

# Otimização e modelagem

Marina Andretta

ICMC-USP

6 de agosto de 2018

Baseado nos livros Numerical Optimization, de J. Nocedal e S. J. Wright, e Pesquisa Operacional, de M. Arenales, V. Armentano, R. Morabito e H. Yanasse.

Otimizar significa encontrar a melhor maneira de fazer algo, dada uma medida do que é ser “melhor”.

Estamos sempre otimizando:

- quando fazemos compras, queremos minimizar o dinheiro gasto, ou maximizar a qualidade do que foi comprado;
- quando fazemos matrícula, queremos fazer o maior número possível de disciplinas, sem, no entanto, prejudicar nosso desempenho;
- quando organizamos as horas de estudo, queremos aprender o máximo possível, de preferência no menor tempo.

Uma maneira de resolver problemas de otimização é formulá-los matematicamente.

O processo de transformar um problema real em uma formulação matemática que o representa é chamado de **modelagem matemática**.

Na maioria das vezes, no processo de modelagem do problema, é necessário fazer simplificações, ou porque o problema não tem todos os dados conhecidos ou simplesmente para facilitar a resolução do modelo.

Matematicamente falando, otimizar significa maximizar ou minimizar uma função sujeita a restrições a suas variáveis.

Usaremos a seguinte notação:

- $x$  é um vetor de **variáveis**;
- $f$  é a **função objetivo**, a função de  $x$  que deve ser minimizada ou maximizada;
- $c$  é um vetor de **restrições** que o ponto  $x$  deve satisfazer. Este é um vetor de funções de  $x$ . O número de componentes de  $c$  é o número de restrições em  $x$ .

Assim, podemos escrever problemas de otimização da seguinte maneira:

$$\begin{array}{ll} \text{Minimizar} & f(x) \\ \text{sujeita a} & c_i(x) = 0, \quad i \in \mathcal{E} \\ & c_i(x) \geq 0, \quad i \in \mathcal{I} \end{array} \quad (1)$$

onde

- $x \in \mathbf{R}^n$ ,  $f \in \mathbf{R}^n \rightarrow \mathbf{R}$ ,  $c_i \in \mathbf{R}^n \rightarrow \mathbf{R}$ ,
- $\mathcal{E}$  e  $\mathcal{I}$  são conjuntos de índices.

## Exemplo de modelo

$$\begin{array}{ll} \text{Minimizar} & (x_1 - 2)^2 + (x_2 - 1)^2 \\ \text{sujeita a} & x_1^2 - x_2 \leq 0, \\ & x_1 + x_2 \leq 2. \end{array}$$

Este problema pode ser escrito da seguinte forma:

$$\begin{array}{ll} \text{Minimizar} & f(x) = (x_1 - 2)^2 + (x_2 - 1)^2, x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \\ \text{sujeita a} & c(x) = \begin{bmatrix} c_1(x) \\ c_2(x) \end{bmatrix} = \begin{bmatrix} -x_1^2 + x_2 \\ -x_1 - x_2 + 2 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \\ & \mathcal{I} = \{1, 2\}, \mathcal{E} = \emptyset. \end{array}$$

# Exemplo

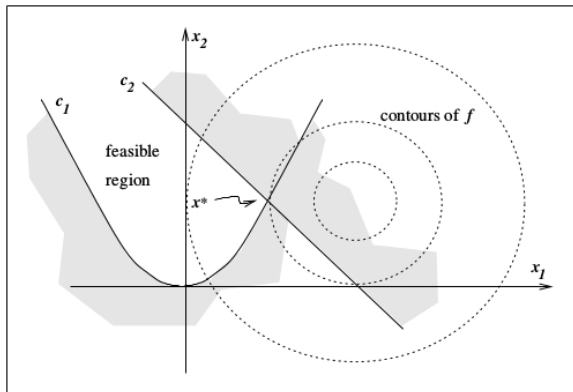


Figura: Exemplo de problema de otimização (Figura 1.1 de Numerical Optimization, de J. Nocedal e S. J. Wright)

Note que um problema de **maximizar**  $f$  pode ser substituído por um problema de **minimizar**  $-f$ .

