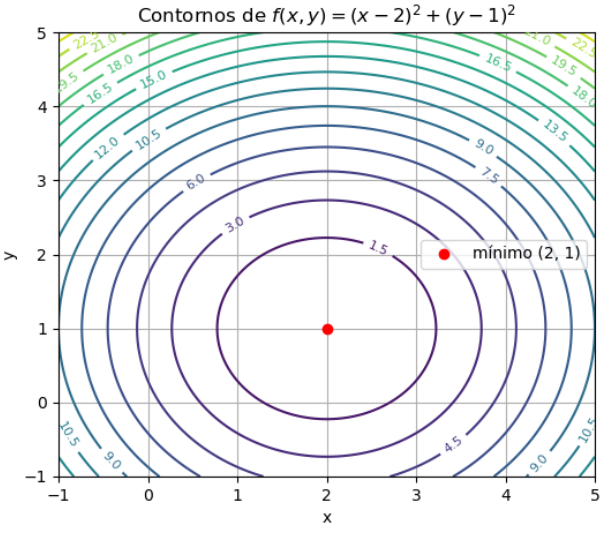
Definição

Otimização consiste em encontrar o ponto $x^* \in \mathbb{R}^n$ que minimiza (ou maximiza) uma função $f(x)$, possivelmente sujeita a restrições.

- $\min f(x)$ ou $\max f(x)$
- Sujeito a:
  - Restrições de igualdade: $h_i(x) = 0$
  - Restrições de desigualdade: $g_j(x) \leq 0$
- Aplicações em engenharia, economia, ciência de dados, etc.

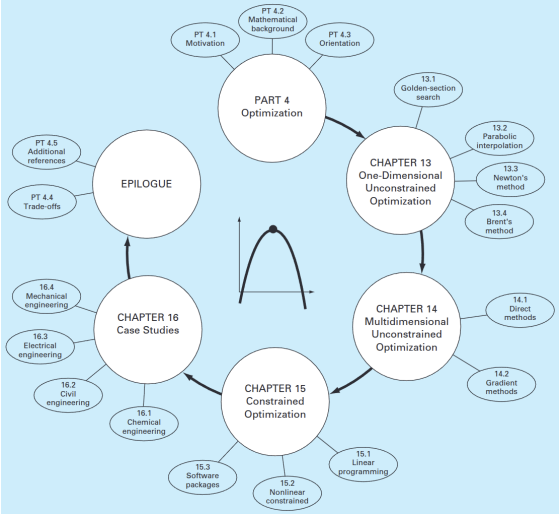
Exemplo Visual: Mínimo de uma Função Quadrática

- Objetivo: encontrar o ponto de menor valor de $f(x, y)$
- Sem restrições $\rightarrow$ busca livre no plano
- Solução analítica: $x = 2, y = 1$



Classificação Geral dos Métodos de Otimização

- **Quanto ao tipo de problema:**
  - **Sem restrições** (unconstrained)
  - **Com restrições** (constrained)
- **Quanto ao número de variáveis:**
  - **Univariados**: $x \in \mathbb{R}$
  - **Multivariados**: $x \in \mathbb{R}^n$
- **Quanto ao uso de derivadas:**
  - **Gradient-based** (usam ∇f)
  - **Gradient-free** (não usam derivadas)



Numerical Methods for Engineers. Steven C. Chapra & Raymond P. Canale. 7ed

Otimização Univariada Gradient-Free

Problema

Minimizar $f(x)$ com $x \in \mathbb{R}$, sem utilizar derivadas.

- Utilizam apenas avaliações de $f(x)$
- Buscam o mínimo em um intervalo $[a, b]$
- Assumem que $f(x)$ é contínua e unimodal no intervalo

Exemplos clássicos:

- Busca Exaustiva (Grid Search)
- Seção Áurea

Busca Exaustiva (Grid Search)

Ideia

Avaliar a função $f(x)$ em pontos igualmente espaçados no intervalo $[a, b]$ e escolher o que apresentar o menor valor.

