Programação Linear (e rudimentos de otimização não linear)

notas de aula – versão 87

Jerônimo C. Pellegrini

1 de outubro de 2016

Apresentação

Esta é uma introdução à Otimização, com ênfase em Programação Linear e aplicações, mas expondo também alguns rudimentos de Otimização Não Linear.

Os pré-requisitos para a leitura são Álgebra Linear, para as partes I e II, e Cálculo em Várias Variáveis para os Capítulos de Programação Quadrática e Otimização não linear.

Sumário

Notação

Negrito é usado para vetores, inclusive colunas de matrizes, quando tratadas isoladamente.

$\arg\max_i\{\ldots\}$ é o índice, dentre todos os i, que determina o máximo do conjunto (usamos arg min de forma análoga).

A notação $\lfloor x \rfloor$ é para "chão de x", ou "maior inteiro menor ou igual a x". Similarmente, $\lceil x \rceil$ é usado para o conceito simétrico (menor inteiro maior ou igual a x, ou "teto de x")[1].

É usada a notação "s.a." para "sujeito a".

Neste texto, o índice j é usado para colunas da matriz A, que descreve as restrições do programa linear. Neste contexto, uso "$j \leq m$" e "$j > m$" como formas abreviadas de "j pertencente à base" e "j fora da base".

[1] O leitor possivelmente conhece a notação $[\![x]\!]$ e $]\!]x[\![$.

Parte I

Programação Linear

Capítulo 1

Programação Linear

"Problema de programação linear" é o nome dado a alguns problemas de otimização. Um problema deste tipo consiste em encontrar o melhor valor possível para uma função linear de n variáveis respeitando uma série de restrições, descritas como desigualdades lineares.

Damos um exemplo inicial, que ilustra a origem dos problemas de programação linear.

Uma fábrica produz dois tipos de tinta, t_1 e t_2. Queremos determinar a quantidade ótima de cada tinta para maximizar o lucro da empresa. Cada tinta tem um custo por litro, e um determinado lucro, e há também restrições relacionadas aos níveis de produção de cada uma.

- As tintas t_1 e t_2 tem custo de produção por litro igual a \$5 e \$4, respectivamente. Os lucros da empresa com as tintas são de \$6 e \$5.5;
- O orçamento da empresa só permite gastar \$1400 com a produção de tintas;
- Um contrato de fornecimento com uma outra empresa exige que pelo menos 60 litros da tinta t_1 seja produzido;
- A empresa pode produzir no máximo 250 litros da tinta t_2 por mês, por restrição do fornecedor, que não consegue atender a demanda maior que essa.

Passamos agora à modelagem deste problema: encontraremos uma descrição formal dele, que facilitará a obtenção da solução.

O lucro, que queremos maximizar, é dado pela função $6t_1 + 5.5t_2$. Esta é nossa *função objetivo*.

As restrições que impomos à solução são:

- $5t_1 + 4t_2 \leq 1400$ (orçamento)

- $t_1 \geq 60$ (contrato)

- $t_2 \leq 250$ (oferta do fornecedor de pigmento)

- $t_1, t_2 \geq 0$ (a empresa não pode "desfazer" tinta)

Como tanto o objetivo como as restrições são lineares nas variáveis t_1 e t_2 que queremos determinar, dizemos que este é um problema de *otimização linear* (ou de *programação linear*). Na maior parte deste livro, trataremos de problemas deste tipo. Em alguns Capítulos abordaremos também problemas de otimização não linear, onde a função objetivo e as restrições não precisam ser funções lineares.

Este é um exemplo bastante prático e aplicado. Há outros problemas, completamente diferentes deste, que podem ser modelados como programação linear – inclusive problemas mais abstratos e sem uma ligação aparente tão clara com aplicações práticas. Mais exemplos de aplicação modelagem serão dados no final deste Capítulo.

Definição 1.1 (problema de programação linear)**.** Um *problema de programação linear* consiste de uma função objetivo, linear em n variáveis, e um conjunto de restrições, representadas por m desigualdades (ou igualdades) lineares.

Uma *solução* para o problema é um vetor em $\mathbb{R}^n$. Uma solução é *viável* se satisfaz as restrições, e *inviável* caso contrário.

Um problema pode ser de *maximização* ou de *minimização*. A *solução ótima* para o problema é a solução viável que dá o maior valor possível à função objetivo, quando o problema é de maximização, ou aquela que dá o menor valor ao objetivo, se o problema é de minimização. ♦

Problemas de programação linear são normalmente descritos em uma *forma padrão*, detalhada na próxima seção.

1.1 Forma padrão

Na maior parte do tempo, presumiremos que o programa linear está na seguinte forma.

$$\begin{aligned} \max\ & c_1x_1 + c_2x_2 + \cdots + c_nx_n \\ \text{s.a.}:\ & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ & \qquad\qquad\qquad\qquad\qquad\ \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{aligned}$$

e $x_i \geq 0$ para alguns i.

Mostramos agora como transformar qualquer programa linear em outro equivalente na forma padrão.

- *Maximização/minimização:* notamos que $\min \mathbf{c}^T\mathbf{x}$ é o mesmo que $\max -\mathbf{c}^T\mathbf{x}$.

- *Desigualdades:* considere a restrição

  $$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \geq b_i.$$

  Esta desigualdade pode ser transformada em

  $$\begin{aligned} a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n - s_i &= b_i, \\ s_i &\geq 0 \end{aligned}$$

  Chamamos s_i de *variável de folga* quando a desigualdade vale com o lado direito igual ao esquerdo, temos $s_i = 0$. Quando o lado direito da desigualdade é menor que o esquerdo, temos $s_i > 0$. Da mesma forma, podemos transformar

  $$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i.$$

  em

  $$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n + u_i = b_i.$$

- *Variáveis com valores negativos:* se, em uma solução, uma variável x_i puder ser negativa, usamos duas variáveis positivas para representá-la. Fazemos $x_i = p - q$, com $p, q \geq 0$.

Exemplo 1.2. Ilustramos as técnicas descritas transformando o seguinte problema para a forma padrão.

$$\begin{aligned} &\min x_1 - 3x_2 + 5x_3 \\ &\text{s.a.} : 4x_1 + x_2 \geq 10 \\ &\qquad x_1 - 3x_3 \leq 20 \\ &\qquad x_3 + x_2 = 15 \\ &\qquad x_1, x_2 \geq 0 \end{aligned}$$

Para transformar o problema de minimização em maximização, multiplicamos a função objetivo por -1.

$$\max -x_1 + 3x_2 - 5x_3$$

Incluímos uma variável nova a cada desigualdade, obtendo

$$\begin{aligned} 4x_1 + x_2 - s_1 &= 10 \\ x_1 - 3x_3 + s_2 &= 20 \end{aligned}$$

Finalmente, determinamos $x_3 = v - w$, com $v, w \geq 0$. O programa linear é então

$$\begin{aligned} &\max \ -x_1 + 3x_2 - 5v + 5w \\ &\text{s.a.} : 4x_1 + x_2 - s_1 = 10 \\ &\qquad x_1 - 3v + 3w + s_2 = 20 \\ &\qquad v - w + x_2 = 15 \\ &\qquad x_i, v, w \geq 0 \end{aligned}$$

◀

Também será conveniente representar programas lineares usando notação matricial. Claramente, um problema

$$\begin{aligned} &\max c_1x_1 + c_2x_2 + \cdots + c_nx_n \\ &\text{s.a.} : a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ &\qquad a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ &\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\ &\qquad a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \\ &\qquad x_i \geq 0 \end{aligned}$$

pode ser descrito de forma mais compacta como

$$\begin{aligned} &\max \mathbf{c}^T\mathbf{x} \\ &\text{s.a.} : A\mathbf{x} = \mathbf{b} \\ &\qquad \mathbf{x} \geq 0 \end{aligned}$$

Exemplo 1.3. Para descrever o problema do exemplo 1.2, renomeamos as variáveis

$$\begin{aligned} x_1 &\to y_1 \\ x_2 &\to y_2 \\ v &\to y_3 \\ w &\to y_4 \\ s_1 &\to y_5 \\ s_2 &\to y_6, \end{aligned}$$

e o programa linear pode ser reescrito como $\max \mathbf{c}^T\mathbf{y}$, sujeito a $A\mathbf{y} = \mathbf{b}$, $\mathbf{y} \geq 0$, com

$$\begin{aligned} \mathbf{y} &= \begin{pmatrix} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 \end{pmatrix}^T \\ \mathbf{c} &= \begin{pmatrix} -1 & 3 & -5 & 5 & 0 & 0 \end{pmatrix}^T \\ \mathbf{b} &= \begin{pmatrix} 10 & 20 & 15 \end{pmatrix}^T \end{aligned} \qquad A = \begin{pmatrix} 4 & 1 & 0 & 0 & -1 & 0 \\ 1 & 0 & -3 & 3 & 0 & 1 \\ 0 & 1 & 1 & -1 & 0 & 0 \end{pmatrix} \quad \blacktriangleleft$$

1.2 Interpretação geométrica

Nesta seção damos a interpretação geométrica de problemas de programação linear, e um método para solução de programação linear por inspeção do gráfico. Este método não deve ser visto como ferramenta prática, já que só é viável para problemas com duas variáveis e poucas restrições. Além disso, Normalmente qualquer ferramenta computacional produzirá resultados mais rapidamente do que este método. Ele é, no entanto, útil para aprofundar a compreensão do que é um problema de programação linear e quais são suas soluções.

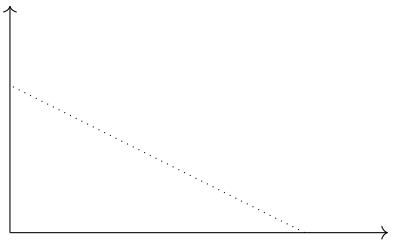
A função objetivo define um hiperplano (uma reta em $\mathbb{R}^2$, um plano em $\mathbb{R}^3$). Note que queremos maximizá-la (ou minimizá-la), portanto não nos fará diferença se a modificarmos adicionando uma constante: maximizar

$$z = 2x_1 + 3x_2$$

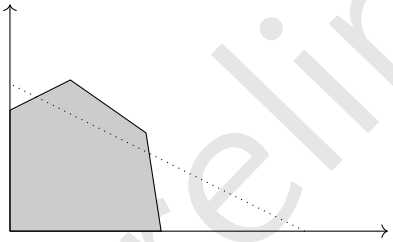
é o mesmo que maximizar

$$z' = 2x_1 + 3x_2 + 50.$$

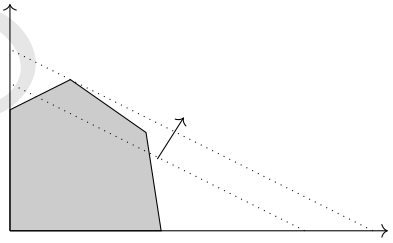
Geometricamente a função objetivo, como a definimos, passa pela origem. Quando somamos a ela um valor constante, ela se afasta dali.



Cada restrição define um semiespaço: em $\mathbb{R}^2$, um dos lados de uma reta; em $\mathbb{R}^3$, um dos lados de um plano.



Queremos o maior valor da função objetivo dentro da região definida pelas restrições. A pequena seta na próxima figura mostra o gradiente da função objetivo.



Se nos movermos, a partir da reta definida pela função objetivo, na direção de seu gradiente, mantendo-nos dentro da região viável, conseguiremos soluções cada vez melhores. Isso é o mesmo que mover a reta definida pela função objetivo na direção de seu gradiente, até que não possa mais ser "empurrada". O último ponto em que ela tocar a região viável é a solução ótima para o problema.

Ainda podemos fazer outra observação importante: vemos claramente que se uma solução ótima existe, ela ocorrerá em um dos pontos extremos ("cantos") da região viável – ou seja, na interseção de duas restrições.

Disso podemos extrair um método para obter a solução ótima de programas lineares por inspeção do gráfico, desde que tenhamos apenas duas variáveis (o que não é comum em problemas reais).

Exemplo 1.4. Resolveremos o seguinte problema de programação linear:

$$\begin{aligned} \max\ & 2x_1 + x_2 \\ \text{s.a.:}\ & 3x_1 + x_2 \leq 15 \\ & x_2 \leq 6 \\ & x_1 + x_2 \leq 7 \\ & x_1 \leq 9/2 \\ & \mathbf{x} \geq 0 \end{aligned}$$

Plotamos apenas as restrições, e verificamos que formam uma região fechada no plano.

Notamos que há poucas restrições, portanto podemos simplesmente calcular os pontos onde as retas se interceptam e verificar qual desses pontos nos dá o maior valor para a função objetivo.

ponto	valor
$A = (0, 6)$	6
$B = (1, 6)$	8
$C = (4, 3)$	11
$D = (9/2, 3/2)$	10.5
$E = (9/2, 0)$	9

Verificamos que o ponto C nos dá o maior valor, portanto a solução $(4, 3)$ é ótima, com valor 11. ◀

1.3 Fluxo em redes

Tratamos nesta seção de uma aplicação onde a correspondência entre os objetos modelados e as variáveis do programa linear não são tão imediatamente óbvias como no caso de mistura ótima.

Informalmente, uma rede é semelhante a um conjunto de tubos por onde escoamos algum fluido. Queremos decidir quanto de fluido devemos passar em cada tubo a fim de maximizar o fluxo entre dois pontos. Para definir redes de fluxo com mais rigor, precisaremos definir grafos dirigidos.

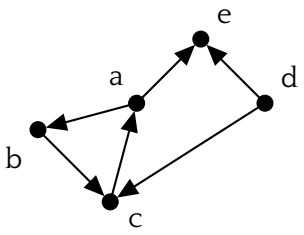
Um *grafo* é um conjunto de nós (ou "vértices") ligado por arestas. Usamos grafos, por exemplo, para representar locais e caminhos entre eles.

Definição 1.5 (grafo dirigido)**.** Um *grafo direcionado* consiste de um conjunto não vazio de vértices V e um conjunto de arestas E. O conjunto de arestas representa ligações entre os vértices, de forma que $E \subseteq V^2$. Denotamos por (u, v) a aresta que leva do vértice u ao vértice v.

Em um grafo com *peso nas arestas*, cada aresta tem um valor associado, dado por uma função $w : E \to \mathbb{R}$. ♦

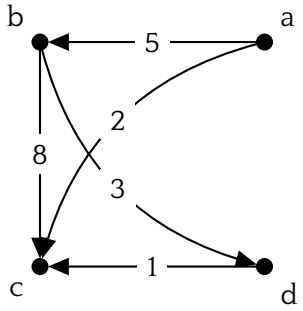
Exemplo 1.6. A seguir mostramos um grafo e sua representação de um grafo como diagrama.

$$V = \{a, b, c, d, e\}$$
$$E = \begin{Bmatrix} (a,b), & (b,c), & (c,a) \\ (d,c), & (d,e), & (a,e) \end{Bmatrix}$$



Note que a disposição dos vértices e arestas no diagrama é arbitrária. O grafo seria o mesmo se movêssemos estes vértices e arestas de lugar, mantendo as ligações entre eles. ◀

Exemplo 1.7. A seguir mostramos um grafo com pesos nas arestas



O grafo pode ser descrito da seguinte maneira:

$$V = \{a, b, c, d\}$$
$$E = \begin{Bmatrix} (a,b), & (a,c), & (b,c) \\ (b,d), & (d,c) & \end{Bmatrix}$$
$$w[(a,b)] = 5$$
$$w[(a,c)] = 2$$
$$w[(b,c)] = 8$$
$$w[(b,d)] = 3$$
$$w[(d,c)] = 1$$

◀

Definição 1.8 (Rede de fluxo)**.** Uma rede de fluxo é um grafo dirigido acíclico onde são identificados dois vértices: um é a *fonte* e o outro é o *destino*. Não há arestas chegando à fonte, e não há arestas saindo do destino.

O peso em cada aresta representa uma *capacidade*.

Um *fluxo* em uma rede é uma atribuição de um número (o "fluxo") a cada aresta (o fluxo deve ser menor ou igual à capacidade da aresta). ♦

Um problema de fluxo em rede consiste em encontrar, em uma rede, dentre todos os fluxos possíveis, aquele com maior valor total.

Seja x_{ij} o fluxo do vértice v_i para o vértice v_j.