Problemas de otimização em geral podem ser escritos usando a formulação (1), que chamaremos de **formulação padrão**.



## Exemplo: problema de transporte

Uma empresa que produz açúcar tem uma fábrica em São Carlos e outra em Araraquara ( $F_1$  e  $F_2$ ) e 3 clientes espalhados pelo estado de São Paulo, que chamaremos de  $C_1$ ,  $C_2$  e  $C_3$ .

A fábrica de São Carlos produz ( $p_1$ ) 50 toneladas de açúcar por semana, enquanto que a fábrica de Araraquara produz ( $p_2$ ) 100 toneladas.

Cada cliente possui uma demanda  $d_j$  (em toneladas, por semana) por açúcar, dada pela tabela abaixo:

$C_1$	$C_2$	$C_3$
20	60	40

## Exemplo: problema de transporte

O custo  $c_{ij}$ , em reais, de enviar uma tonelada de produto de cada fábrica  $i$  para cada cliente  $j$  é dado pela tabela abaixo:

	$C_1$	$C_2$	$C_3$
$F_1$	35	20	40
$F_2$	90	55	77

O problema é determinar quanto açúcar enviar, em uma semana, de cada fábrica para cada cliente de modo a satisfazer todas as restrições e minimizar o custo.

## Exemplo: problema de transporte

Para modelar este problema matematicamente, vamos definir **variáveis**  $x_{ij}$  que terão como valor a quantidade de toneladas de açúcar enviadas, em uma semana, da fábrica  $F_i$  para um cliente  $C_j$ .

## Exemplo: problema de transporte

Para modelar este problema matematicamente, vamos definir **variáveis**  $x_{ij}$  que terão como valor a quantidade de toneladas de açúcar enviadas, em uma semana, da fábrica  $F_i$  para um cliente  $C_j$ .

Com as variáveis definidas, podemos definir nossa função objetivo. Esta é uma função de  $\mathbf{R}^6$  em  $\mathbf{R}$  que, dados os valores das variáveis  $x_{ij}$ , devolve o custo.

## Exemplo: problema de transporte

Para modelar este problema matematicamente, vamos definir **variáveis**  $x_{ij}$  que terão como valor a quantidade de toneladas de açúcar enviadas, em uma semana, da fábrica  $F_i$  para um cliente  $C_j$ .

Com as variáveis definidas, podemos definir nossa função objetivo. Esta é uma função de  $\mathbf{R}^6$  em  $\mathbf{R}$  que, dados os valores das variáveis  $x_{ij}$ , devolve o custo.

Usando os custos de transporte de cada fábrica para cada cliente, podemos definir a função objetivo como

$$f(x) = 35x_{11} + 20x_{12} + 40x_{13} + 90x_{21} + 55x_{22} + 77x_{23}.$$

## Exemplo: problema de transporte

Para modelar este problema matematicamente, vamos definir **variáveis**  $x_{ij}$  que terão como valor a quantidade de toneladas de açúcar enviadas, em uma semana, da fábrica  $F_i$  para um cliente  $C_j$ .

Com as variáveis definidas, podemos definir nossa função objetivo. Esta é uma função de  $\mathbf{R}^6$  em  $\mathbf{R}$  que, dados os valores das variáveis  $x_{ij}$ , devolve o custo.

Usando os custos de transporte de cada fábrica para cada cliente, podemos definir a função objetivo como

$$f(x) = 35x_{11} + 20x_{12} + 40x_{13} + 90x_{21} + 55x_{22} + 77x_{23}.$$

No caso deste problema, desejamos minimizar a função objetivo.

# Exemplo: problema de transporte

Agora precisamos definir quais são as restrições do nosso problema.