- Método direto e simples, porém ineficiente
- Útil para ter uma ideia inicial do comportamento de $f(x)$
- Requer definir um número de subdivisões n

Pseudocódigo:

1. Dividir o intervalo $[a, b]$ em n pontos
2. Avaliar $f(x_i)$ para cada ponto
3. Retornar $x^* = \arg\min f(x_i)$

Motivação: Otimização de Função Univariada

Função Unimodal

Uma função $f(x)$ é unimodal em um intervalo $[a, b]$ se possui apenas um mínimo (ou máximo) local nesse intervalo.

- Avaliar $f(x)$ em dois pontos simétricos ao redor do centro
- Estratégia lembra a bisseção, mas para localizar extremos
- Essa ideia fundamenta o método da **Seção Áurea**

Método da Seção Áurea: Dedução da Razão

- Seja um intervalo dividido em duas partes: ℓ_1 e ℓ_2
- Queremos que:

$$\ell = \ell_1 + \ell_2$$
$$\frac{\ell_1}{\ell} = \frac{\ell_2}{\ell_1}$$

- Substituindo:

$$\frac{\ell_1}{\ell_1 + \ell_2} = \frac{\ell_2}{\ell_1}$$

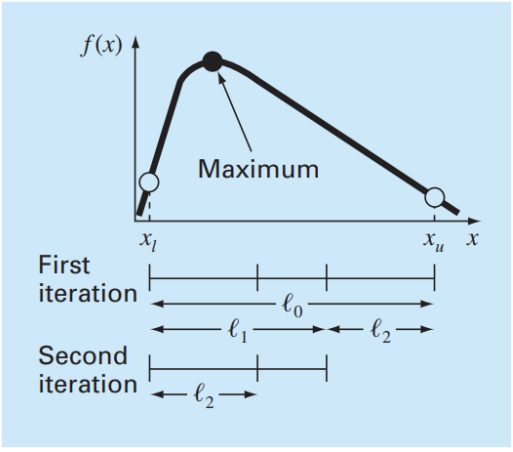
- Seja $R = \frac{\ell_2}{\ell_1} \Rightarrow 1 + R = \frac{1}{R}$

$$\Rightarrow R^2 + R - 1 = 0$$

- Solução positiva:

$$R = \frac{\sqrt{5} - 1}{2} \approx 0.618 \text{ (Razão Áurea ou } \textit{Golden Ratio}\text{)}$$

- Fibonacci: 0, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...



Método da Seção Áurea: Algoritmo

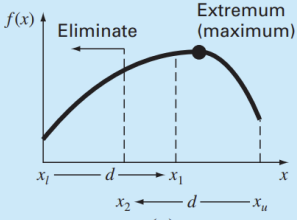
- Objetivo: minimizar $f(x)$ no intervalo $[x_l, x_u]$
- Razão áurea: $R = \frac{\sqrt{5}-1}{2} \approx 0.61803$
- Cálculo da distância:

$$d = R \cdot (x_u - x_l)$$

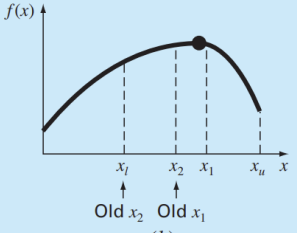
- Pontos internos:

$$x_1 = x_l + d, \quad x_2 = x_u - d$$

- Avalie $f(x_1)$ e $f(x_2)$:
  - Se $f(x_1) > f(x_2)$: elimine $[x_l, x_2]$, novo $x_l = x_2$
  - Senão: elimine $[x_1, x_u]$, novo $x_u = x_1$
- Novo ponto interno reutiliza valor anterior, evitando reavaliação



(a)



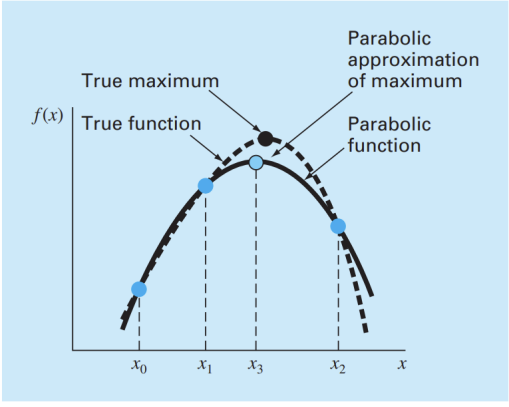
(b)