O fluxo entrando em um nó i deve ser igual ao fluxo saindo dele.

$$\sum_j x_{ji} = \sum_j x_{ij}$$

Os fluxos saindo da fonte e chegando ao destino são iguais.

$$\sum_j x_{jd} = \sum_j x_{sj}$$

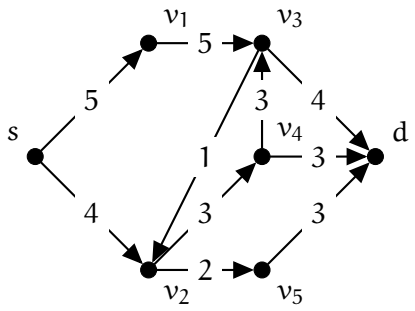
O fluxo em uma aresta deve ser menor ou igual que a capacidade da aresta.

$$x_{ij} \leq w_{ij}$$

Concluímos portanto que o fluxo ótimo na rede é dado pela solução ótima do problema de programação linear a seguir.

$$\begin{aligned} \max \ & \sum x_{ij} \\ \text{s.a.}: \ & \sum_j x_{ji} - \sum_j x_{ij} = 0 \qquad (\forall i) \\ & \sum_j x_{jd} - \sum_j x_{sj} = 0 \\ & x_{ij} \leq w_{ij} \\ & x_{ij} \geq 0 \end{aligned}$$

Exemplo 1.9. Considere a rede de fluxo a seguir.



Sua solução ótima é também a solução ótima do programa linear a seguir

(para simplificar a notação, fazemos $x_0 = s$ e $x_6 = d$.

$$
\begin{aligned}
&\max \mathbf{c}^T\mathbf{x} \\
&\text{s.a.} : x_{01} - x_{10} = 0 \\
&\qquad x_{12} - x_{21} = 0 \\
&\qquad\quad \vdots \\
&\qquad x_{56} - x_{65} = 0 \\
&\qquad x_{01} + x_{02} - x_{36} - x_{46} - x_{56} = 0 \\
&\qquad x_{01} \leq 5 \\
&\qquad x_{02} \leq 4 \\
&\qquad x_{13} \leq 5 \\
&\qquad x_{24} \leq 3 \\
&\qquad x_{25} \leq 2 \\
&\qquad x_{32} \leq 1 \\
&\qquad x_{36} \leq 4 \\
&\qquad x_{43} \leq 3 \\
&\qquad x_{46} \leq 3 \\
&\qquad x_{56} \leq 3 \\
&\qquad \mathbf{x} \geq 0
\end{aligned}
$$

◀

Notas

Os exemplos dados neste Capítulo não foram apresentados de forma detalhada; sua função aqui é apenas ilustrar a utilidade da modelagem de problemas como programas lineares.

Programação linear não é o melhor método para resolver problemas de fluxo em redes, mas é útil para entender a estrutura dos problemas e como para modelagem e solução rápida de pequenos exemplos. O livro de Cormen, Leiserson, Rivest e Stein [Cor+09] contém um capítulo sobre fluxo em redes. O livro de Bazaraa, Jarvis e Sherali [BSS06] trata mais extensivamente do assunto.

Exercícios

Ex. 1 — Resolva os programas lineares usando o método gráfico.

$$\begin{aligned} \min\ & x_1 - 2x_2 \\ \text{s.a.}: \ & 3x_1 + x_2 \leq 6 \\ & 2x_1 - x_2 \leq 9 \\ & x_1, x_2 \geq 0 \end{aligned} \qquad \begin{aligned} \min\ & x_1 + 3x_2 \\ \text{s.a.}: \ & x_1 - 2x_2 \leq 50 \\ & 2x_1 + 4x_2 \geq 30 \\ & -x_1 + 4x_2 \geq 15 \\ & x_1, x_2 \geq 0 \end{aligned} \qquad \begin{aligned} \min\ & x_1 - 3x_2 \\ \text{s.a.}: \ & x_1 + x_2 \leq 10 \\ & x_1 - x_2 \geq 5 \\ & 2x_1 + 3x_2 \geq 10 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Ex. 2 — Converta o programa linear a seguir para a forma padrão de problema de maximização.

$$\begin{aligned} \min\ & x_1 - 3x_2 + 5x_3 \\ \text{s.a.}: \ & 8x_1 + x_2 - x_3 \geq 100 \\ & 2x_1 - x_2 + 4x_3 \leq 120 \\ & x_1 + x_2 \geq x_3 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Ex. 3 — Converta o programa linear apresentado na introdução para a forma padrão de problema de maximização.

Ex. 4 — Como seria a formulação, como programa linear, de uma rede de fluxo onde (i) não há conservação de fluxo nos nós (ou seja, cada nó v_i pode produzir ou consumir uma quantidade fixa f_i de fluxo), e (ii) onde cada canal, além de ter um fluxo máximo suportado, tem também um mínimo?

Capítulo 2

Modelagem

Nesta Capítulo descrevemos a modelagem de alguns problemas de otimização. Estes devem ser tratados como exemplos, e *não* constituem uma lista exaustiva de técnicas de modelagem, porque esta é uma atividade criativa, e não teríamos como catalogar todas as formas diferentes de pensar a modelagem de problemas.

No entanto, dividimos o capítulo em seções. Cada seção contém exemplos de problemas modelados em grandes categorias. A primeira é programação linear simples, e as outras são variantes desta: fracionária, onde a função objetivo é uma função racional; inteira, onde só são admitidos valores inteiros para variáveis; e convexa. Em algumas seções apresentamos alguns truques simples de modelagem.

2.1 Programação Linear

Nesta seção apresentamos alguns exemplos importantes de aplicação de programação linear. Deixamos de lado tres deles, que apresentamos em mais detalhes mais adiante: o Capítulo 10 trata detalhadamente da estrutura de *problemas de transporte* e de algoritmos para resolvê-los; já o Capítulo 11 aborda a modelagem de problemas em Teoria de Jogos; finalmente, o Capítulo 12 mostra como alguns problemas de controle podem ser resolvidos com o uso de programação linear.

2.1.1 Exemplos básicos

Exemplo 2.1 (Mistura ótima)**.** É comum que se queira determinar a proporção ótima de mistura de diferentes ingredientes, insumos, recursos, etc, a fim de otimizar alguma função. Dizemos que estes são problemas

de *mistura ótima*. O problema apresentado no início do Capítulo um é de mistura ótima (procura-se determinar a proporção ideal de dois produtos para maximizar o lucro). Damos aqui outro exemplo de mistura ótima.

◀

Exemplo 2.2 (Dieta)**.** O problema da dieta foi um dos primeiros a serem estudados no contexto da otimização linear. A idéia é tentar conseguir uma dieta que ofereça o mínimo necessário para manter saudáveis milhares de soldados, reduzindo tanto quanto possível o custo da dieta. O problema, claro, pode ser traduzido para outros contextos.

Um nutricionista determinou que, pra um de seus pacientes, as necessidades diárias de alguns itens alimentares são como na tabela a seguir.

	min	max
carboidratos	268	387
fibras	25	50
proteínas	50	100
Ca	1000	2500
B_1	1.2	
B_6	1.3	100

Suponha (e esta é de fato uma suposição não realista, e *não* uma recomendação de dieta) que nutricionista e paciente tenham chegado a um acordo quanto aos alimentos que serão usados na dieta. A composição desses alimentos e seus preços são dados na próxima tabela. Todos os valores são por 100g de cada alimento[1].

	ch (g)	fibra (g)	prot	Ca (mg)	B_1	B_6	preço
arroz 1	28.1	1.6	2.5	4	Tr	Tr	0.25
pão	74.6	3.4	10.5	19	0.38	Tr	1.5
far rosca	75.8	4.8	11.4	35	0.25	0.09	0.6
batata	11.9	1.3	1.2	4	0.05	0.08	0.25
brócolis	4.4	3.4	2.1	51	0.04	Tr	1.3
tomate	3.1	1.2	1.1	7	0.12	0.02	0.5
b. nanica	23.8	1.9	1.4	3	Tr	0.14	0.25
figo	10.2	1.8	1.0	27	0.05	Tr	0.6
merluza	0	–	26.6	36	0.05	Tr	2.2
pto frango	0	–	31.5	5	0.1	Tr	1.2

[1] O valor nutricional foi obtido da tabela de composição de alimentos do NEPA/Unicamp [Lim+06].

Tentaremos construir uma dieta de preço mínimo que respeite as restrições de nutrição dadas e usando esses alimentos. Nosso objetivo é minimizar o preço, portanto

$$\min 0.25a+1.5p+0.6fa+0.25ba+1.3br+0.5t+0.25bn+0.6fi+22m+12fr$$

As restrições dadas são mostradas abaixo.

$$\begin{array}{rcc}
268 & \leq & 28.1a + 74.6p + 75.8fa + 11.9ba + 4.4br + 3.1t + 23.8bn + 10.2fi \leq 387 \\
25 & \leq & 1.6a + 3.4p + 4.8fa + 1.3ba + 3.4br + 1.2t + 1.9bn + 1.8fi \leq 50 \\
50 & \leq & 2.5a + 10.5p + 11.4fa + 1.2ba + 2.1br + 1.1t + 1.4bn + 1.0fi + 26.6m + 31.5fr \leq 100 \\
1000 & \leq & 4a + 19p + 35fa + 4ba + 51br + 7t + 3bn + 27fi + 36m + 5fr \leq 2500 \\
1.2 & \leq & 0.38p + 0.25fa + 0.05ba + 0.04br + 0.12t + 0.05fi + 0.05m + 0.1fr \\
1.3 & \leq & 0.09fa + 0.08ba + 0.02t + 0.14bn \leq 100 \\
 & & a + p + fa + ba + br + t + bn + fi + m + fr \leq 18
\end{array}$$

A última restrição determina que a quantidade não seja maior que 1800g (cada variável nos dá a quantidade de "porções de 100g" de alimento).

Como já temos a função objetivo e as restrições, o modelo está pronto. ◀

Exemplo 2.3 (Planejamento de produção)**.** Um produtor de leite precisa planejar sua produção ao longo do ano. Há capacidade para produzir 900ℓ por mês sem pagamento de hora extra; e 1100ℓ por mês, pagando hora extra. O custo de produção de 100ℓ de leite é \$10 sem hora extra, e 13 com hora extra. O custo de armazenamento de 100ℓ de leite (quando não é vendido no mesmo mês) é de \$1.5. O produtor tem dados históricos com a demanda típica de cada mês.

mês	demanda	mês	demanda
1	500	7	540
2	350	8	450
3	550	9	400
4	400	10	400
5	600	11	400
6	600	12	500

As variáveis cujos valores precisamos determinar são

- p_i, a produção no mes i
- e_i, a produção com hora extra no mês i
- a_i, a quantidade de leite produzido no mes i e armazenada para o mes seguinte.

Como a demanda do ano todo não deve variar muito, o produtor precisa apenas minimizar o custo de produção dessa demanda, dado por

$$10\sum_i p_i + 13\sum_i e_i + 1.5\sum_i a_i.$$

As restrições são

- $\sum_i p_i = 900$

- $\sum_i e_i = 200$

- Se a demanda em um mês é x, então a sobra do mês anterior mais produção do mes atual, menos o que é levado ao próximo mês, é igual a x.

$$\begin{aligned} p_1 + e_1 - a_1 &= 500 \\ p_2 + e_2 - a_2 + a_1 &= 350 \\ p_3 + e_3 - a_3 + a_2 &= 550 \\ &\vdots \\ p_{11} + e_{11} - a_{11} + a_{10} &= 400 \\ p_{12} + e_{12} + a_{11} &= 500 \end{aligned}$$

Já temos agora o modelo completo, que mostramos a seguir.

$$
\begin{aligned}
&\min 10\sum_i p_i + 15\sum_i e_i + 1.5\sum_i a_i. \\
&\text{s.a.: } \sum_i p_i = 900 && \text{(uma restrição para cada } i\text{)} \\
&\quad \sum_i e_i = 200 && \text{(uma restrição para cada } i\text{)} \\
&\quad p_1 + e_1 - a_1 = 500 \\
&\quad p_2 + e_2 - a_2 + a_1 = 350 \\
&\quad p_3 + e_3 - a_3 + a_2 = 550 \\
&\quad p_4 + e_4 - a_4 + a_3 = 550 \\
&\quad p_5 + e_5 - a_5 + a_4 = 550 \\
&\quad p_6 + e_6 - a_6 + a_5 = 550 \\
&\quad p_7 + e_7 - a_7 + a_6 = 550 \\
&\quad p_8 + e_8 - a_8 + a_7 = 550 \\
&\quad p_9 + e_9 - a_9 + a_8 = 550 \\
&\quad p_{10} + e_{10} - a_{10} + a_9 = 550 \\
&\quad p_{11} + e_{11} - a_{11} + a_{10} = 400 \\
&\quad p_{12} + e_{12} + a_{11} = 500 \\
&\quad p_i, e_i, a_i \geq 0
\end{aligned}
$$

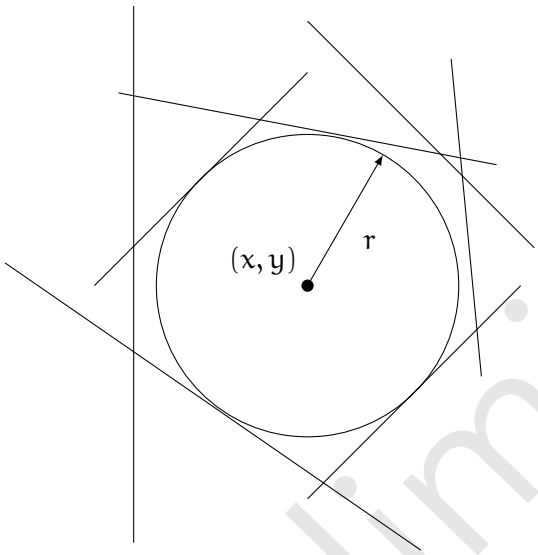
◀

Exemplo 2.4 (Geometria)**.** Em sua introdução à Programação Linear [MG06], Jiri Matousek e Bernd Gartner dão um interessante exemplo em geometria, que elaboramos também neste exemplo.

Um polígono convexo em $\mathbb{R}^2$ pode ser descrito por um número de retas. Como cada reta é da forma $y = ax + b$, o polígono é descrito por

$$
\begin{aligned}
y &= a_1 x + b_1 \\
y &= a_2 x + b_2 \\
&\vdots \\
y &= a_n x + b_n
\end{aligned}
$$

Queremos saber qual é a maior circunferência que podemos descrever dentro desse polígono. A circunferência é descrita por centro e raio – portanto, três variáveis: x, y, r.



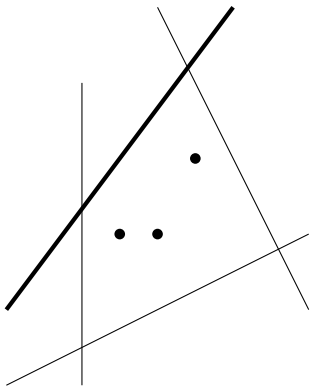
Claramente, para maximizar a área temos que maximizar o raio, portanto já temos nossa função objetivo.

As restrições devem descrever uma única limitação: para cada parede do polígono, o raio deve ser menor que a distância do centro até a parede.

A distância do centro a cada uma das paredes é

$$d_i = \left| \frac{y - a_i x - b_i}{\sqrt{a_i^2 + 1}} \right|. \tag{2.1}$$

Uma das paredes do polígono estará acima ou abaixo de *todos* os pontos no interior do polígono, como mostra a figura a seguir: a parede destacada em negrito está acima de todos os pontos no interior do polígono (alguns deles são mostrados).



O valor dentro do módulo na fórmula 2.1 será positivo quando a linha estiver abaixo do centro (ou de qualquer ponto dentro do polígono), e negativa quando a linha estiver acima do centro.