Teremos 3 grupos de restrições:

- 1 As restrições do primeiro grupo servirão para garantir que uma fábrica não envia mais produtos do que é capaz de produzir.
- 2 As restrições do segundo grupo irão garantir que as demandas de cada cliente serão atendidas.
- 3 As restrições do terceiro grupo irão garantir que a quantidade de produto enviada de uma fábrica a um cliente nunca é negativa.

## Exemplo: problema de transporte

Para garantir que fábrica  $F_1$  não envia mais produto do que é capaz de produzir (50 toneladas), temos a restrição

$$x_{11} + x_{12} + x_{13} \leq 50.$$



## Exemplo: problema de transporte

Para garantir que fábrica  $F_1$  não envia mais produto do que é capaz de produzir (50 toneladas), temos a restrição

$$x_{11} + x_{12} + x_{13} \leq 50.$$

Analogamente, para garantir que fábrica  $F_2$  não envia mais produto do que é capaz de produzir (100 toneladas), temos a restrição

$$x_{21} + x_{22} + x_{23} \leq 100.$$

## Exemplo: problema de transporte

Para garantir que o cliente  $C_1$  receba a quantidade de produto desejada (20 toneladas), temos a restrição

$$x_{11} + x_{21} = 20.$$

## Exemplo: problema de transporte

Para garantir que o cliente  $C_1$  receba a quantidade de produto desejada (20 toneladas), temos a restrição

$$x_{11} + x_{21} = 20.$$

Analogamente, para garantir que os clientes  $C_2$  e  $C_3$  recebam as quantidades de produtos desejadas (60 e 40 toneladas, respectivamente), temos as restrições

$$x_{12} + x_{22} = 60,$$

$$x_{13} + x_{23} = 40.$$

## Exemplo: problema de transporte

Por fim, para garantir que as quantidades de produto enviadas de cada fábrica  $F_i$  para cada cliente  $C_j$  não seja menor que zero, temos as restrições

$$x_{11} \geq 0,$$

$$x_{12} \geq 0,$$

$$x_{13} \geq 0,$$

$$x_{21} \geq 0,$$

$$x_{22} \geq 0,$$

$$x_{23} \geq 0.$$

## Exemplo: problema de transporte

Então, nosso modelo para este problema de transporte fica:

**minimizar**  $35x_{11} + 20x_{12} + 40x_{13} + 90x_{21} + 55x_{22} + 77x_{23}$

**sujeita a**

$$\begin{aligned}x_{11} + x_{12} + x_{13} &\leq 50, \\x_{21} + x_{22} + x_{23} &\leq 100, \\x_{11} + x_{21} &= 20, \\x_{12} + x_{22} &= 60, \\x_{13} + x_{23} &= 40, \\x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23} &\geq 0.\end{aligned}$$

# Exemplo: problema de transporte

Usando nossa formulação padrão, temos

**minimizar**  $35x_{11} + 20x_{12} + 40x_{13} + 90x_{21} + 55x_{22} + 77x_{23}$

**sujeita a**

$$\begin{aligned}c_1(x) &= -x_{11} - x_{12} - x_{13} + 50 \geq 0, \\c_2(x) &= -x_{21} - x_{22} - x_{23} + 100 \geq 0, \\c_3(x) &= x_{11} + x_{21} - 20 = 0, \\c_4(x) &= x_{12} + x_{22} - 60 = 0, \\c_5(x) &= x_{13} + x_{23} - 40 = 0, \\c_6(x) &= x_{11} \geq 0, \\c_7(x) &= x_{12} \geq 0, \\c_8(x) &= x_{13} \geq 0, \\c_9(x) &= x_{21} \geq 0, \\c_{10}(x) &= x_{22} \geq 0, \\c_{11}(x) &= x_{23} \geq 0,\end{aligned}$$

# Exemplo: problema de transporte

com

$$\bullet x = \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{21} \\ x_{22} \\ x_{23} \end{bmatrix},$$

$$\bullet \mathcal{I} = \{1, 2, 6, 7, 8, 9, 10, 11\} \text{ e}$$

$$\bullet \mathcal{E} = \{3, 4, 5\}.$$

## Exemplo: problema de transporte

Para este modelo, a solução é

$$x_{11} = 20, \quad x_{12} = 0, \quad x_{13} = 30,$$

$$x_{21} = 0, \quad x_{22} = 60, \quad x_{23} = 10.$$

O custo total para enviar todas as toneladas necessárias de açúcar para os clientes será de 5970 reais.