Interpolação Parabólica: Ideia Geral

- Uma parábola (polinômio de 2ª ordem) costuma aproximar bem a forma de $f(x)$ próximo de um ótimo
- Três pontos que cercam um extremo permitem ajustar uma parábola única
- O novo ponto x_3 é estimado como o vértice da parábola:

$$x_3 = \frac{f(x_0)(x_1^2 - x_2^2) + f(x_1)(x_2^2 - x_0^2) + f(x_2)(x_0^2 - x_1^2)}{2[f(x_0)(x_1 - x_2) + f(x_1)(x_2 - x_0) + f(x_2)(x_0 - x_1)]}$$

- Estratégias de atualização:
  - Atualização sequencial ($x_0 \leftarrow x_1$, $x_1 \leftarrow x_2$, $x_2 \leftarrow x_3$)
  - Atualização com bracketing (semelhante à seção áurea)
- Pode convergir lentamente se "prender" em uma extremidade (como no método da posição falsa)



Método de Brent (Univariado)

Ideia geral

Método híbrido desenvolvido por Brent para minimização unidimensional. Combina:

- **Seção Áurea** $\rightarrow$ confiável, porém lenta
- **Interpolação parabólica** $\rightarrow$ rápida, porém pode falhar

- Primeiro tenta interpolação parabólica
- Se o resultado for aceitável, continua com ela
- Se falhar (instável ou fora do intervalo), recorre à seção áurea
- Estratégia adaptativa: alterna entre precisão e velocidade
- Muito utilizado em bibliotecas numéricas (ex: ‘scipy.optimize.minimize_scalar‘)

Métodos Univariados Gradient-Based: Visão Geral

- Utilizam informações da derivada de $f(x)$ (ou aproximações) para guiar a busca
- Em geral, convergem mais rapidamente que métodos gradient-free
- Requerem que $f(x)$ seja diferenciável (pelo menos localmente)

Métodos Clássicos:

- **Método de Newton:** usa $f'(x)$ e $f''(x)$, converge rapidamente perto do mínimo

Método de Newton (Univariado)

Objetivo

Encontrar o ponto x^* tal que $f'(x^*) = 0$, assumindo que $f''(x) \neq 0$

- Baseado na expansão de Taylor de $f'(x)$:

$$f'(x) \approx f'(x_k) + f''(x_k)(x - x_k)$$

- Impondo $f'(x_{k+1}) = 0$, obtemos:

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$$

- Interpretação geométrica: interseção do eixo x com a tangente à curva $f'(x)$
- Convergência quadrática (se bem condicionado e próximo do mínimo)

Otimização Multivariada Gradient-Free

Características

- Não utilizam gradientes (derivadas) de $f(\vec{x})$
- Buscam soluções apenas com base em avaliações de $f(\vec{x})$
- Úteis quando:
  - $f(\vec{x})$ não é diferenciável ou é ruidosa
  - Avaliações de $f(\vec{x})$ vêm de simulações ou experimentos
  - Há múltiplos mínimos locais ou a forma da função é desconhecida

- **Métodos determinísticos:**
  - Nelder-Mead (simplex adaptativo)
  - Powell
  - COBYLA (com restrições)

- **Meta-heurísticas estocásticas (Busca Global):**
  - Algoritmos Genéticos (GA)
  - Differential Evolution (DE)
  - Particle Swarm Optimization (PSO)
  - Simulated Annealing

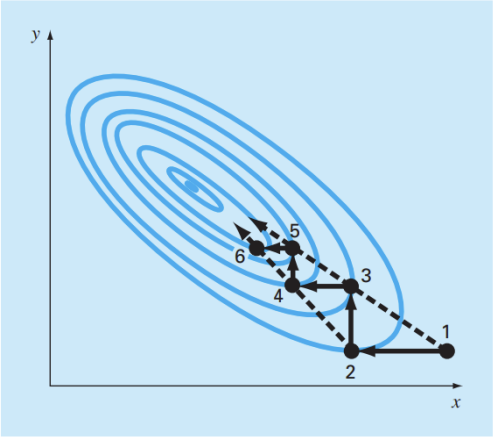
Nota: esses métodos são úteis em problemas não convexos, com ruído ou onde o cálculo do gradiente é inviável.