Sem perda de generalidade, suponha que as linhas $1, 2, \ldots, k$ estejam abaixo do centro, e que as linhas $k+1, k+2, \ldots, n$ estejam acima do centro. Com estas condições, nossas restrições serão

$$\frac{y - a_i x - b_i}{\sqrt{a_i^2 + 1}} \geq r, \quad \text{para } i \leq k$$
$$\frac{y - a_i x - b_i}{\sqrt{a_i^2 + 1}} \leq -r, \quad \text{para } i > k$$

Reorganizando as desigualdades, temos

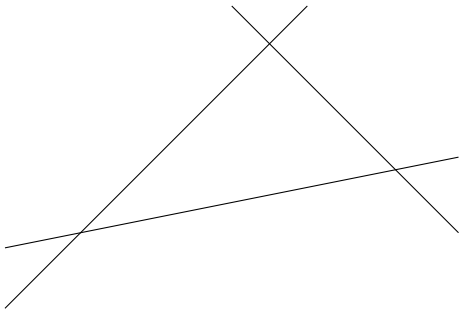
$$\left(\frac{1}{\sqrt{a_i^2 + 1}}\right) y - \left(\frac{a_i}{\sqrt{a_i^2 + 1}}\right) x - r \geq \left(\frac{b_i}{\sqrt{a_i^2 + 1}}\right) \quad \text{para } i \leq k$$
$$\left(\frac{1}{\sqrt{a_i^2 + 1}}\right) y - \left(\frac{a_i}{\sqrt{a_i^2 + 1}}\right) x + r \leq \left(\frac{b_i}{\sqrt{a_i^2 + 1}}\right) \quad \text{para } i > k$$

Note que os valores entre parenteses são constantes, porque fazem parte da descrição do polígono. O modelo completo é mostrado a seguir.

$$\begin{aligned}
&\max r \\
&\text{s.a.:} \left(\frac{1}{\sqrt{a_i^2 + 1}}\right) y - \left(\frac{a_i}{\sqrt{a_i^2 + 1}}\right) x - r \geq \left(\frac{b_i}{\sqrt{a_i^2 + 1}}\right) \quad \text{para } i \leq k \\
&\qquad \left(\frac{1}{\sqrt{a_i^2 + 1}}\right) y - \left(\frac{a_i}{\sqrt{a_i^2 + 1}}\right) x + r \leq \left(\frac{b_i}{\sqrt{a_i^2 + 1}}\right) \quad \text{para } i > k \\
&\qquad x, y, r \geq 0
\end{aligned}$$

Damos um exemplo simples: tomamos o triângulo definido pelas retas

$$y = x$$
$$y = x/5$$
$$y = -x + 3$$



A descrição das retas é dada pelos pares (a_i, b_i) a seguir.

$$(1,0)$$
$$(-1,5)$$
$$(1/5,0)$$

Para $(-1,5)$, por exemplo, temos

$$\left(\frac{1}{\sqrt{a_i^2+1}}\right) y - \left(\frac{a_i}{\sqrt{a_i^2+1}}\right) x - r \geq \left(\frac{b_i}{\sqrt{a_i^2+1}}\right)$$
$$\left(\frac{1}{\sqrt{(-1)^2+1}}\right) y - \left(\frac{(-1)}{\sqrt{(-1)^2+1}}\right) x - r \geq \left(\frac{5}{\sqrt{(-1)^2+1}}\right)$$
$$\left(\frac{1}{\sqrt{2}}\right) y + \left(\frac{1}{\sqrt{2}}\right) x - r \geq \left(\frac{5}{\sqrt{2}}\right)$$

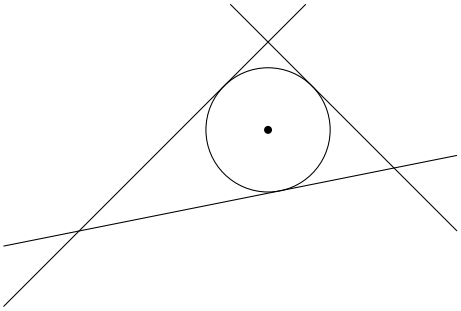
A construção das restrições para os outros pontos é semelhante. O modelo final é mostrado a seguir.

$$\begin{aligned} \max\ & r \\ \text{s.a.:}\ & \frac{1}{\sqrt{2}}y - \frac{1}{\sqrt{2}}x + r \leq 0 \\ & \frac{1}{\sqrt{2}}y + \frac{1}{\sqrt{2}}x + r \leq \frac{5}{\sqrt{2}} \\ & \frac{5}{\sqrt{26}}y - \frac{1}{\sqrt{26}}x - r \geq 0 \\ & x, y, r \geq 0 \\ & x, y, r \in \mathbb{R} \end{aligned}$$

O ótimo para este problema é

$$(x, y) = (2.5, 1.3376), \quad r = 0.821687$$

A próxima figura mostra a circunferencia inscrita no triângulo[2].



Embora Matousek e Gartner tenham apresentado este problema apenas abstratamente como como otimização em geometria, é fácil perceber sua aplciabilidade prática: poderíamos ter que determinar, por exemplo, quão largo pode ser um cilindro que queremos encaixar em um dado local delimitado por superfícies planas em um trabalho de engenharia. ◀

2.1.2 Valores absolutos

Quando queremos otimizar o valor absoluto de uma variável x_i, mas não restringí-la a valores positivos, podemos expressar x_i como a diferença de duas outras variáveis, ambas positivas:

$$x_i = x_i^p - x_i^n.$$

Necessariamente teremos sempre um dentre x_i^p, x_i^n igual a zero, e

$$|x_i| = x_i^p + x_i^n.$$

Exemplo 2.5 (Regressão linear)**.** Temos diversos dados obtidos experimentalmente, e queremos determinar a curva que melhor aproxima esses dados.

Os dados estão na forma (x_i, y_i).

Se decidirmos por uma reta definida por $ax + b$, o erro em cada ponto será

$$z_i = ax_i + b - y_i$$

Não podemos simplesmente minimizar o erro, porque ele pode ser negativo. Precisamos minimizar o valor absoluto do erro,

$$\min \sum_i |ax_i + b - y_i|,$$

[2] O posicionamento da circunferência na página foi de fato calculado através do programa linear que apresentamos!

e representamos cada erro explicitamente como $z_i \geq 0$, portanto a função objetivo é

$$\min z_1 + z_2 + \cdots + z_n$$

Construiremos as restrições da seguinte maneira:

$$ax_i + b - y_i = \underbrace{z_i^p + z_i^n}_{|z_i|}$$

Mas podemos também evitar duplicar cada z_i, da seguinte maneira:

$$\begin{aligned} ax_i + b - y_i &\leq z_i \\ -(ax_i + b - y_i) &\leq z_i \end{aligned}$$

E as restrições são

$$\begin{aligned} ax_1 + b - y_1 &\leq z_1 \\ -(ax_1 + b - y_1) &\leq z_1 \\ ax_2 + b - y_2 &\leq z_2 \\ -(ax_2 + b - y_2) &\leq z_2 \\ &\vdots \\ ax_n + b - y_n &\leq z_n \\ -(ax_n + b - y_n) &\leq z_n \\ \mathbf{z} &\geq \mathbf{0} \end{aligned}$$

onde a, b e os z_i são as variáveis a serem determinadas, e os x_i, y_i são dados. ◀

2.1.3 Objetivo minimax

Podemos querer também minimizar o máximo (ou maximizar o mínimo) de um conjunto de funções.

$$\min \quad \max_q c_{qi} x_i$$

Podemos criar uma nova variável w, que representa o máximo a ser minimizado:

$$w = \max_q c_{qi} x_i$$

Para cada q, adicionamos uma restrição,

$$\sum_i c_{qi} x_i \leq w.$$

Desta forma, ao minimizarmos z,

- As restrições garantem que w será maior que a maior das somas para cada q;
- Como minimizamos w, ele não será maior que nenhuma das somas.

Exemplo 2.6 (regressão linear)**.** No exemplo 2.5, minimizamos a soma dos erros. Se quisermos minimizar o maior dos erros, teremos o objetivo

$$\min \quad \max_i |ax_i + b - y_i|,$$

ou

$$\min \quad \max_i |z_i|.$$

Podemos criar uma nova variável w, que representa o máximo a ser minimizado:

$$w = \max_i |z_i|$$

E construimos nosso modelo minimizando w:

$$\begin{aligned} &\min w \\ &\text{s.a.: } ax_i + b - y_i \leq w \quad \forall i \\ &\qquad -(ax_i + b - y_i) \leq w \quad \forall i \\ &\qquad a, b, w \geq 0 \end{aligned}$$

As variáveis são a, b, w. ◀

2.2 Programação Fracionária

No problema a seguir a função objetivo não é linear, mas é dado pelo quociente de duas funções afim:

$$\begin{aligned} \min \ & \frac{\mathbf{c}^T \mathbf{x} + d}{\mathbf{g}^T \mathbf{x} + f} \\ \text{s.a.: } & A\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned} \tag{2.2}$$

Damos o nome a este tipo de problema de *problema de programação fracionária*. Podemos chegar a formulação de problemas deste tipo quando quisermos levar em consideração na função objetivo, uma relação custo/benefício, como por exemplo o o quociente entre lucro por unidade de produto e custo de cada unidade.

Exemplo 2.7 (mistura ótima)**.** Há uma variante do problema de mistura ótima que é formulada como problema de programação fracionária.

Suponha que uma fábrica precise produzir uma liga metálica contendo entre 0.6% e 2.1% de Carbono e no mínimo 7% de Tungstênio, Molibdênio e Vanádio[3]. Esta liga será vendida por \$240/kg. Ele pretende misturar outras quatro ligas disponíveis, cuja quantidade de Carbono (Ca), Tungstênio (W), Molibdênio (Mo) e Vanádio (V) é dada na tabela a seguir.

	L_1	L_2	L_3	L_4
Fe	2.5%	2.0%	2.0%	2.0%
W	2.0%	1.0%	4.0%	0.5%
Mo	2.0%	1.0%	3.0%	1.0%
V	1.0%	1.0%	3.0%	0.0%
qtde	100	150	100	200
custo	180	100	280	190

Para determinar a mistura ideal das quatro ligas, definimos as variáveis x_1, x_2, x_3, x_4, que expressam a quantidade de cada uma. O custo será dado por

$$180x_1 + 100x_2 + 280x_3 + 190x_4.$$

Como a liga será vendida por 240/kg, o retorno será de

$$240(x_1 + x_2 + x_3 + x_4)$$

Isto nos dá o objetivo,

$$\frac{240x_1 + 240x_2 + 240x_3 + 240x_4}{180x_1 + 100x_2 + 280x_3 + 190x_4}.$$

As restrições são

$$0.6\% \leq \text{Fe} \leq 2.1\%, \quad W + \text{Mo} + \text{V} \geq 7\%,$$

[3]Uma liga como esta é chamada de Aço Rápido, muito usado na produção de ferramentas duras (brocas, alicates, etc.

ou seja,

$$0.006 \leq \frac{2.5x_1 + 2.0x_2 + 2.0x_3 + 2.0x_4}{x_1 + x_2 + x_3 + x_4}$$
$$0.021 \geq \frac{2.5x_1 + 2.0x_2 + 2.0x_3 + 2.0x_4}{x_1 + x_2 + x_3 + x_4}$$
$$0.07 \leq \frac{(2.0 + 2.0 + 1.0)x_1 + (1.0 + 1.0 + 1.0)x_2 + (4.0 + 3.0 + 3.0)x_3 + (0.5 + 1.0)x_4}{x_1 + x_2 + x_3 + x_4}$$

Resolvemos o sistema e reescrevemos as restrições, obtendo o modelo.

$$\begin{aligned}
\max\ & \frac{240x_1 + 240x_2 + 240x_3 + 240x_4}{180x_1 + 100x_2 + 280x_3 + 190x_4} \\
\text{s.a.:}\ & 15x_1 + x_2 + x_3 + x_4 \geq 0 \qquad \text{(redundante!)} \\
& -4x_1 + x_2 + x_3 + x_4 \geq 0 \\
& -2x_1 - 4x_2 + 3x_3 - 5.5x_4 \geq 0 \\
& x_1 \leq 100 \\
& x_2 \leq 150 \\
& x_3 \leq 100 \\
& x_4 \leq 200 \\
& \mathbf{x} \geq \mathbf{0}
\end{aligned}$$

◀

2.2.1 Transformação em programa linear

Todo problema de programação fracionária pode ser transformado em um programa linear, mas este método é ineficiente, resultando em um problema maior que o necessário. Pode ser usado, no entanto, na modelagem de problemas pequenos.

O problema a seguir é equivalente ao problema apresentado no início da seção:

$$\begin{aligned}
\min\ & \mathbf{c}^T\mathbf{y} + dz \qquad (2.3) \\
\text{s.a.:}\ & A\mathbf{y} - z\mathbf{b} \geq 0 \\
& \mathbf{g}^T\mathbf{y} + fz = 1 \\
& \mathbf{y} \geq \mathbf{0}
\end{aligned}$$

onde as variáveis são $\mathbf{y}$ e z.

Pode-se mostrar que a solução ótima para o problema 2.3 é ótima para 2.2,

definindo

$$\begin{aligned} \mathbf{y} &= \left(\frac{1}{\mathbf{g}^T\mathbf{x} + f}\right)\mathbf{x} \\ z &= \frac{1}{\mathbf{g}^T\mathbf{x} + f}. \end{aligned}$$

Exemplo 2.8. A seguir temos um problema de programação fracionária linear:

$$\begin{aligned} \min\ & \frac{2x_1 + x_2 - 3}{x_1 - x_2 + 2} \\ \text{s.a.:}\ & x_1 + x_2 \geq 10 \\ & 2x_1 - x_2 \geq 5 \\ & \mathbf{x} \geq 0 \end{aligned}$$

Temos

$$\begin{aligned} \min\ & \frac{\mathbf{c}^T\mathbf{x} + d}{\mathbf{g}^T\mathbf{x} + f} \\ \text{s.a.:}\ & A\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

com

$$A = \begin{pmatrix} 1 & 1 \\ 2 & -1 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 10 \\ 5 \end{pmatrix}, \quad \mathbf{c} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \quad \mathbf{g} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \quad d = -3, \quad f = 2$$

O problema de programação linear equivalente é

$$\begin{aligned} \min\ & \mathbf{c}^T\mathbf{y} + dz \\ \text{s.a.:}\ & A\mathbf{y} - z\mathbf{b} \geq 0 \\ & \mathbf{g}^T\mathbf{y} + fz = 1 \\ & \mathbf{y} \geq 0, \end{aligned}$$

ou seja,

$$\begin{aligned} \min\ & 2y_1 + y_2 - 3z \\ \text{s.a.:}\ & y_1 + y_2 - 10z \geq 0 \\ & 2y_1 - y_2 - 5z \geq 0 \\ & y_2 - y - 2 + 2z = 1 \\ & \mathbf{y} \geq \mathbf{0}. \end{aligned}$$

◀

2.3 Programação Inteira

Muitas vezes queremos resolver um problema de otimização linear onde só nos interessam as soluções inteiras. Um problema de programação inteira é semelhante a um problema de programação linear, mas restringindo parte das variáveis a valores inteiros:

$$\begin{aligned} &\max \mathbf{c}^T\mathbf{x} \\ &\text{s.a.} : A\mathbf{x} = \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0} \\ &\qquad x_j \in \mathbb{Z}, \text{ se } j \in K \end{aligned}$$

onde o conjunto K contém os índices das variáveis inteiras.

Exemplo 2.9 (mistura ótima inteira)**.** Uma empresa produz três produtos, A, B e C, feitos em três máquinas, M_1, M_2 e M_3. Todas as máquinas podem produzir todos os produtos, exceto que a máquina M_3 não pode ser usada para fabricar o produto C. A tabela a seguir mostra quanto cada produto usa do tempo de cada máquina para a fabricação de uma unidade, e quanto é o lucro de cada produto por unidade, bem como o tempo disponível em cada máquina.

produto	tempo necessário			lucro
A	4	1	1	7
B	7	3	3	25
C	2	2	0	6
tempo disponível	230	170	40	

Além disso, a empresa se comprometeu a entregar 10 unidades de A *e* 5 unidades de B.

Para decidir quanto produzir de A, B e C, maximizando o lucro total, formulamos o problema como programa linear.

Denominaremos a quantidade de produtos a na máquina M_i por am_i. As máquinas não podem fabricar parcialmente um produto, portanto exigimos que cada uma destas variáveis tenha valor inteiro.

A função objetivo expressa a maximização do lucro obtido com a produção dos três produtos:

$$\max\ 7A + 25B + 6C.$$

ou, já especificando quando cada máquina produzirá,

$$\max\ 7\left(\sum_i am_i\right) + 25\left(\sum_i bm_i\right) + 6\left(\sum_i cm_i\right).$$

As restrições são de dois tipos: limite de tempo e mínimo necessário de cada produto.

Os limites de tempo são

$$\begin{aligned} 4am_1 + 7bm_1 + 2cm_1 &\leq 230 \\ am_2 + 3bm_2 + 2bm_3 &\leq 170 \\ am_3 + 3bm_3 &\leq 40 \end{aligned}$$

Já os limites mínimos de produção são

$$\begin{aligned} am_1 + am_2 + am_3 &\geq 10 \\ bm_1 + bm_2 + bm_3 &\geq 5 \end{aligned}$$

O modelo está pronto!

$$\begin{aligned} \max\ & 7am_1 + 7am_2 + 7am_3 + \\ & 25bm_1 + 25bm_2 + 25bm_3 + \\ & 6cm_1 + 6cm_2 + 6cm_3 \\ \text{s.a.:}\ & 4am_1 + 7bm_1 + 2cm_1 \leq 230 \\ & am_2 + 3bm_2 + 2bm_3 \leq 170 \\ & am_3 + 3bm_3 \leq 40 \\ & am_1 + am_2 + am_3 \geq 10 \\ & bm_1 + bm_2 + bm_3 \geq 5 \\ & am_i, bm_i, cm_i \in \mathbb{Z}^+ \qquad \text{(soluções inteiras positivas!)} \end{aligned}$$

É comum escrever o modelo de forma mais compacta,

$$\begin{aligned} \max\ & 7\sum_i am_i + 25\sum_i bm_i + 6\sum_i cm_i \\ \text{s.a.:}\ & 4am_1 + 7bm_1 + 2cm_1 \leq 230 \\ & am_2 + 3bm_2 + 2bm_3 \leq 170 \\ & am_3 + 3bm_3 \leq 40 \\ & \sum_i am_i \geq 10 \\ & \sum_i bm_i \geq 5 \\ & am_i, bm_i, cm_i \in \mathbb{Z}^+ \end{aligned}$$

◀

2.3.1 Valores descontínuos

No processo de modelagem, pode acontecer de nos darmos conta de que uma determinada variável possa assumir valores em um conjunto descontínio – por exemplo, ou zero ou em $[3, 5]$.