# Problema *versus* instância

Note que, no exemplo dado, temos definidos

- a quantidade  $n$  de fábricas  $F_i$ ;
- a quantidade  $m$  de clientes  $C_j$ ;
- os valores de  $p_i$  (produção de açúcar de cada fábrica  $F_i$ );
- os valores de  $d_j$  (demanda pelo produto de cada cliente  $C_j$ );
- os valores de  $c_{ij}$  (custo de transportar o produto da fábrica  $F_i$  para o cliente  $C_j$ ).

Trocando os valores para estes parâmetros, temos **instâncias** diferentes para o **problema** de transporte.

# Problema *versus* instância

Para o caso geral, podemos escrever o modelo para o problema de transporte da seguinte forma:

$$\text{minimizar } \sum_{i=1}^n \sum_{j=1}^m c_{ij} x_{ij}$$

$$\text{sujeita a } \sum_{j=1}^m x_{ij} \leq p_i, \quad \text{para } i = 1, \dots, n,$$

$$\sum_{i=1}^n x_{ij} = d_j, \quad \text{para } j = 1, \dots, m,$$

$$x_{ij} \geq 0, \quad \text{para } i = 1, \dots, n \text{ e } j = 1, \dots, m.$$

Em alguns casos, o valor das variáveis pode somente assumir valores inteiros.

Suponha que a empresa do exemplo anterior fabricasse pratos, no lugar de produtos químicos. Neste caso, a quantidade de produto a ser enviada das fábricas para os clientes ( $x_{ij}$ ) deveria ser inteira. Ou seja, deveria haver uma restrição do tipo  $x_{ij} \in Z$ , para todo  $i$  e  $j$ . Este problema seria um problema de **programação inteira**.

# Otimização contínua *versus* discreta

O termo genérico **otimização discreta** se refere a problemas cujas variáveis assumem valores em um conjunto finito. Em contraste, o termo **otimização contínua** se refere a problemas cujas variáveis podem assumir valores em conjuntos infinitos não-enumeráveis, tipicamente, um conjunto de valores reais.

Problemas de **otimização contínua** normalmente são mais fáceis de resolver do que problemas de **otimização discreta**. A suavidade da função objetivo e das funções das restrições permite que, a partir da informação em um ponto  $x$ , possamos deduzir o comportamento destas funções em uma vizinhança de  $x$ . Métodos para resolver problemas de **otimização contínua** fazem uso destas propriedades quando buscam a solução do problema.

No caso de problemas de **otimização discreta**, uma solução “vizinha” de outra pode não apresentar valores da função objetivo próximos. Neste caso, uma alternativa seria enumerar as possíveis soluções, buscando a de menor valor de função objetivo. No entanto, para problemas maiores, o tempo gasto com essa busca é proibitivo.

Em alguns casos, algumas variáveis do problema são contínuas e outras são discretas. Estes problemas são chamados de problemas de **programação inteira mista**.

Problemas do tipo (1) podem ser classificados de várias maneiras:

- Quanto à **natureza da função objetivo**: linear, não-linear, convexa, etc;
- Quanto ao **número de variáveis**: pequeno, médio ou grande;
- Quanto à **suavidade das funções**: diferenciável ou não-diferenciável;
- Etc.

Uma distinção importante entre problemas do tipo (1) é com relação às restrições. Se não há restrições, dizemos que o problema é **irrestrito**. Se há pelo menos uma restrição, dizemos que o problema é **restrito**.