Busca Univariada em Problemas Multivariados

Motivação

Estratégia simples e eficiente de otimização sem derivadas: variar uma variável de cada vez, mantendo as outras constantes.

- Reduz problema multivariado a sequência de buscas unidimensionais
- Cada etapa usa métodos como seção áurea ou Brent
- Permite progressão por etapas alternadas entre as variáveis



Método de Nelder-Mead (Simplex Adaptativo)

Ideia

O método usa um **simplex** (um polígono com $n+1$ vértices em $\mathbb{R}^n$) que se adapta à paisagem da função a cada iteração.

- Avalia a função nos vértices do simplex
- A cada passo, modifica o simplex com operações:
  - Reflexão
  - Expansão
  - Contração
  - Redução
- Não requer gradientes
- Funciona bem para problemas de baixa dimensão ($n \leq 5$)
- Pode falhar em problemas mal-condicionados ou com muitos mínimos locais
- Exemplo do processo de busca em `https://www.youtube.com/watch?v=j2gcuRVbwR0`

Método de Powell

Ideia

Executa minimizações unidimensionais sucessivas ao longo de direções atualizadas a cada ciclo. Não utiliza derivadas.

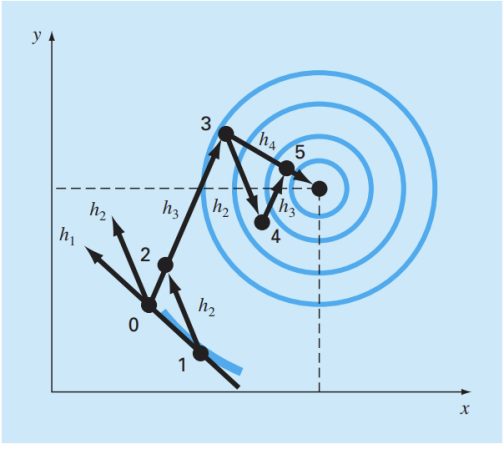
- Inicia com direções coordenadas (eixos)
- Minimiza $f(\vec{x})$ ao longo de cada direção
- Atualiza as direções com base no progresso global
- Razoavelmente eficiente para problemas suaves
- Pode falhar se a função for mal-condicionada ou com múltiplos mínimos

Método de Powell: Intuição Geométrica

Resumo da ideia

Powell propôs um método que busca o ótimo sem derivadas, movendo-se iterativamente ao longo de direções otimizadas e combinadas geometricamente.

- Início em ponto 0, com duas direções iniciais $\vec{h}_1$ e $\vec{h}_2$
- Movimentos:
  1. Ao longo de $\vec{h}_1$: ponto 1
  2. Ao longo de $\vec{h}_2$: ponto 2
  3. Nova direção $\vec{h}_3 = \vec{x}_2 - \vec{x}_0$, leva ao ponto 3
  4. De 3, move em $\vec{h}_2 \rightarrow$ ponto 4, depois em $\vec{h}_3 \rightarrow$ ponto 5
  5. De 3 a 5 define nova direção $\vec{h}_4$, conjugada com $\vec{h}_3$
- Com direções conjugadas, o método converge ao ótimo
- Convergência quadrática, sem derivadas



O Gradiente em Otimização Multivariada

O que é o gradiente?