Nesta situação, é útil criar uma nova variável (por exemplo, y), que chamamos de *variável indicadora*, cujo valor determina em qual dos intervalos está o valor da variável descontínua (no nosso exemplo, x):

$$y = \begin{cases} 0 & \text{se } x = 0 \\ 1 & \text{se } x \in [a, b] \end{cases}$$

Tendo criado a nova variável, as restrições passam a ser

$$\begin{aligned} x &\leq by \\ x &\geq ay \\ y &\in \{0, 1\} \end{aligned}$$

O modelo passa a ser de programação inteira, porque y é binária.

Exemplo 2.10. ◀

2.3.2 Restrições condicionais

Exemplo 2.11. ◀

2.3.3 Eliminando produtos de variáveis

É possível que ao formularmos um problema como programação inteira, tenhamos que usar no objetivo ou alguma restrição o produto de duas variáveis, resultando em um problema de programação não linear. No entanto, em algumas situações, é possível reformular o problema como programa linear.

Se o produto é de duas variáveis binárias x_1 e x_2, podemos substituir o produto por uma nova variável y, com as seguintes restrições:

$$\begin{aligned} y &\leq x_1 \\ y &\leq x_2 \\ y &\geq x_1 + x_2 - 1 \\ y &\in \{0, 1\} \end{aligned}$$

Quando o produto é de uma variável binária x_1 com uma contínua x_2, e há a restrição $x_2 \leq K$, introduzimos a variável $y = x_1 x_2$, e as restrições

$$\begin{aligned} y &\leq Kx_1 \\ y &\leq x_2 \\ y &\geq x_2 - K(1 - x_1) \\ y &\geq 0 \end{aligned}$$

Exemplo 2.12. ◀

2.4 Programação Não-Linear

Quando a função objetivo ou as restrições não são descritas por funções lineares, temos um problema de *programação não linear*.

Exemplo 2.13 (regressão não-linear)**.** No exemplo 2.5, tentamos obter uma reta que aproximasse um conjunto de dados, minimizando a soma dos erros. Se quisermos ajustar uma parábola (ou polinômio de grau maior), teremos um problema de programação não-linear.

A função objetivo é

$$\min z_1 + z_2 + \cdots + z_n$$

E as restrições são

$$\begin{aligned} ax_1^2 + bx_1 + c - y_1 &\leq z_1 \\ -(ax_1^2 + bx_1 + c - y_1) &\leq z_1 \\ ax_2^2 + bx_2 + c - y_2 &\leq z_2 \\ -(ax_2^2 + bx_2 + c - y_2) &\leq z_2 \\ &\vdots \\ ax_n^2 + bx_n + c - y_n &\leq z_n \\ -(ax_n^n + bx_n + c - y_n) &\leq z_n \\ \mathbf{z} &\geq \mathbf{0} \end{aligned}$$

onde a, b, c e os z_i são as variáveis a serem determinadas, e os x_i, y_i são dados.

O Exercício 10 pede a generalização deste exemplo. ◀

Notas

O problema da dieta foi proposto nos anos 40 por George Stigler: tratava-se de tentar encontrar a mais barata dieta que pudesse manter saudável um homem adulto de constituição física usual. Stigler, usando uma heurística, encontrou uma dieta que custava somente 39.93 dólares por homem, e *por ano* (a dieta custaria pouco mais de 500 dólares no ano 2000). George Dantzig, depois de criar o método Simplex para resolver problemas de programação linear, aplicou-o ao problema original da dieta, encontrando a solução ótima, que custava somente 24 centavos menos que a de Stigler por ano. O próprio George Dantzig, em um artigo de 1990 [Dan90], conta a história do problema da dieta. Anos depois de ter publicado o modelo, seu médico teria recomendado que ele perdesse peso, e ele então tentou modelar sua própria dieta e seguir o resultado produzido por um computador, mas por diversas vezes deparou-se com soluções absurdas que cladamente indicavam problemas na modelagem, como a recomendação do consumo diário de "quinhentos galões de vinagre", "duzentos cubos de caldo de carne", e "900g de farinha ou melaço".

Alguns dos exemplos desde Capítulo são adaptados do manual do texto de Bisschop [Bis14].

O método dado neste Capítulo para programação linear fracionária é simples, mas resulta em um aumento do problema a ser resolvido. Há métodos mais eficientes. Os livros de Bajalinov [Eri03] e de Stancu-Minasian [Sta97] trazem exemplos de modelagem e desenvolvem a teoria da programação linear fracionária.

Exercícios

Ex. 5 — Suponha que uma fábrica possa produzir dois produtos diferentes: jaquetas e calças de poliéster. Cada lote de jaquetas resulta em lucro de 500 para a empresa, e cada lote de calças resulta em lucro de 700.

- Cada lote de jaquetas custa 10, e cada lote de calças custa 15. O orçamento mensal da empresa que pode ser destinado à produção é igual a 400. O lucro só é realizado muito mais tarde, e cada lote deve ser pago antes da produção iniciar (isso significa que há um limite para a quantidade de lotes produzidos em um mês).
- Cada lote de jaquetas leva um dia para ser produzido, e cada lote de calças precisa de dois dias. A fábrica tem 22 dias por mês disponíveis para a produção.

- A empresa tem um contrato com o governo que a obriga a produzir no mínimo 5 lotes de jaquetas por mês.
- Além disso, há baixa oferta de fechos para as calças – somente um fornecedor, com capacidade limitada – e isto limita a produção de calças a 10 lotes por mês.

Formule o problema.

Ex. 6 — Um fazendeiro está estudando a divisão de sua propriedade nas seguintes atividades produtivas:

- A (Arrendamento) - Destinar certa quantidade de alqueires para a plantação de cana-de-açúcar, a uma usina local, que se encarrega da atividade e paga pelo aluguel da terra $300,00 por alqueire por ano.
- P (Pecuária) - Usar outra parte para a criação de gado de corte. A recuperação das pastagens requer adubação (100Kg/Alq.) e irrigação (100.000 litros de água/Alq.) por ano. O lucro estimado nessa atividade é de $400,00 por alqueire por ano.
- S (Plantio de Milho) - Usar uma terceira parte para o plantio de milho. Essa cultura requer 200Kg por alqueire de adubos 200.000 litros de água/Alq. para irrigação por ano. O lucro estimado nessa atividade é de $500,00/alqueire no ano.

A disponibilidade de recursos por ano é 12.750.000 litros de água, 14.000 Kg de adubo e 100 alqueires de terra. O fazendeiro quer determinar quantos alqueires deverá destinar a cada atividade para proporcionar o melhor retorno. Formule o problema como programa linear.

Ex. 7 — Um veículo tem 100 unidades de combustível, e deve andar por um labirinto acíclico. Caminhos diferentes consomem mais ou menos combustível, e dão ao veículo recompensas diferentes. Sabendo o mapa e os custos e recompensas, o veículo deve usar da melhor maneira possível suas 100 unidades de combustível. Modele este problema como programa linear.

Ex. 8 — No exercício 7, exigimos que o labirinto (e consequentemente o grafo que o representa) fosse acíclico. Explique porque, e diga o que aconteceria com seu modelo de programa linear se o grafo tivesse ciclos.

Ex. 9 — Demonstre a equivalência dos dois problemas mostrados na Seção 2.2.

Ex. 10 — Generalize o exemplo 2.13 para usar uma função não-linear qualquer ao invés de um polinômio.

Capítulo 3

Conjuntos Convexos e Soluções Viáveis

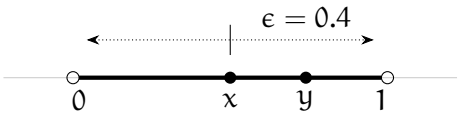
O conjunto das soluções viáveis para um programa linear é um conjunto *convexo*. Iniciamos este Capítulo com uma breve introdução à convexidade e depois abordamos algumas propriedades das soluções de programas lineares.

3.1 Conjuntos convexos

Definição 3.1 (Conjunto aberto)**.** Um conjunto X é aberto se para todo ponto $x \in X$ há uma distância ϵ tal que qualquer ponto y cuja distância de x seja menor que ϵ também pertence a X. ♦

A próxima figura mostra o intervalo $(0, 1)$, que é um conjunto aberto de números reais. Com zero e um não pertencem ao intervalo, temos que

Para qualquer $x \in (0, 1)$, há alguma distancia ϵ tal que podemos encontrar algum $y \in (0, 1)$, que fica não mais longe que ϵ de x. Isto claramente acontece porque nem zero nem um pertencem ao conjunto, e sempre podemos escolher algum número entre x e zero ou entre x e um.



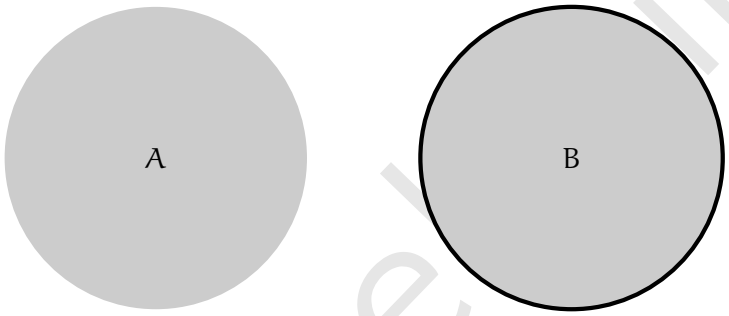
Já o intervalo $[0, 1]$ não é aberto, porque para $x = 1$ e $x = 0$ não conseguimos uma vizinhança ϵ totalmente contida no intervalo.

Definição 3.2 (Conjunto fechado)**.** Um conjunto é fechado se seu complemento é aberto. ♦

Exemplo 3.3. Os pontos internos de uma circunferência, *sem a borda*, que satisfazem $x^2 + y^2 < r^2$, formam um conjunto aberto.

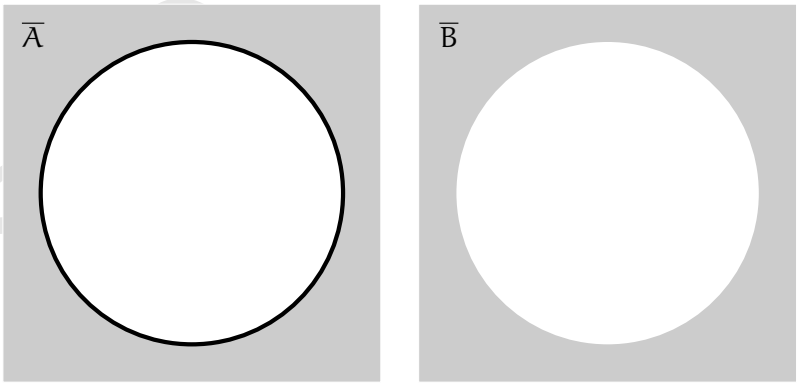
Já os pontos internos da circunferência, determinados por $x^2 + y^2 \leq r^2$, formam um conjunto fechado.

A seguir mostramos os dois casos: os pontos de uma circunferência sem a borda (A) e com a borda (B).



aberto não aberto

Mostramos a seguir os complementos desses conjuntos. Note que como estamos trabalhando em $\mathbb{R}^2$, os retângulos *não* significam que os complementos são limitados.



complemento não aberto complemento aberto

A figura da esquerda mostra que o complemento do conjunto não é aberto, portanto o conjunto A não é fechado. A da direita tem complemento aberto, logo o conjunto B é fechado. ◄

Observe que um conjunto pode ser fechado e aberto ao mesmo tempo, e também pode não ser nem fechado e nem aberto.

Exemplo 3.4. O conjunto $(0, 1]$ não é aberto: para $x = 1$, escolha qualquer distância ϵ. Não há valor $\delta > 0$ com $\delta < \epsilon$ e $1 + \delta \in (0, 1]$.

Mas o conjunto também não é fechado, porque seu complemento, $(-\infty, 0] \cup (1, +\infty)$ não é fechado – teremos o mesmo problema na fronteira do zero. ◀

Exemplo 3.5. O espaço $\mathbb{R}^n$ é aberto, porque para qualquer $\mathbf{x} \in \mathbb{R}^n$ e distância ϵ há um $\mathbf{y} \in \mathbb{R}^n$ cuja distância de $\mathbf{x}$ é menor que ϵ.

O complemento de $\mathbb{R}^n$ é o vazio. E o vazio *é também aberto, por vacuidade: a afirmação "para todo* $\mathbf{x} \in \emptyset$*, ..." sempre é verdadeira*.

Assim, $\mathbb{R}^n$ é fechado e também aberto. ◀

Trabalharemos com problemas de otimização que são bem definidos em conjuntos fechados: no conjunto $x \in (0, 1)$, por exemplo, não podemos selecionar o maior elemento.

Definição 3.6 (Combinação afim)**.** Uma *combinação afim* de $x_1, x_2, \ldots, x_n$ é uma combinação linear dos x_i, com a soma dos coeficientes igual a um. ⧫

Exemplo 3.7. Sejam x_1, x_2, x_3 elementos quaisquer. Então

$$-0.1x_1 + 0.4x_2 + 0.7x_3$$

é combinação afim de x_1, x_2, x_3. ◀

Definição 3.8 (envoltória afim)**.** A *envoltória afim* de um conjunto $X \subset \mathbb{R}^n$, denotada $\text{aff}(X)$, é o conjunto de todas as combinações afim dos elementos de X. ⧫

Exemplo 3.9. A envoltória afim de dois pontos diferentes é a reta passando por eles: seja $A = \{x, y\}$, com $x = (1, 2)$ e $y = (1, 1)$. A envoltória afim dos dois pontos é

$$\text{aff}(X) = \{\alpha x + (1 - \alpha)y,\ \alpha \in \mathbb{R}\}$$

ou

$$\begin{aligned}\text{aff}(X) &= \left\{\big(\alpha + (1-\alpha), 2\alpha + (1-\alpha)\big)\ \alpha \in \mathbb{R}\right\} \\ &= \{(1, \alpha + 1), \alpha \in \mathbb{R}\}\end{aligned}$$

A envoltória afim de quatro ou mais pontos não coplanares em $\mathbb{R}^3$ é todo o $\mathbb{R}^3$. Seja $B = \{x, y, z\}$, com $x = (1, 1)$, $y = (1, 2)$ e $z = (3, 1)$. Temos portanto

$$\begin{aligned}\text{aff}(B) &= \left\{\big(\alpha x + \beta y + (1-\alpha-\beta)z, \alpha, \beta \in \mathbb{R}\big)\right\} \\ &= \{(\alpha, \alpha) + (\beta, 2\beta) + (3 - 3(\alpha+\beta), 1 - (\alpha+\beta))\}\end{aligned}$$

A envoltória afim de três ou mais pontos não colineares (afim-independentes) mas coplanares é o plano em que eles estão. ◀

Definição 3.10 (Combinação convexa)**.** Uma *combinação convexa* de $x_1, x_2, \ldots, x_n$ é uma combinação afim com coeficientes não negativos. ♦

Exemplo 3.11. Sejam x_1, x_2, x_3 elementos quaisquer. Então

$$0.2x_1 + 0.3x_2 + 0.5x_3$$

é combinação convexa de x_1, x_2, x_3. ◀

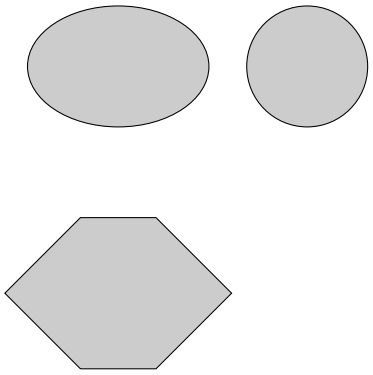
Definição 3.12 (Conjunto convexo)**.** $X \subset \mathbb{R}^n$ é *convexo* se $\forall x, y \in X$, todas as combinações convexas de x e y também estão em X. ♦

Alternativamente, podemos dizer que um conjunto S de pontos é convexo se para toda reta r, $S \cap r$ é conexo, ou seja, $S \cap r$ forma um único segmento (ou reta).