**Problemas irrestritos** surgem de modelos de problemas que naturalmente não possuem restrições. Em alguns casos, o problema teria restrições, mas elas podem ser ignoradas. Em outros casos, problemas originalmente restritos têm as restrições transformadas em um termo de penalidade na função objetivo, tornando-se irrestritos.

# Otimização irrestrita *versus* restrita

**Problemas restritos** surgem de modelos que possuem restrições explícitas. As restrições podem ser:

- Simples limitantes nas variáveis, como  $0 \leq x \leq 10$ . Estas restrições são chamadas de **restrições de caixa**;
- **Restrições lineares**, como  $\sum_i x_i \leq 1$ ;
- **Restrições não-lineares**, como  $\sin(x) + \cos(x) = 1$ .

Problemas nos quais tanto a função objetivo como as restrições são lineares em  $x$ , são chamados de **problemas de programação linear**. Se a função objetivo ou alguma das restrições não é linear, o problema é chamado de **problema de programação não-linear**.



# Otimização local *versus* global

Os termos **otimização local** e **otimização global** se referem à qualidade da solução a ser buscada.

Os algoritmos mais rápidos de otimização buscam encontrar uma solução local, ou seja, um ponto no qual o valor da função objetivo vale menos (no caso de minimização) do que os pontos em sua vizinhança.

Um algoritmo para otimização global se compromete a encontrar um ponto  $x$ , dentre os possíveis valores que  $x$  pode assumir, que possui o menor valor de função objetivo possível. Em geral, encontrar uma solução global é muito mais difícil do que encontrar uma solução local.

# Algoritmos para otimização

Para encontrar uma solução para um modelo matemático, usamos algoritmos e métodos computacionais.

Em geral, algoritmos de otimização são iterativos. Eles partem de um “chute” inicial dos valores das variáveis e geram uma sequência com valores aprimorados das variáveis até atingir uma solução.

A estratégia usada para ir de um iterando a outro é o que distingue um algoritmo de outro. A maioria das estratégias usa o valor da função objetivo  $f$ , o valor das restrições  $c$  e, possivelmente, a primeira e segunda derivadas dessas funções.

Alguns algoritmos armazenam as informações obtidas em cada iteração, enquanto outros usam apenas informação local do ponto  $x$  atual.

Todo bom algoritmo de otimização deve possuir os seguintes objetivos:

- **Robustez:** deve funcionar bem em uma grande variedade de problemas da classe que ele pretende resolver, para todas as escolhas razoáveis de pontos iniciais;
- **Eficiência:** não deve exigir muito tempo computacional ou espaço em memória para ser executado;
- **Precisão:** deve ser capaz de identificar uma solução com precisão, sem ser muito sensível a erros nos dados ou a erros de arredondamento que podem ocorrer durante sua execução.

Estes objetivos podem ser conflitantes. Por exemplo, um método que convirja rapidamente para a solução pode necessitar de muito espaço de armazenamento para problemas de grande porte. Por outro lado, métodos mais robustos podem ser mais lentos.

O equilíbrio entre velocidade de convergência e armazenamento necessário, robustez e velocidade, etc, são pontos centrais em otimização numérica.

Quando um algoritmo se propõe a encontrar a solução ótima de um modelo (a menos de erros de arredondamento), e tem essa propriedade demonstrada, ele é dito **exato**.

Para encontrar a solução de um modelo, um algoritmo pode levar tempo computacional muito grande, inviabilizando sua utilização na prática. Neste caso, outros algoritmos podem ser usados que, no entanto, não garantem encontrar a solução ótima.

Algoritmos que garantem alguma distância máxima pré-definida entre a solução ótima e a solução obtida pelo algoritmo são chamados de **algoritmos de aproximação**

Algoritmos que não têm nenhuma garantia de qualidade a respeito da solução obtida são chamados de **heurísticas**. Em geral, apesar de nenhuma garantia, as soluções obtidas por estes algoritmos são de boa qualidade e obtidas em tempo computacional baixo.