É o vetor das derivadas parciais da função objetivo:

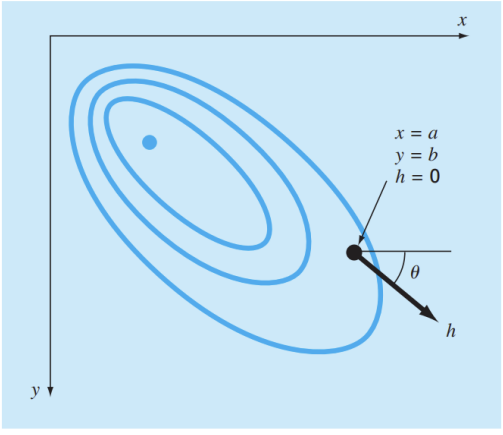
$$\vec{\nabla} f(\vec{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

- Representa a direção de **maior crescimento local** da função
- Direção de **descida mais inclinada** é $-\vec{\nabla} f$
- O módulo do gradiente indica a intensidade da variação local
- Em um ponto $\vec{x}^*$, se $\vec{\nabla} f(\vec{x}^*) = \vec{0}$, pode haver um ótimo
- O vetor gradiente é ortogonal às curvas de nível de $f(\vec{x})$

Interpretação Geométrica do Gradiente

- Para $f(x, y)$, o gradiente $\vec{\nabla} f(x, y)$ aponta a direção de maior subida
- O gradiente em um ponto também é a direção de maior variação da função
- O gradiente é perpendicular às curvas de nível
- Exemplo:

$$f(x, y) = xy^2, \quad \vec{\nabla} f(2, 2) = \langle 4, 8 \rangle$$



Método da Descida Mais Inclinada (*Steepest Descent*)

Ideia

Mover-se, a cada iteração, na direção oposta ao gradiente:

$$\vec{p}_k = -\vec{\nabla} f(\vec{x}_k)$$

- O gradiente aponta a direção de maior crescimento de f
- A descida mais rápida ocorre na direção $-\vec{\nabla} f$
- A nova posição é dada por:

$$\vec{x}_{k+1} = \vec{x}_k + \alpha_k \vec{p}_k$$

- α_k é o passo ideal — obtido por minimização univariada:

$$\alpha_k = \arg\min_{\alpha} f(\vec{x}_k + \alpha \vec{p}_k)$$

- Repetir até convergência: $\|\vec{\nabla} f(\vec{x}_k)\| < \varepsilon$

Motivação: Por que seguir o gradiente negativo?

- Dada uma função $f(\vec{x})$ continuamente diferenciável, queremos minimizar f
- Em $\mathbb{R}^n$, o gradiente $\vec{\nabla} f(\vec{x})$ aponta a direção de maior crescimento local
- Logo, a direção de descida mais acentuada é:

$$\vec{p} = -\vec{\nabla} f(\vec{x})$$

- Essa é a direção de maior decréscimo da função no ponto $\vec{x}$
- O método de Steepest Descent itera seguindo essa direção, com um passo α_k escolhido para minimizar f ao longo da reta

Steepest Descent: Atualização com Busca Linear

- Definida a direção $\vec{p}_k = -\vec{\nabla} f(\vec{x}_k)$, procuramos o passo α_k que minimiza f ao longo da reta:

$$\alpha_k = \arg\min_{\alpha > 0} f(\vec{x}_k + \alpha \vec{p}_k)$$

- Isso pode ser feito por métodos univariados (ex: Brent)
- A nova iteração é:

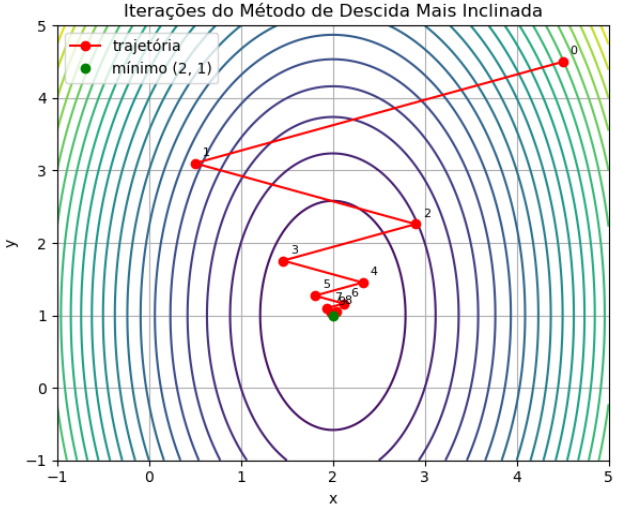
$$\vec{x}_{k+1} = \vec{x}_k + \alpha_k \vec{p}_k$$