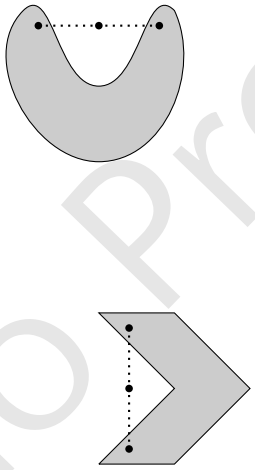
Exemplo 3.13. Qualquer intervalo fechado $[a, b] \subset \mathbb{R}$ é convexo. Suponha que $x, y \in [a, b]$, e $x < y$. Claramente, temos $a \leq x \leq y$. Então para qualquer combinação convexa $z = \lambda x + (1-\lambda)y$, temos $x \leq z \leq y$, que implica que $a \leq z \leq b$, e a combinação $z \in [a, b]$.

Lembramos que os intervalos $(-\infty, b]$ e $[a, \infty)$ são fechados. ◀

Exemplo 3.14. Damos exemplos de conjuntos convexos e não convexos em $\mathbb{R}^2$. São convexos:



Os seguintes conjuntos de pontos não são convexos, e em cada um há a indicação de dois pontos com combinação convexa fora do conjunto.

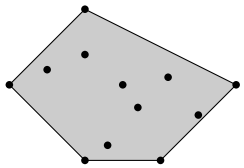


◀

Definição 3.15 (Envoltória Convexa)**.** A *envoltória convexa* de um conjunto $X \subset \mathbb{R}^n$ é o menor subconjunto convexo de $\mathbb{R}^n$ contendo X. Denotamos a envoltória convexa de X por $[X]$. ♦

Note que em muitos textos de Álgebra Linear, usa-se $[X]$ para o subespaço gerado por X (todas as combinações *lineares* de vetores de X – o que é diferente da envoltória convexa).

A figura a seguir ilustra um conjunto de pontos e sua envoltória convexa em $\mathbb{R}^2$.



Teorema 3.16. [X] *é igual ao conjunto de todas as combinações convexas de subconjuntos finitos de* X*.*

Demonstração. Temos $C = \cap Y_i$, onde Y_i são conjuntos convexos contendo X.

C' é o conjunto de combinações convexas finitas de pontos de X.

$(\Leftarrow), C \subset C'$: basta verificar que C' é convexo, porque $X \subset C'$.

Sejam $x, y \in C'$. Então x, y são combinações convexas de pontos de X. Seja $t \in [0, 1]$. Então, $tx + (1 - t)y$ é combinação convexa de pontos de X, e C' é convexo.

$(\Rightarrow), C' \subset C$: realizamos a prova por indução no número de pontos, que denotaremos por m.

A base pode ser verificada trivialmente para $m = 1$ e $m = 2$. Para o passo, consideramos $m \geq 3$.

Nossa hipótese de indução diz que todo ponto em C' que é combinação convexa de $m - 1$ pontos de X também está em todos os conjuntos convexos contendo X (está em C).

Seja $x = t_1x_1 + \ldots + t_mx_m$. (Note que $x \in C'$)

Se $t_m = 1$, então $x = x_m$ e o resultado vale trivialmente.

Se $t_m < 1$, seja então

$$t'_i = \frac{t_i}{1 - t_m}, \quad i = 1, 2, \ldots, m - 1$$

Então temos

$$\begin{aligned} \sum_{i=1}^{m-1} t'_i &= \sum_{i=1}^{m-1} \frac{t_i}{1 - t_m} \\ &= \frac{(\sum_{i=1}^{m} t_i) - t_m}{1 - t_m} \\ &= \frac{1 - t_m}{1 - t_m} \\ &= 1. \end{aligned}$$

Então

$$x' = t'_1x_1 + t'_2x_2 + \ldots + t'_{m-1}x_{m-1}$$

é combinação convexa de $x_1, x_2, \ldots, x_{m-1}$. Pela hipótese de indução, $x' \in C$. Como $x_m \in X$, então $x_m \in C$. Então

$$x = (1 - t_m)x' + t_m x_m$$

é combinação convexa de dois pontos em C e, como C é convexo, $x \in C$. ■

Cada restrição de um programa linear define uma região do espaço que chamamos de *semiespaço*. Definimos a seguir hiperplano e semiespaço.

Definição 3.17 (hiperplano)**.** Seja X um conjunto de pontos $X \subset \mathbb{R}^n$ tal que para todo $x \in X$, existem a_i e b, com pelo menos um a_i diferente de zero,

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n = b.$$

Então X é um *hiperplano* em $\mathbb{R}^n$. ♦

Exemplo 3.18. Retas são hiperplanos em $\mathbb{R}^2$, e planos são hiperplanos em $\mathbb{R}^3$. Similarmente, pontos são hiperplanos em $\mathbb{R}$. ◄

Definição 3.19 (semiespaço)**.** Em $\mathbb{R}^n$, um *semiespaço* é a região de um dos lados de um hiperplano. Em outras palavras, são os pontos **x** tais que

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n \geq b,$$

para determinados $a_i, b \in \mathbb{R}$. ♦

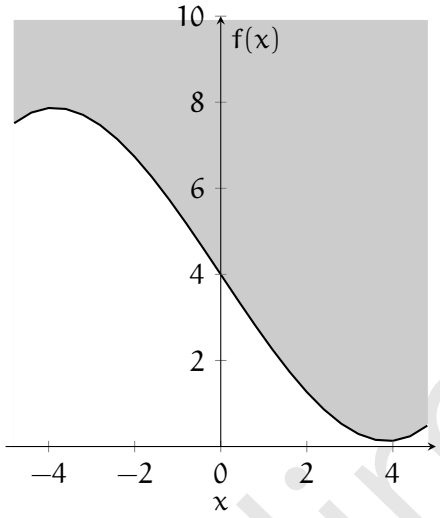
Exemplo 3.20. O conjunto $\{x \in \mathbb{R} : x \leq k\}$ com k constante é o intervalo $(-\infty, k]$, que é um semiespaço de $\mathbb{R}$. ◄

Exemplo 3.21. As soluções para uma desigualdade em $\mathbb{R}^n$ definem um semiespaço. ◄

Semiespaços são convexos: se S é o conjunto de pontos de um dos lados de uma reta, as combinações convexas de pontos de S também estarão naquele mesmo lado da reta.

Definição 3.22 (Epigrafo)**.** Seja $f : \mathbb{R}^n \to \mathbb{R}$ uma função. O *epigrafo* de f é o conjunto de pontos $(x_1, x_2, \ldots, x_n, x_{n+1}) \in \mathbb{R}^{n+1}$ tais que $x_{n+1} \geq f(x_1, x_2, \ldots, x_n)$. ♦

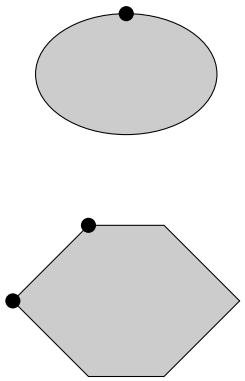
A figura a seguir mostra o epigrafo da função $f(x) = \frac{x^3}{30} - \frac{3}{2}x + 4$.



Definição 3.23 (Ponto extremo de conjunto convexo)**.** Um ponto $x \in S$ convexo é um *ponto extremo* de S se não é combinação convexa de outros pontos de S. ♦

Exemplo 3.24. Qualquer ponto de uma parábola é ponto extremo de seu epigrafo. ◄

Exemplo 3.25. Ilustramos alguns dos pontos extremos (não todos) nos conjuntos a seguir.



◄

Teorema 3.26. *Sejam* $S_1, S_2, \ldots, S_n$ *conjuntos convexos. Então também são convexos* $\cap S_i$, $\sum S_i$ *e* αS_i, $\alpha \in \mathbb{R}$.

Demonstração. Demonstramos apenas o caso da soma. Os outros são pedidos no exercício 13.

Sejam $a, b \in S_i + S_j$. Então $a = a_1 + a_2$ e $b = b_1 + b_2$, com $a_1, b_1 \in S_i$ e $a_2, b_2 \in S_j$. Para $0 \leq \lambda \leq 1$,

$$\begin{aligned}\lambda a + (1-\lambda)b &= \lambda(a_1 + a_2) + (1-\lambda)(b_1 + b_2)\\ &= \lambda a_1 + \lambda a_2 + (1-\lambda)b_1 + (1-\lambda)b_2\\ &= \lambda a_1 + (1-\lambda)b_1 + \lambda a_2 + (1-\lambda)b_2\\ &= [\lambda a_1 + (1-\lambda)b_1] + [\lambda a_2 + (1-\lambda)b_2]\,,\end{aligned}$$

que está em $S_i + S_j$. ■

Definição 3.27 (poliedro)**.** A interseção de um número finito de semiespaços em $\mathbb{R}^n$ é um *poliedro*. ♦

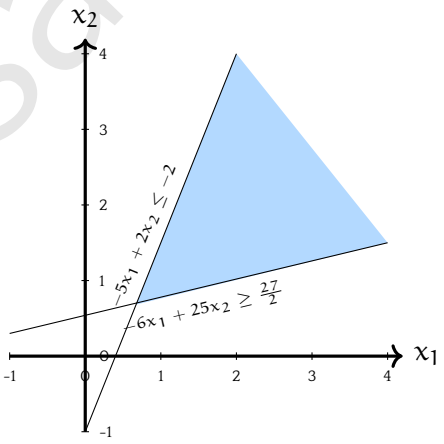
Podemos reescrever esta definição.

Definição 3.28 (poliedro)**.** Um conjunto de pontos $P \in \mathbb{R}^n$ é um poliedro se e somente se pode ser descrito como $P = \{\mathbf{x} \in \mathbb{R}^n : A\mathbf{x} = \mathbf{b}\}$, onde A é uma matriz e $\mathbf{b} \in \mathbb{R}^n$. ♦

Um poliedro pode não ser limitado, como por exemplo o poliedro definido pelas inequações

$$\begin{aligned}-5x_1 + 2x_2 &\leq -2\\ -6x_1 + 25x_2 &\geq \frac{27}{2},\end{aligned}$$

ilustrado na figura a seguir.



Definição 3.29 (Politopo)**.** Um *politopo* é um poliedro limitado. ♦

Exemplo 3.30. Semiespaços e epigrafos de funções não são politopos. Um polígono convexo (triângulo, quadrado, pentágono, etc) em $\mathbb{R}^2$ é um politopo. ◀

A noção de dependência linear estende-se naturalmente para combinações afim e convexas.

Definição 3.31 (afim-independente)**.** Um conjunto de pontos é *afim-independente* quando nenhum deles é combinação afim dos outros. ♦

Exemplo 3.32. Os pontos $(1,1)$ e $(2,3)$ são afim-independentes. Se tentarmos escrever um como combinação afim do outro, teremos

$$(1,1) = \lambda(2,3),$$

o que seria impossível.

Os pontos $(2,2)$ e $(6,6)$ também são afim-independentes, *mesmo sendo linearmente dependentes*: se

$$(6,6) = \lambda(2,2),$$

deveríamos ter $\lambda = 3$, mas para que a combinação fosse afim, λ teria que ser um. ◀

Exemplo 3.33. Os pontos $(1,2)$, $(2,-2)$ e $(5/4,1)$ são afim-dependentes. Para verificar, mostraremos que existem a, b, com $a + b = 1$, tais que

$$(1,2) = a(2,-2) + b(5/4,1).$$

Para obter os coeficientes resolvemos o sistema

$$\begin{aligned} 2a + \frac{5}{4}b &= 1 \\ -2a + b &= 2 \\ a + b &= 1 \end{aligned}$$

O sistema tem solução $a = -1/3$ e $b = 4/3$, e portanto $(1,2)$ é combinação afim de $(2,-2)$ e $(5/4,1)$. ◀

Definição 3.34 (Dimensão de um politopo)**.** Um politopo P tem *dimensão* igual a d se $d+1$ é o maior número de pontos afim-independentes que podem existir em P. ♦

Exemplo 3.35. Um ponto tem dimensão zero, porque é um conjunto de um (0+1) único ponto. Uma reta é um politopo de dimensão um, porque um terceiro ponto sempre pode ser descrito como combinação afim de dois (1+1) outros na mesma reta. Um triângulo tem dimensão dois, porque um quarto ponto pode ser descrito como combinação afim de outros três (2+1) no mesmo plano. ◀

Teorema 3.36. *Um politopo* P *tem* dimensão *igual à dimensão de* $\text{aff}(P)$.

Exemplo 3.37. Sabemos que um triângulo tem dimensão dois. De fato, a envoltória afim do triângulo é $\mathbb{R}^2$, com dimensão dois. ◀

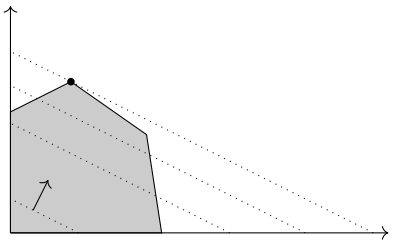
3.2 Soluções viáveis para programas lineares

Teorema 3.38. *O conjunto* S *de soluções viáveis para um programa linear é fechado, convexo e limitado por baixo.*

Demonstração. Pela restrição $\mathbf{x} \geq 0$, S é limitado por baixo. Além disso, S é interseção dos semiespaços definidos pelas restrições do problema e pela restrição de não negatividade. Como os semiespaços são convexos e fechados, S também é. ■

Teorema 3.39. *Seja* S *o conjunto de soluções viáveis para um programa linear. Então, se existe solução ótima para o programa linear, existe um ponto extremo em* S *com o valor ótimo.*

Antes de elaborarmos a demonstração formal deste Teorema, notamos que é intuitivamente simples perceber o que está sendo enunciado. Escolhemos um ponto dentro do poliedro. Traçamos neste ponto um hiperplano ortogonal ao gradiente do objetivo e movemos esse hiperplano na direção do gradiente. Há uma distância máxima que esse movimento pode ser feito sem que o hiperplano fique completamente fora do poliedro. Quando essa distância é máxima, o hiperplano toca o poliedro em um ponto extremo (pode tocar em uma aresta ou face, mas ainda assim ele toca um ponto extremo). A figura a seguir ilustra graficamente esta intuição.



A seguir damos uma demonstração formal desse Teorema.

Demonstração. (do Teorema 3.39).
S tem um número finito de pontos extremos, que denotamos[1] $\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_p^*$.
Seja $\mathbf{x}_0$ um ponto viável maximizando $\mathbf{c}^T\mathbf{x}$ em S:

$$\forall \mathbf{x} \in S, \quad \mathbf{c}^T\mathbf{x_0} \geq \mathbf{c}^T\mathbf{x}.$$

Suponha que $\mathbf{x}_0$ não é ponto extremo. Então $\mathbf{x}_0$ pode ser descrito como combinação convexa dos pontos extremos de S.

$$\mathbf{x}_0 = \sum_{i=1}^{p} \lambda_i \mathbf{x}_i^*, \quad \lambda_i \geq 0, \sum \lambda_i = 1$$

Seja então $\mathbf{x}_r^*$ o ponto extremo com maior valor objetivo. Então,

$$\begin{aligned}
\mathbf{c}^T\mathbf{x}_0 &= \mathbf{c}^T \left(\sum_{i=1}^{p} \lambda_i \mathbf{x}_i^* \right) \\
&= \sum_{i=1}^{p} \lambda_i \left(\mathbf{c}^T \mathbf{x}_i^* \right) \\
&\leq \sum_{i=1}^{p} \lambda_i \left(\mathbf{c}^T \mathbf{x}_r^* \right) \\
&= \mathbf{c}^T \mathbf{x}_r^* \sum_{i=1}^{p} \lambda_i \\
&= \mathbf{c}^T \mathbf{x}_r^*.
\end{aligned}$$

Temos então $\mathbf{c}^T\mathbf{x}_0 \leq \mathbf{c}^T\mathbf{x}_r^*$. Como x_0 é ótimo, $\mathbf{c}^T\mathbf{x}_0 = \mathbf{c}^T\mathbf{x}_r^*$, e existe o ponto extremo x_r^* onde o valor do objetivo é ótimo. ■

Representamos as restrições de um problema na forma matricial como $A\mathbf{x} = \mathbf{b}$. A matriz A normalmente terá m linhas e n colunas, com $n > m$. Presumiremos que o posto da matriz é m – caso não seja, sempre podemos eliminar uma das restrições.

Cada coluna de A representa uma variável. Como a matriz tem posto m, podemos tomar m colunas LI (ou m variáveis), formando uma matriz quadrada B, de ordem m, e resolver o sistema

$$B\mathbf{x}_B = \mathbf{b},$$

[1] Insistimos em mostrar os pontos como vetores!

onde $\mathbf{x}_B$ é o vetor composto pelas variáveis relacionadas às colunas em B.

A matriz B tem posto completo, portanto o sistema sempre terá solução. As variáveis representadas pelas colunas que usamos para formar B são chamadas de *variáveis básicas*, e as outras são as *variáveis não básicas*.

Ao resolver o sistema, determinamos valores para m variáveis. Se fizermos as outras variáveis iguais a zero, teremos uma solução para $A\mathbf{x} = \mathbf{b}$.

Definição 3.40 (solução básica)**.** Seja $A\mathbf{x} = \mathbf{b}$ o conjunto de restrições de um problema de programação linear, onde A tem m linhas e n colunas. Seja B uma matriz formada por m colunas linearmente independentes de A. Então os vetores $\mathbf{x}$ tais que $B\mathbf{x} = \mathbf{b}$ são chamados de *soluções básicas* para o problema. ⧫

Suponha que a matriz A tenha posto m. A matriz B com m colunas LI é uma base para o espaço-coluna de A – daí o nome "solução básica".

Definição 3.41 (solução viável básica)**.** $\mathbf{x} \in \mathbb{R}^n$ é *solução viável básica* para um problema de programação linear se é viável e também básica. ⧫

Exemplo 3.42. Suponha que as restrições de um P.L. sejam

$$\begin{aligned} 2x_1 + 3x_2 &\leq 8 \\ x_1 - 2x_2 &\leq 2 \end{aligned}$$

Na forma padrão, estas duas restrições são

$$\begin{aligned} 2x_1 + 3x_2 + x_3 &= 8 \\ x_1 - 2x_2 + x_4 &= 2 \end{aligned}$$

Estas restrições podem ser descritas como $A\mathbf{x} = \mathbf{b}$, com

$$A = \begin{pmatrix} 2 & 3 & 1 & 0 \\ 1 & -2 & 0 & 1 \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 8 \\ 2 \end{pmatrix}.$$

Como as duas linhas são LI, claramente o espaço coluna de A é igual a $\mathbb{R}^2$.

Tomamos duas colunas LI de A, e obtemos uma base para seu espaço-coluna:

$$B = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}$$

Qualquer solução para

$$\begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_3 \end{pmatrix} = \begin{pmatrix} 8 \\ 2 \end{pmatrix}$$

ou para qualquer outro sistema obtido da mesma forma, com colunas LI de A, é uma *solução básica* para o problema de programação linear. Observe que não usamos x_2. Em uma solução básica *gerada por esta base*, sempre teremos $x_2 = 0$. Se mudarmos a base, x_2 poderá ser positiva. ◀

Teorema 3.43. *Seja S o conjunto de soluções viáveis para um programa linear. Então uma solução viável $\mathbf{x} \in S$ é básica se e somente se é um ponto extremo de S.*

Demonstração. ($\Leftarrow$) Seja $\mathbf{x}$ ponto extremo de S e, sem perda de generalidade, presuma que $x_1, x_2, \ldots, x_k > 0$ e $x_{k+1}, \ldots, x_n = 0$. Mostraremos que as k primeiras colunas formam um conjunto LI, sendo portanto uma solução viável básica.

Denotando a j-ésima coluna de A por $\mathbf{a_j}$, temos

$$\mathbf{a_1}x_1 + \mathbf{a_2}x_2 + \ldots + \mathbf{a_k}x_k = \mathbf{b}. \tag{3.1}$$

Suponha que $\mathbf{a_1}, \ldots, \mathbf{a_k}$ são linearmente dependentes. Então, pela definição de dependência linear,

$$\exists \alpha_1, \alpha_2, \ldots, \alpha_k, \quad \sum \alpha_i \mathbf{a_i} = 0, \tag{3.2}$$

com algum $\alpha_i \neq 0$.

Usaremos os coeficientes desta combinação linear para descrever o ponto xtremo $\mathbf{x}$ como combinação linear de outros pontos (o que seria uma contradição).

Seja $\varepsilon > 0$. Sejam

$$\mathbf{x}^1 = (x_1 + \varepsilon\alpha_1, x_2 + \varepsilon\alpha_2, \ldots, x_k + \varepsilon\alpha_k, 0, 0, \ldots, 0)^T$$
$$\mathbf{x}^2 = (x_1 - \varepsilon\alpha_1, x_2 - \varepsilon\alpha_2, \ldots, x_k - \varepsilon\alpha_k, 0, 0, \ldots, 0)^T$$

Temos $x_j > 0$ para $j \leq k$. Então existe $\varepsilon > 0$ tal que os primeiros k elementos de $\mathbf{x}^1$ e $\mathbf{x}^2$ são maiores que zero:

$$x_j + \varepsilon\alpha_j > 0$$
$$x_j - \varepsilon\alpha_j > 0$$

Pode-se verificar que

$$0 < \varepsilon < \min_j \left\{ \frac{x_j}{|\alpha_j|}, \alpha_j \neq 0 \right\}$$

satisfaz estas condições.

Temos portanto $\mathbf{x}^1$ e $\mathbf{x}^2$ viáveis (ambos maiores que zero). Mas $\mathbf{x} = \frac{1}{2}\mathbf{x}^1 + \frac{1}{2}\mathbf{x}^2$, e $\mathbf{x}$ é combinação convexa de $\mathbf{x}^1$ e $\mathbf{x}^2$ – não pode ser ponto extremo!

Concluímos que nossa hipótese era falsa: $a_1, \ldots, a_k$ é linearmente independente, e a solução é básica.

($\Rightarrow$) Suponha que x^* é uma solução viável básica, com componentes $x_1^*, x_2^*, \ldots, x_m^*, 0, 0, \ldots, 0$. O valor dos x_i^* é dado por

$$\sum_{j \leq m} a_{ij} x_j = b_i, \quad i = 1, 2, \ldots, m. \tag{3.3}$$

Como $a_1, a_2, \ldots, a_m$ é linearmente independente, as soluções são únicas.

Suponha que x^* não é ponto extremo de S. Então deve ser combinação linear de duas soluções viáveis x^1 e x^2, diferentes entre si.

$$\begin{aligned} x_j^* &= \lambda x_j^1 + (1-\lambda) x_j^2, \quad j \leq m \\ 0 &= \lambda x_j^1 + (1-\lambda) x_j^2, \quad j > m \end{aligned}$$

para $0 < \lambda < 1$.

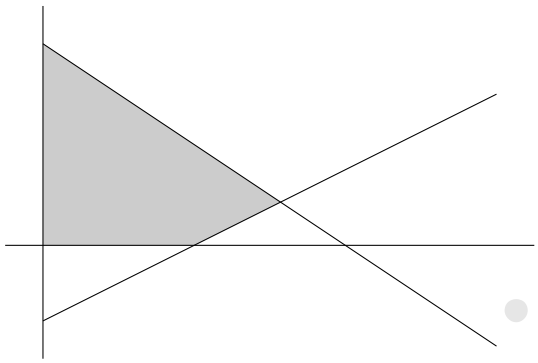
Como $x_j^1, x_j^2 \geq 0$ então $x_j^1, x_j^2 = 0$ para $j > m$.

Restam os x_j^1 e x_j^2 com $j \leq m$. Como as soluções para o sistema (3.3) são únicas, necessariamente temos $x_j^* = x_j^1 = x_j^2$, mas havíamos presumido $x^1 \neq x^2$.

Concluímos então que x^* não pode ser descrito como combinação linear de pontos de S – é um ponto extremo. ■

Exemplo 3.44. Novamente tomamos como exemplo um sistema com restrições $A\mathbf{x} = \mathbf{b}$, e

$$A = \begin{pmatrix} 2 & 3 & 1 & 0 \\ 1 & -2 & 0 & 1 \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 8 \\ 2 \end{pmatrix}.$$



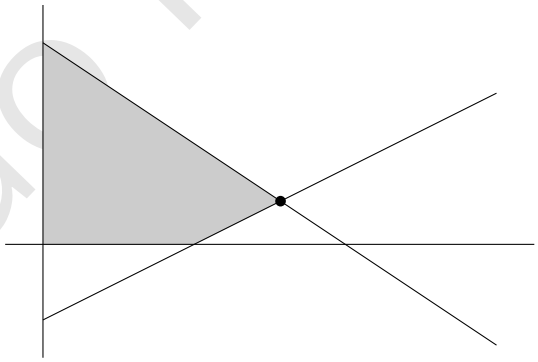
Usando a base $B = \left(\begin{smallmatrix} 2 & 3 \\ 1 & -2 \end{smallmatrix}\right)$,

$$\begin{pmatrix} 2 & 3 \\ 1 & -2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 8 \\ 2 \end{pmatrix}$$

tem solução

$$x_1 = 22/7, \quad x_2 = 4/7,$$

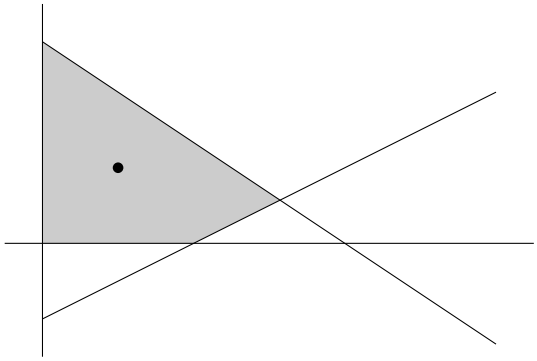
A solução é viável básica, e na figura a seguir podemos observar que está em um ponto extremo.



Observe que uma solução viável que não esteja em um ponto extremo não é básica: seja

$$x_1 = 1, \quad x_2 = 1, \quad x_3 = 3, \quad x_4 = 3.$$

Esta solução é viável, porque todas as variáveis são positivas, e respeitam as restrições.



No entanto, não está em ponto extremo, logo não é básica. ◀

Teorema 3.45. *O conjunto de soluções ótimas para um programa linear é um conjunto convexo.*

Demonstração. Seja K o conjunto de soluções ótimas, $\mathbf{x}_1^*, \mathbf{x}_2^* \in K$, e z^* o valor ótimo da função objetivo. Então,

$$\mathbf{c}^T\mathbf{x}_1^* = \mathbf{c}^T\mathbf{x}_2^* = z^*.$$

Como são viáveis, $\mathbf{x}_1^*, \mathbf{x}_2^* \in S$ e S é convexo, portanto

$$\lambda\mathbf{x}_1^* + (1-\lambda)\mathbf{x}_2^* \in S, \quad 0 \leq \lambda \leq 1.$$

Temos então

$$\begin{aligned}\mathbf{c}^T(\mathbf{x}_1^*, \mathbf{x}_2^*) &= \lambda\mathbf{c}^T\mathbf{x}_1^* + (1-\lambda)\mathbf{c}^T\mathbf{x}_2^* \\ &= \lambda z^* + (1-\lambda)z^* = z^*.\end{aligned}$$

Assim, $\lambda\mathbf{x}_1^* + (1-\lambda)\mathbf{x}_2^* \in K$, para $0 \leq \lambda \leq 1$, e K é convexo. ■

Corolário 3.46. *Se há mais de uma solução ótima para um programa linear, há uma quantidade infinita e não enumerável delas.*

Claramente, o conjunto de soluções ótimas contém um único ponto se for um ponto extremo isolado, ou infinitos pontos, se for a envoltória convexa de alguns pontos extremos.

Teorema 3.47. *Se existe solução viável para um programa linear, então também existe solução viável básica.*

Demonstração. Sejam $\mathbf{a_1}, \mathbf{a_2}, \ldots, \mathbf{a_n}$ as colunas de A, e $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ viável:

$$x_1\mathbf{a_1} + x_2\mathbf{a_2} + \ldots = \mathbf{b}.$$

Suponha, sem perda de generalidade, que as k primeiras variáveis de $\mathbf{x}$, são maiores que zero, e portanto $x_1\mathbf{a_1} + x_2\mathbf{a_2} + \ldots + x_k\mathbf{a_k} = \mathbf{b}$.

Trataremos dois casos: no primeiro, $\mathbf{a_1}, \ldots, \mathbf{a_k}$ são linearmente independente. No segundo, são linearmente dependentes.

Primeiro caso (colunas L.I.): nesta situação, $k \leq m$, porque A tem posto completo.

- Se $k = m$ a solução é, por definição, básica.
- Se $k < m$, então $m - k$ colunas podem ser obtidas de A para formar uma matriz $m \times m$ com colunas $a_1, \ldots, a_k, \ldots, a_m$, todas L.I.

Segundo caso (colunas L.D.): se $a_1, \ldots, a_k$ são L.D., então existem $\alpha_1, \ldots, \alpha_k$, com pelo menos um $\alpha_j > 0$ tais que

$$\sum \alpha_j \mathbf{a_j} = 0$$

Multiplicamos a equação por ε:

$$\sum_{j=1}^{k} \varepsilon\alpha_j \mathbf{a_j} = 0 \tag{3.4}$$

Temos também, da definição do programa linear, que

$$\sum_{j=1}^{k} x_j \mathbf{a_j} = \mathbf{b} \tag{3.5}$$

Calculamos agora (3.5) − (3.4):

$$\sum_{j=1}^{k} x_j \mathbf{a_j} - \varepsilon\alpha_j \mathbf{a_j} = \mathbf{b}$$

$$\sum_{j=1}^{k} \mathbf{a_j} (x_j - \varepsilon\alpha_j) = \mathbf{b}$$

e portanto os $(x_j - \varepsilon\alpha_j)$ formam uma solução para o sistema $A\mathbf{x} = \mathbf{b}$.

Para garantir que a solução é viável escolhemos

$$\varepsilon = \min_j \left\{ \frac{x_i}{\alpha_i}, \alpha_i > 0 \right\}$$

Seja p o índice do mínimo definido acima, de forma que $\varepsilon = x_p/\alpha_p$. Para p,

$$(x_p - \varepsilon\alpha_p) = x_p - \frac{x_p\alpha_p}{\alpha_p} = 0$$

Para $i \neq p$,

$$(x_i - \varepsilon\alpha_i) = x_i - \frac{x_p\alpha_i}{\alpha_p}.$$

Como $x_i > 0$ e $\frac{x_p\alpha_i}{\alpha_p} \leq x_i$, temos $x_i \geq 0$ (não teremos uma solução inviável).

A variável de índice p torna-se zero, e a nova solução

$$x' = (x_1 - \varepsilon\alpha_1, \ldots, x_{k-1} - \varepsilon\alpha_{k-1}, 0, \ldots, 0)$$

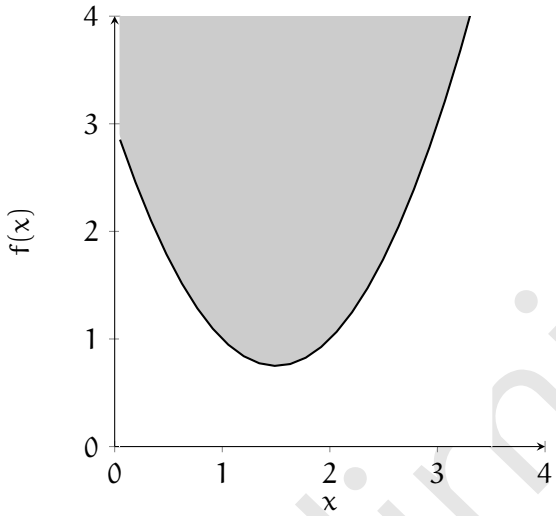
tem então no máximo $k-1$ variáveis positivas. Se as colunas associadas com estas $k-1$ variáveis ainda forem L.D. repetimos a operação; senão, o primeiro caso se aplica. ■

3.3 Funções Convexas

Nesta seção tratamos brevemente de funções convexas, que são importantes para o desenvolvimento da teoria de otimização não linear.

Definição 3.48 (Função convexa)**.** Uma função é convexa se seu epigrafo é convexo. Uma função f é côncava se $-f$ é convexa. ♦

A figura a seguir mostra o epigrafo da função convexa $f(x) = x^2 - 3x + 3$.



Teorema 3.49. *Se* $f : \mathbb{R}^n \to \mathbb{R}$ *é convexa e* $\mathrm{Dom}(f)$ *é convexo, então*

$$\lambda f(x) + (1-\lambda)f(y) \geq f(\lambda x + (1-\lambda)y)$$

para todos $x, y \in \mathrm{Dom}(f)$ *e todo* $\lambda \in [0,1]$*.*

Teorema 3.50. *Se* $f_1, f_2, \ldots, f_m : \mathbb{R}^n \to \mathbb{R}$ *são convexas, então também são convexas*

$$g(x) = \max\{f_i(x)\}$$
$$h(x) = \sum f_i(x)$$

O teorema 3.51 nos dá uma segunda caracterização de funções convexas.