- Iteramos até que:

$$\|\vec{\nabla} f(\vec{x}_{k+1})\| < \varepsilon$$

Steepest Descent: Interpretação Geométrica

- O gradiente é ortogonal às curvas de nível
- A trajetória segue a direção de descida, mas pode oscilar se os contornos forem elípticos
- O método pode apresentar "zigue-zague" e convergência lenta em funções mal condicionadas



Método do Gradiente Conjugado (*Conjugate Gradient - CG*)

Motivação

O método da descida mais inclinada (Steepest Descent) pode ser ineficiente para funções com contornos alongados — ele "ziguezagueia".

- O método do **Gradiente Conjugado (CG)** melhora isso:
  - Em vez de sempre usar $-\vec{\nabla} f$, ele usa direções **conjugadas**
  - Cada nova direção "corrige"a anterior, evitando repetir movimentos
- Para funções quadráticas com Hessiana simétrica definida positiva:
  - Converge em no máximo n passos ($n =$ número de variáveis)
  - Dispensa cálculo explícito da Hessiana
- Muito usado para resolver grandes sistemas lineares $A\vec{x} = \vec{b}$

Disponível em `scipy.optimize.minimize(method='CG')`

Método de Newton Multivariado: Dedução

Expansão de Taylor de 2ª ordem em $\mathbb{R}^n$

$$f(\vec{x}_k + \vec{p}) \approx f(\vec{x}_k) + \vec{\nabla} f(\vec{x}_k)^T \vec{p} + \frac{1}{2} \vec{p}^T H_k \vec{p}$$

- $H_k = \nabla^2 f(\vec{x}_k)$: matriz Hessiana (simétrica)
- Objetivo: encontrar $\vec{p}$ que minimiza a aproximação quadrática
- Derivando em relação a $\vec{p}$ e igualando a zero:

$$\vec{\nabla} f(\vec{x}_k) + H_k \vec{p} = 0 \Rightarrow \vec{p} = -H_k^{-1} \vec{\nabla} f(\vec{x}_k)$$

Atualização e Interpretação

Atualização do ponto

$$\vec{x}_{k+1} = \vec{x}_k - H_k^{-1}\vec{\nabla} f(\vec{x}_k)$$

- Leva em conta a curvatura da função por meio da Hessiana
- Convergência quadrática (rápida) perto do ótimo
- Exige o cálculo (ou aproximação) de H_k
- Pode ser adaptado para evitar inversão direta:

Nota: Quando H_k não está disponível, usa-se BFGS ou Newton-CG

Vantagens e Limitações do Método de Newton

Vantagens

- Convergência quadrática (mais rápida que Steepest Descent)
- Direção de descida com curvatura (uso da Hessiana)

Limitações

- Requer cálculo da Hessiana (pode ser caro ou inviável)
- Pode falhar se H_k não for definida positiva
- Necessita resolver sistema linear em cada iteração

Newton-CG e BFGS: Métodos Avançados Gradient-Based

Newton-CG (ou Truncated Newton)

- Variante do método de Newton que **não inverte explicitamente a Hessiana**
- Usa método **Conjugate Gradient** internamente para resolver $H\vec{p} = -\vec{\nabla} f$
- Permite aproveitar $H\vec{v}$ via produto vetorial (sem formar H completo)
- Bom para problemas de grande porte com Hessiana esparsa

BFGS (quasi-Newton)

- Método de otimização que **aproxima a Hessiana** iterativamente
- Atualiza uma matriz B_k que imita H^{-1}, com base em gradientes
- Mais estável que Newton quando H não é facilmente disponível
- Versão limitada para grandes problemas: `L-BFGS-B`

Ambos disponíveis em `scipy.optimize.minimize` via os métodos `'Newton-CG'`, `'BFGS'` e `'L-BFGS-B'`