Teorema 3.51. *Uma função* f *com domínio* $D \subseteq \mathbb{R}^n$ *é convexa se e somente se, para todos* $x_1, x_2 \in D$*, e para todo* $\lambda \in [0,1]$*,*

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda)f(x_2).$$

Definição 3.52. Seja A uma matriz quadrada simétrica. A é *semidefinida positiva* se $\mathbf{x}^T A\mathbf{x} \geq 0$ para todo $\mathbf{x}$. A é *definida positiva* se $\mathbf{x}^T A\mathbf{x} \geq 0$ para todo $\mathbf{x} \neq 0$. E A é indefinida se $\mathbf{x}^T A\mathbf{x}$ assume valores positivos e negativos. ♦

Teorema 3.53. *Uma matriz quadrada é semidefinida positiva se todos seus autovalores são não-negativos.*

Exemplo 3.54. A matriz

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 6 \end{pmatrix}$$

é semidefinida positiva. Observamos que

$$\mathbf{x}A\mathbf{x} = 6x_3^2 + 2x_2^2 + x_1^2 + 6x_2x_3 + 2x_1x_3 + 2x_1x_2$$

Como não é imediatamente óbvio que este valor é sempre positivo, verificamos os autovalores da matriz, que são

$$\begin{aligned} \lambda_1 &= 4 - \sqrt{15} \\ \lambda_2 &= 4 + \sqrt{15} \\ \lambda_3 &= 1 \end{aligned}$$

◀

Exemplo 3.55. Seja

$$A = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}.$$

Tomamos agora dois vetores,

$$\begin{aligned} \mathbf{x} &= \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \\ \mathbf{y} &= \begin{pmatrix} 1 \\ -1 \end{pmatrix}. \end{aligned}$$

Como

$$\mathbf{x}A\mathbf{x} = 6, \quad \mathbf{y}A\mathbf{y} = -2,$$

a matriz A é indefinida. ◀

Definição 3.56 (Matriz Hessiana)**.** Seja $f : \mathbb{R}^n \to \mathbb{R}$ duas vezes diferenciável. A matriz Hessiana de f em x, denotada por $H_f(x)$ ou $\nabla^2 f(x)$ é a forma bilinear

$$\nabla^2(\mathbf{x})_{i,j} = \frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j},$$

ou

$$\nabla^2 f(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2}(\mathbf{x}) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(\mathbf{x}) & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(\mathbf{x}) \\ \frac{\partial^2 f}{\partial x_2 \partial x_1}(\mathbf{x}) & \frac{\partial^2 f}{\partial x_2^2}(\mathbf{x}) & \dots & \frac{\partial^2 f}{\partial x_2 \partial x_n}(\mathbf{x}) \\ \vdots & \vdots & \ddots & \\ \frac{\partial^2 f}{\partial x_n \partial x_1}(\mathbf{x}) & \frac{\partial^2 f}{\partial x_n \partial x_2}(\mathbf{x}) & \dots & \frac{\partial^2 f}{\partial x_n^2}(\mathbf{x}) \end{pmatrix}.$$

♦

Exemplo 3.57. Seja $F; \mathbb{R}^2 \to \mathbb{R}$, tal que $f(x,y) = \cos(y) + 2x^2y$. A Hessiana de f é

$$\nabla^2 f = \begin{pmatrix} 4y & 4x \\ 4x & -\cos(y) \end{pmatrix}. \qquad \blacktriangleleft$$

Lema 3.58. *Seja S um subconjunto não vazio, aberto e convexo de* $\mathbb{R}^n$. *Seja* $f : S \to \mathbb{R}$ *diferenciável. Então* f *é convexa se e somente se, para todos* $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$,

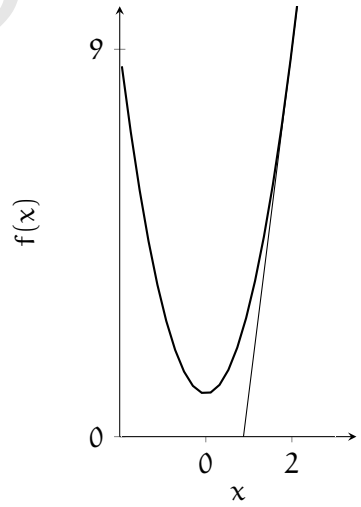
$$f(\mathbf{x}') \geq f(\mathbf{x}) + \nabla(f(\mathbf{x}))^T(\mathbf{x}' - \mathbf{x}).$$

O Lema 3.58 nos diz que f é convexa se e somente se seu gráfico está sempre acima de qualquer plano tangente a ele – ou seja, que seu epigrafo é convexo.

Exemplo 3.59. A função $f(x) = 2x^2 + 1$ é convexa, já que sua Hessiana é $\nabla^2 f(x) = (4)$. Verificamos que, de fato,

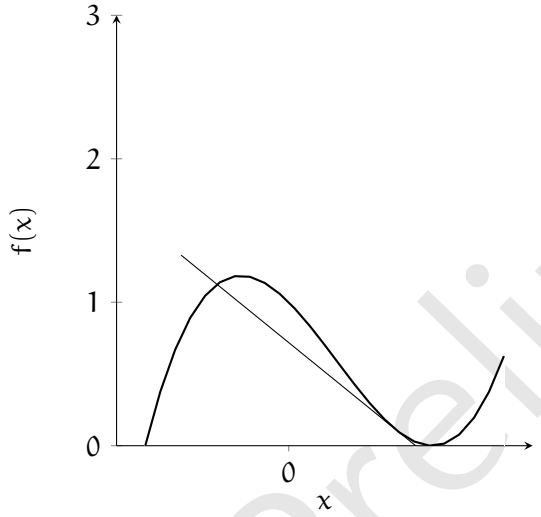
$$\begin{aligned} f(x) + \nabla(f(x))^T(x' - x) &= f(x) + 4x(x' - x) \\ &= 2x^2 + (4x)(x' - x) \\ &= 2x^2 + 4x(x' - x) \\ &= 2x^2 + 4xx' - 4x^2 \\ &\leq 2(x')^2 \end{aligned}$$

Além de verificar algebricamente, observamos o gráfico de $f(x)$ e percebemos que realmente, os planos tangentes sempre ficam abaixo do gráfico.



◀

Exemplo 3.60. A função $f(x) = x^3 - x^2 - x + 1$ não é convexa. Vemos que o plano tangente não fica sempre acima de todos os pontos do gráfico, como é possível perceber na figura a seguir.



◀

Teorema 3.61. *Seja* $f : D \to \mathbb{R}$*, com* $D \subset \mathbb{R}^n$ *não vazio aberto e convexo, e* f *contínua e duas vezes diferenciável.* f *é convexa se e somente se* $\nabla^2 f(x)$ *é semidefinida positiva para todo* $x \in \mathrm{Dom}(f)$*.*

Exemplo 3.62. Seja $f(a, b) = a^4 + b^2$. Temos

$$\nabla^2 f(a, b) = \begin{pmatrix} 12a^2 & 0 \\ 0 & 2 \end{pmatrix}$$

Já podemos perceber que os autovalores da Hessiana são a constante dois e $12a^2$, que são sempre positivos, e a função é convexa.

Podemos também verificar que $\mathbf{x}^T(\nabla^2 f(a, b))\mathbf{x}$ é sempre positivo. Para qualquer $\mathbf{x} \in \mathbb{R}^2$,

$$\mathbf{x}^T \nabla^2 f(a, b)\mathbf{x} = \begin{pmatrix} 12a^2 x_1 & 2x_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 12a^2 x_1^2 + 2x_2^2 > 0,$$

portanto f é convexa em $\mathbb{R}^2$. ◀

Exemplo 3.63. Seja $f(a,b) = e^a + e^b$. Então

$$\nabla^2 f(a,b) = \begin{pmatrix} e^a & 0 \\ 0 & e^b \end{pmatrix}$$

Novamente notamos que os autovalores são positivos – a função é convexa.

Para qualquer $\mathbf{x} \in \mathbb{R}^2$,

$$\mathbf{x}^T \nabla^2 f(a,b)\mathbf{x} = \begin{pmatrix} x_1 e^a & x_2 e^b \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = x_1^2 e^a + x_2^2 e^b > 0,$$

portanto f é convexa em $\mathbb{R}^2$.

Poderíamos também ter observado que f é a soma de e^a com e^b, duas funções convexas – e a soma de funções convexas sempre é convexa. ◀

Exemplo 3.64. Seja $f(a,b) = (a+b)^2$. Então

$$\nabla^2 f(a,b) = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}$$

Para qualquer $\mathbf{x} \in \mathbb{R}^2$,

$$\mathbf{x}^T \nabla^2 f(a,b)\mathbf{x} = \begin{pmatrix} 2x_1 + 2x_2 & 2x_1 + 2x_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 2x_1^2 + 2x_2^2 + 4x_1x_2.$$

A função quadrática $2x_1^2 + 2x_2^2 + 4x_1x_2$ é sempre positiva, portanto f é convexa em $\mathbb{R}^2$.

Os autovalores da Hessiana são 0 e 4. ◀

Exemplo 3.65. Seja $f(a,b) = a^3 + b^2$. Construímos a Hessiana no ponto (a,b),

$$\nabla^2 f(a,b) = \begin{pmatrix} 6a & 0 \\ 0 & 2 \end{pmatrix}$$

Os autovalores são $6a$ e 2. Quando a for menor que zero, o primeiro autovalor será negativo, e f não é convexa.

No entanto, podemos dizer que f é convexa se tiver seu domínio restrito a $a \geq 0$. ◀

Notas

Normalmente livros abordando Otimização Convexa (inclusive Programação Linear) tratam de Análise Convexa e sua relação com otimização. Por exemplo, os livros de Sinha [Sin06], Matousek [MG06] e Luenberger [Lue10] tratam do assunto.

O Teorema 3.61 é demonstrado em diversos livros de otimização não linear – por exemplo nos livros de Mokhtar Bazaraa, Hanif Sherali e C. M. Shetty [BSS06] e de Andrzej Ruszczyski [Rus06].

Os livros de Alexander Barvinok [Bar02] e de Steven Lay [Lay07] abordam detalhadamente o tópico de Convexidade. O livro de Dimitri Bertsekas e Angelia Nedic [BN03] também aborda convexidade e sua relação com otimização. Uma discussão em maior densidade de conjuntos e funções convexas é encontrada no livro de Rockafellar [Roc96].

Exercícios

Ex. 11 — Mostre como construir um poliedro com $\binom{n-k}{m}$ vértices, para qualquer $1 \leq k \leq n$ e quaisquer m, n, com $n \geq m$, descrevendo o poliedro como interseção de semiespaços.

Ex. 12 — Dissemos que um conjunto S de pontos é convexo se e somente se para toda reta r, $S \cap r$ é um único segmento de reta (ou seja, conexo). Demonstre que esta definição é equivalente à definição 3.12.

Ex. 13 — Demonstre o que falta do Teorema 3.26.

Ex. 14 — Mostre que para quaisquer $a, b, c, \alpha, \beta, \gamma$, com $a, \alpha > 0$, a interseção dos conjuntos $\{(x, y) : y < -ax^2 + bx + c\}$ e $\{(x, y) : y > \alpha x^2 + \beta x + \gamma\}$ em $\mathbb{R}^2$ é convexa.

Ex. 15 — Sejam a e b dois pontos. O conjunto de pontos que *não* são mais próximos de b do que a é $\{x | \operatorname{dist}(x, a) \leq \operatorname{dist}(x, b)\}$. Este conjunto é convexo?

Ex. 16 — Determine precisamente quando um subconjunto de $\mathbb{R}^2$ definido por um conjunto de inequações quadráticas $\{(x, y) : y \leq ax^2 + bx + c\}$ é convexo. Generalize para mais que duas dimensões.

Ex. 17 — Prove o lema 3.58.

Ex. 18 — Mostre que o conjunto de todas as matrizes semidefinidas posi-

tivas é convexo.

Ex. 19 — Mostre que uma função f em $\mathbb{R}^n$ é linear se e somente se f é convexa e côncava.

Ex. 20 — Mostre que nenhuma função polinomial de grau ímpar maior ou igual a 3 e em uma variável é convexa.

Ex. 21 — Determine se são convexas:

i) $f(a, b) = e^a + \log(b)$

ii) $g(a, b) = e^a - \log(b)$

iii) $h(a, b) = e^a e^b$

iv) $j(a, b) = \log(a)\log(b)$

Ex. 22 — Prove que $f(\mathbf{x}) = x_1^2/x_2$ é convexa em $\mathbb{R}^2$ onde $x_2 > 0$.

Ex. 23 — Identifique subdomínios onde as funções são convexas:

i) $f(x, y) = \operatorname{sen}(x) + \cos(y)$

ii) $g(x, y) = \operatorname{sen}(x)\cos(y)$

iii) $h(x) = x\log(x)$

iv) $j(x, y, z) = e^x - z\log(y)$

v) $k(x, y, z) = e^x - \log(xy) + z^2$

Ex. 24 — Prove o Teorema 3.49.

Ex. 25 — Prove o Teorema 3.50.

Ex. 26 — Suponha que queiramos otimizar uma função linear, sujeita a restrições *estritamente não lineares*. Se soubermos que a região viável é convexa, quantas soluções ótimas podemos ter, no máximo? E se a região viável não for convexa?

Ex. 27 — Suponha que saibamos que um problema de programação linear é viável e limitado, e portanto tem solução ótima. Sem tentar resolvê-lo, há como verificar se ele tem somente uma ou infinitas soluções ótimas?

Ex. 28 — Seja $\mathbb{R}[x]$ o conjunto de todos os polinômios em x com coeficientes reais. Seja S o subconjunto de $\mathbb{R}[x]$ que são positivos no intervalo $[0, 1]$. Mostre que S é convexo.

Capítulo 4

O Método Simplex

Neste Capítulo tratamos do método mais conhecido para resolver problemas de Programação Linear – o método Simplex.

4.1 Exemplo inicial

Introduzimos o método com um exemplo. Considere o problema de programação linear:

$$\max x_1 + 2x_2$$

sujeito a:

$$\begin{aligned} x_1 + x_2 &\leq 10 \\ -2x_1 + \frac{3}{4}x_2 &\leq 5 \\ x_1 - x_2 &\leq -3 \end{aligned}$$

Transformamos o problema para que as restrições fiquem na forma de igualdades:

$$\begin{array}{rrrrrcr} x_1 & +x_2 & +x_3 & & & = & 10 \\ -2x_1 & +\frac{3}{4}x_2 & & +x_4 & & = & 5 \\ x_1 & -x_2 & & & +x_5 & = & -3 \end{array}$$

e usaremos agora alguns passos do algoritmo Simplex.

Temos cinco variáveis, e queremos inicialmente uma solução básica viável. Olhando para o problema, notamos que podemos usar $x_3 = 10$, $x_4 = 5$, e $x_5 = -3$, com as outras variáveis iguais a zero. Temos então três variáveis não zero, e a solução é claramente viável. Temos uma solução

viável básica. Note que os coeficientes das variáveis não zero é um, e os manteremos assim durante toda a execução do algoritmo Simplex.

Representaremos o sistema de equações na forma de matriz:

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & \mathbf{b} \\ 1 & 1 & 1 & 0 & 0 & 10 \\ -2 & 3/4 & 0 & 1 & 0 & 5 \\ 1 & -1 & 0 & 0 & 1 & -3 \end{pmatrix}$$

e podemos então reescrever as variáveis da forma como quisermos simplesmente usando operações elementares, semelhante aos passos de eliminação de Gauss, exceto que nosso objetivo não é transformar a matriz em forma triangular. *O objetivo é escrever as variáveis básicas como função das não básicas:* No tableau anterior a segunda linha, por exemplo, significa

$$-2x_1 + \frac{3}{4}x_2 + 0x_3 + x_4 + 0x_5 = 5.$$

Isolando x_4, *que é a única variável básica com coeficiente diferente de zero nesta linha*,

$$x_4 = 5 - 2x_1 - \frac{3}{4}x_2$$

Neste momento, a solução atual dá o valor $5 - 0 - 0 = 0$ para x_4, mas a representação da linha nesta forma nos ajuda a compreender o que acontecerá quando dermos valores positivos a x_1 e x_2.

Todo tableau Simplex pode ser lido, portanto, como uma expressão das variáveis básicas (ou da solução atual) em função das não-básicas.

Agora introduzimos x_2 na base, retirando x_5. Reescreveremos o sistema de forma que a coluna relativa a x_2 torne-se igual à coluna que agora representa x_5.