Panorama dos Métodos em `scipy.optimize.minimize`

Método	Gradiente	Hessiana	Bounds	Restrições
`Nelder-Mead`	×	×	✓	×
`Powell`	×	×	✓	×
`COBYLA`	×	×	✓	✓ (ineq.)
`BFGS`	✓	×	×	×
`L-BFGS-B`	✓	×	✓	×
`CG`	✓	×	×	×
`Newton-CG`	✓	opcional	×	×
`trust-constr`	✓	✓ ou approx.	✓	✓ (eq./ineq.)
`TNC`	✓	×	✓	×
`SLSQP`	✓	×	✓	✓ (eq./ineq.)
`dogleg`	✓	✓	×	×
`trust-ncg`	✓	✓ ou Hess-prod.	×	×

Legenda:

Fonte: SciPy Documentation (v1.11+)

Problemas com restrições

Queremos minimizar $f(\vec{x})$, sujeito a:

- Restrições de desigualdade: $g_i(\vec{x}) \leq 0$
- Restrições de igualdade: $h_j(\vec{x}) = 0$
- Possivelmente, limites simples (*bounds*) em cada variável

Exemplo:

$$\begin{aligned} \text{minimize} \quad & f(x,y) = (x-2)^2 + (y-3)^2 \\ \text{subject to} \quad & x + y \leq 4 \\ & x \geq 0, \quad y \geq 0 \end{aligned}$$

Tipos de Restrições

1. Desigualdade

$$g(\vec{x}) \leq 0$$

Exemplo: $x^2 + y^2 - 1 \leq 0$ (interior de um círculo)

2. Igualdade

$$h(\vec{x}) = 0$$

Exemplo: $x + y - 1 = 0$ (linha reta)

Importante: problemas com restrições são mais difíceis numericamente. Métodos diferentes tratam as restrições de formas distintas (e.g., penalização, multiplicadores de Lagrange, projeção).

Métodos com Suporte a Restrições

Método	Desigualdade (`ineq`)	Igualdade (`eq`)
`SLSQP`	Sim	Sim
`trust-constr`	Sim	Sim
`COBYLA`	Sim	Não
`Powell, Nelder-Mead, L-BFGS-B`	Não	Não

Recomendações

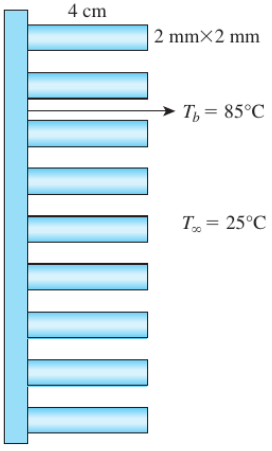
- `SLSQP`: rápido e confiável para problemas suaves
- `trust-constr`: mais robusto, permite Jacobiano/Hessiana
- `COBYLA`: útil quando não há derivadas disponíveis

Projeto de Superfície Aletada para Placa de Circuito

Uma placa de circuito impresso de 15 cm $\times$ 20 cm deve ser refrigerada com aletas de alumínio ($k = 237\,\mathrm{W/m{\cdot}K}$), cada uma com comprimento de 4 cm e seção quadrada de 2 mm $\times$ 2 mm, fixadas em um dos lados da placa.

A temperatura ambiente é de 25 °C e o coeficiente de transferência de calor por convecção é $h = 20\,\mathrm{W/m^2 \cdot K}$. Para evitar superaquecimento da placa, sua superfície superior deve permanecer a no máximo 85 °C.

Deseja-se projetar uma superfície com aletas com efetividade global igual a 3, determinando o número apropriado de aletas que mantenha a placa dentro da faixa térmica desejada.



Projeto de Superfície Aletada para Placa de Circuito

Agora, para o mesmo problema anterior, deseja-se manter a **mesma temperatura da base** e a **mesma efetividade global** ($\varepsilon = 3$).

Entretanto, um estudo de otimização será conduzido para minimizar o número de aletas e a quantidade de material utilizado.

As variáveis de projeto são:

- **Espessura da aleta** $t \in [2{,}5\,\text{mm},\ 10\,\text{mm}]$
- **Comprimento da aleta** $L \in [5\,\text{mm},\ 50\,\text{mm}]$
- **Número máximo de aletas:** $N \leq 200$

O objetivo é encontrar os valores de t e L que resultem no menor número possível de aletas e menor massa total de alumínio, respeitando as restrições acima.