Conceitualmente, tomaremos a linha onde tínhamos x_5 representada como função das outras, e isolaremos x_2, que passará a ser então representada como função das não básicas. Para isso, usamos eliminação de Gauss de forma a tornar $a_{32} = 1$ e $a_{12} = a_{22} = 0$. Primeiro, dividimos a linha 3 por $a_{32} = -1$, e obtemos

$$\begin{pmatrix} -1 & 1 & 0 & 0 & -1 & 3 \end{pmatrix}$$

Somamos múltiplos desta linha às outras, para obter zeros no resto da coluna dois (somamos -3/4 da linha 3 à linha 2, e somamos -1 vezes a linha 3 à linha 1):

$$\begin{pmatrix} 2 & 0 & 1 & 0 & 1 & 7 \\ -5/4 & 0 & 0 & 1 & 3/4 & 11/4 \\ -1 & 1 & 0 & 0 & -1 & 3 \end{pmatrix}$$

Temos agora $x_1 = x_5 = 0$, $x_2 = 3$, $x_3 = 7$, e $x_4 = 11/4$. O valor do objetivo para esta solução é $x_1 + 2x_2 = 6$. Representamos também a função objetivo como um vetor coluna $\mathbf{c} = \begin{pmatrix}1 & 2 & 0 & 0 & 0\end{pmatrix}^T$, e a solução $\mathbf{x} = \begin{pmatrix}0 & 3 & 7 & 11/4 & 0\end{pmatrix}^T$. Assim, podemos expressar

$$\mathbf{c}^T\mathbf{x} = \begin{pmatrix}1 & 2 & 0 & 0 & 0\end{pmatrix} \begin{pmatrix} 0 \\ 3 \\ 7 \\ 11/4 \\ 0 \end{pmatrix} = 6.$$

Conseguimos uma solução, com $x_2 = 3$, melhor que a inicial. Se incluirmos x_1 também na base, a solução será ainda melhor. O algoritmo Simplex determina quais variáveis devemos incluir e excluir da base a cada passo de forma que cada solução seja melhor que a anterior, e que todas sejam viáveis (garantindo assim que após um número finito de passos, obtenhamos a solução ótima).

4.2 Formulação

No resto do texto, usaremos uma matriz A, $m \times n$, e vetores coluna $\mathbf{b}$ com m elementos e $\mathbf{c}$ com n elementos, de forma que programas lineares sejam descritos como

$$\begin{aligned} \max\ & \mathbf{c}^T\mathbf{x} \\ \text{s.a.: } & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0, \end{aligned}$$

sendo $\mathbf{x}$ um vetor coluna com n elementos representando as variáveis para as quais queremos obter valores que maximizem $\mathbf{c}^T\mathbf{x}$.

Será útil dividir estes objetos (matriz A e vetores $\mathbf{c}$ e $\mathbf{x}$) em duas partes: as que se referem às variáveis da base ($A_B, \mathbf{c}_B, \mathbf{x}_B$) e as que se referem às variáveis fora da base ($A_N, \mathbf{c}_N, \mathbf{x}_N$). Presumimos que as colunas de A e os elementos de $\mathbf{x}$ e $\mathbf{c}$ sempre são reordenados de forma que as variáveis da base estejam à esquerda das outras – isso nos garante que a base é representada por uma submatriz $m \times m$ no lado esquerdo de A (isso apenas facilita a exposição do assunto, não interferindo em nada mais):

$$\begin{aligned} A &= (A_B, A_N) \\ \mathbf{c}^T &= (\mathbf{c}_B^T, \mathbf{c}_N^T) \\ \mathbf{x} &= (\mathbf{x}_B, \mathbf{x}_N) \end{aligned}$$

Isto significa que a matriz A_N e os vetores $\mathbf{c}_N$ e $\mathbf{x_N}$ tem índices iniciando em $m+1$:

$$\begin{aligned} A_N &= \left(\mathbf{a_{m+1}}, \mathbf{a_{m+2}}, \ldots, \mathbf{a_n}\right), \\ \mathbf{c}_N^T &= (c_{m+1}, c_{m+2}, \ldots, c_n), \\ \mathbf{x}_N &= (x_{m+1}, x_{m+2}, \ldots, x_n). \end{aligned}$$

4.3 Intuição geométrica

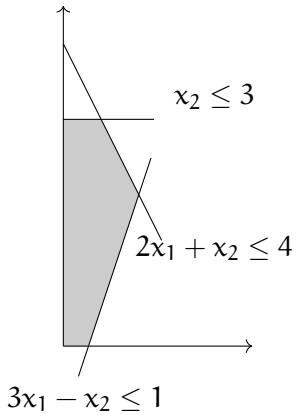
No capítulo 3, verificamos que as soluções viáveis básicas para um problema de programação linear são os pontos extremos do poliedro definido pelas restrições. Nas próximas seções desenvolveremos o método Simplex, que verifica uma sequência de pontos extremos (ou de soluções viáveis básicas) para o problema até chegar à solução ótima.

Damos aqui uma intuição simples de como é o funcionamento do Simplex: Saindo de alguma solução viável básica (ponto extremo), o Simplex procura outra solução que possa obter trocando uma única coluna da base (portanto algum ponto extremos adjacente ao atual), e *que ofereça vantagem*, ou seja, cujo valor objetivo seja melhor que o da solução atual. Prossegue assim até o momento em que não haja pontos extremos adjacentes com valor melhor que o atual – que portanto deve ser ótimo.

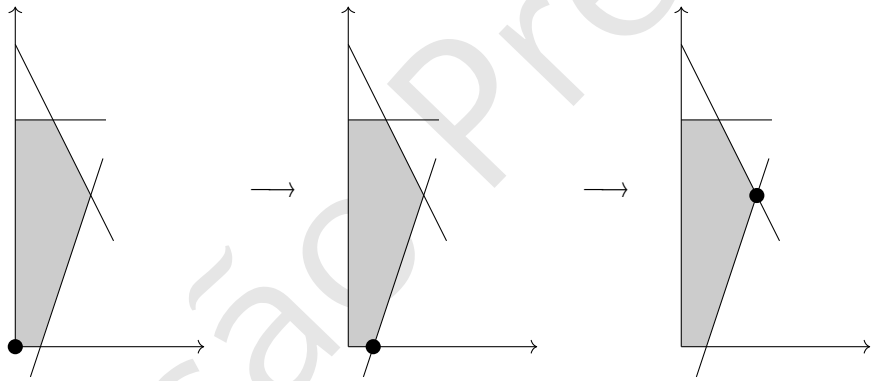
Exemplo 4.1. Ilustraremos esta idéia com o seguinte problema com duas variáveis.

$$\begin{aligned} \max\ & 2x_1 + x_2 \\ \text{s.a.:}\ & x_2 \leq 3 \\ & 2x_1 + x_2 \leq 4 \\ & 3x_1 - x_2 \leq 1 \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

A representação gráfica da região viável deste problema é dada a seguir.



Começamos com a solução viável básica $x_1 = 0, x_2 = 0$ (ou seja, $\mathbf{x} = \mathbf{0}$, e mudamos uma coluna da base por vez, primeiro para $(1/3, 0)$ e depois para a solução ótima $(1, 2)$.



Tínhamos inicialmente uma base com variáveis de folga. Na primeira mudança, de $(0, 0)$ para $(1/3, 0)$, retiramos a folga da restrição $3x_1 - x_2 \leq 1$, e determinamos para x_1 o maior valor possível respeitando a restrição: chegamos a uma nova base, onde a folga desta restrição é zero, e o valor de x_1 é $1/3$. Depois retiramos a folga da restrição $2x_1 + x_2 \leq 4$, e aumentamos o valor de x_2, chegando à solução ótima $(1, 2)$.

As derivadas parciais do objetivo no ponto $(1, 2)$ e na direção de seus pontos extremos vizinhos são todas negativas, portanto não há como melhorar a solução. ◀

4.4 Coeficientes reduzidos de custo

Embora possamos facilmente identificar a restrição (e consequentemente a coluna) que deve entrar na base no próximo passo do Simplex, esta intuição visual não é suficiente para implementar o algoritmo – que deve funcionar para dimensões arbitrárias. Para cada variável não básica em um tableau Simplex existe um número que usaremos para decidir se ela deve ou não entrar na base. Daremos duas definições diferentes (mas equivalentes) para estes números, chamados de *coeficientes reduzidos de custo*.

4.4.1 Primeira definição

Nossa primeira definição para coeficiente reduzido de custo é como a derivada do objetivo na direção de cada variável não básica, porque com elas poderemos determinar exatamente quais variáveis melhorarão o objetivo se as incluirmos na base.

O valor do objetivo usando apenas as variáveis básicas é dado por

$$z_0 = \mathbf{c}_B^T \mathbf{x}_B.$$

Agora partimos do sistema $A\mathbf{x} = \mathbf{b}$:

$$\begin{aligned} A\mathbf{x} &= \mathbf{b} \\ A_B\mathbf{x}_B + A_N\mathbf{x}_N &= \mathbf{b} \end{aligned}$$

Como estamos considerando $A_B = I$,

$$\begin{aligned} \mathbf{x}_B &= \mathbf{b} - A_N\mathbf{x}_N \\ \mathbf{c}_B^T\mathbf{x}_B &= \mathbf{c}_B^T\mathbf{b} - \mathbf{c}_B^T A_N\mathbf{x}_N \end{aligned}$$

Seja $\mathbf{z} = \mathbf{c}_B^T A_N$, de forma que $z_j = c_1a_{1j} + c_2a_{2j} + \ldots + c_na_{mj}$, para $j > m$ (lembre-se que os índices das colunas de A_N começam com $m+1$).

$$\mathbf{c}_B^T A_N = (c_1, c_2, \ldots, c_m) \begin{pmatrix} a_{1,m+1} & a_{1,m+2}, & \cdots & a_{1,n} \\ \vdots & \ddots & & \\ a_{m,m+1} & & & a_{m,n} \end{pmatrix} = \begin{pmatrix} z_{m+1} \\ z_{m+2} \\ \vdots \\ z_n \end{pmatrix},$$

onde $z_j = \sum_{i \leq m} c_i a_{ij}$. Indexamos $\mathbf{z}$ a partir de $m+1$, porque também tem $n-m$ elementos (um para cada variável não básica).

Então,

$$\begin{aligned}\mathbf{c}_B^T\mathbf{x}_B &= \mathbf{c}_B^T\mathbf{b} - \mathbf{z}\mathbf{x}_N\\ \mathbf{c}_B^T\mathbf{x}_B &= z_0 - \mathbf{z}\mathbf{x}_N,\end{aligned}$$

porque $\mathbf{z}\mathbf{x}_N = 0$ (o vetor $\mathbf{x}_N$, com variáveis não básicas, vale zero). Mesmo que esta expressão valha zero, a manteremos ali, porque posteriormente ela será importante quando quisermos saber o que acontece quando mudamos o valor de alguma variável não básica. Assim,

$$\sum_{j\leq m} c_j x_j = z_0 - \sum_{j>m} z_j x_j.$$

Temos somente as variáveis básicas no lado esquerdo. Se somarmos as não básicas obteremos $\mathbf{c}^T\mathbf{x}$. Assim, somamos $\sum_{j>m} c_j x_j$ aos dois lados da equação, obtendo

$$\begin{aligned}\mathbf{c}^T\mathbf{x} &= z_0 + \sum_{j>m} c_j x_j - \sum_{j>m} z_j x_j\\ &= z_0 + \sum_{j>m} (c_j - z_j)x_j\end{aligned}$$

Podemos então descrever o valor da função objetivo usando apenas z_0 (o valor calculado usando as variáveis básicas) e os valores das variáveis não básicas:

$$\mathbf{c}^T\mathbf{x} = z_0 + \sum_{j>m} (c_j - z_j)x_j$$

Esta equação é importante porque mostra o quanto o valor da função objetivo aumenta se aumentarmos o valor de uma das variáveis não básicas: *se aumentarmos uma variável não básica* x_j *de zero para* k*, o objetivo aumentará em* $k(c_j - z_j)$. Esta informação será usada para determinar que variáveis devem ter seu valor aumentado pelo algoritmo Simplex.

Observe também que

$$\frac{\partial \mathbf{c}^T\mathbf{x}}{\partial x_j} = c_j - z_j,$$

e que os valores de $(c_j - z_j)$ nos dizem as direções em que podemos melhorar o valor da função objetivo.

Definição 4.2 (Coeficiente reduzido de custo)**.** Em um programa linear, os valores

$$r_j = \frac{\partial \mathbf{c}^T \mathbf{x}}{\partial x_j} = c_j - z_j$$

são chamados de *coeficientes reduzidos (ou "relativos") de custo*, e determinam em quanto cada variável não básica melhora o objetivo quando seu valor é aumentado em uma unidade. ♦

4.4.2 Segunda definição

A segunda definição de coeficiente reduzido de custo (equivalente à primeira) é como a diferença de valor que uma coluna tem na base e fora dela (ou seja, quanto ganhamos se a incluirmos na base).

Também podemos definir o coeficiente reduzido de custo de outra maneira. Chamamos A_B de "base" porque ela é de fato base para o espaço coluna de A. Qualquer coluna de A pode, portanto, ser escrita como combinação linear das colunas de A_B. Podemos facilmente calcular o valor de uma variável quando ela está na base – basta olhar para seu coeficiente c_j no objetivo.

Quando a variável não está na base, podemos calcular o valor de sua coluna se a escrevermos como combinação linear da base, usandos os valores das colunas básicas. Por exemplo, suponha que uma coluna $\mathbf{a_j}$ esteja fora da base. Então

$$\begin{aligned}\mathbf{a_j} &= \alpha_1 \mathbf{a_1} + \cdots + \alpha_1 \mathbf{a_m} \\ &= \alpha_1 \begin{pmatrix} a_{11} \\ \vdots \\ a_{m1} \end{pmatrix} + \cdots + \alpha_m \begin{pmatrix} a_{1m} \\ \vdots \\ a_{mm} \end{pmatrix}\end{aligned}$$

Para obter o valor da coluna n básica $\mathbf{a_j}$, usamos sua expressão como combinação linear da base, e multiplicamos cada elemento por seu custo c_j:

$$\begin{aligned}&c_1\alpha_1 \mathbf{a_1} + \cdots + c_m \alpha_1 \mathbf{a_m} \\ &= c_1\alpha_1 \begin{pmatrix} a_{11} \\ \vdots \\ a_{m1} \end{pmatrix} + \cdots + c_m\alpha_m \begin{pmatrix} a_{1m} \\ \vdots \\ a_{mm} \end{pmatrix} \\ &= \mathbf{c}_B^T A_B.\end{aligned}$$

Este é o valor de *uma coluna de* A_N. Quando fazemos o mesmo com todas as colunas, temos

$$\mathbf{z} = \mathbf{c}_B^T A_B A_N,$$

e se mantivermos a base representada como identidade, temos

$$\mathbf{z} = \mathbf{c}_B^T A_N.$$

Assim, enquanto c_j nos dá o valor de uma variável quando ela está na base, z_j nos dá seu valor quando a construímos sinteticamente quando ela não está na base. O coeficiente reduzido de custo, $r_j = c_j - z_j$ é a diferença entre os dois, e nos informa quanto poderemos ganhar incluindo a variável na base. Claramente, quando todos os r_j forem negativos, não há como melhorar a solução.

4.4.3 Representação no tableau

Adicionamos a equação que dá o valor do objetivo ao sistema $A\mathbf{x} = \mathbf{b}$:

$$\begin{array}{rcrcrcrcrcr}
a_{11}x_1 & + & a_{12}x_2 & + & \dots & + & a_{1n}x_n & & = & & b_1 \\
a_{21}x_1 & + & a_{22}x_2 & + & \dots & + & a_{2n}x_n & & = & & b_2 \\
\vdots & & & & & & & & & & \vdots \\
a_{m1}x_1 & + & a_{m2}x_2 & + & \dots & + & a_{mn}x_n & & = & & b_m \\
c_1x_1 & + & c_2x_2 & + & \dots & + & c_nx_n & -z & = & & -z_0
\end{array}$$

Reescrevendo a última linha usando apenas as variáveis não básicas,

$$z = z_0 + r_{m+1}x_1 + r_{m+2}x_2 + \dots + r_nx_n$$

obtemos

$$\begin{array}{rcrcrcrcrcrcr}
a_{11}x_1 & + & \dots & a_{1m}x_2 & + & a_{1m+1}x_{m+1} & + & \dots & + & a_{1n}x_n & 0 & = & b_1 \\
a_{21}x_1 & + & \dots & a_{2m}x_2 & + & a_{2m+1}x_{m+1} & + & \dots & + & a_{2n}x_n & 0 & = & b_2 \\
\vdots & & & & & & & & & & \vdots & & \\
a_{m1}x_1 & + & \dots & a_{mm}x_2 & + & a_{mm+1}x_{m+1} & + & \dots & + & a_{mn}x_n & 0 & = & b_m \\
0 & + & \dots & 0 & + & r_{m+1}x_{m+1} & + & \dots & + & r_nx_n & -z & = & -z_0
\end{array}$$

onde $r_j = (c_j - z_j)$. A coluna contendo zeros e $-z$ nunca será parte de uma base, e não a representaremos nos tableaux.

4.4.4 Exemplo

Damos agora um exemplo do cálculo dos coeficientes reduzidos de custo.

Exemplo 4.3. Suponha que um programa linear tenha como objetivo maximizar a função $2x_1 + 3x_2 + x_3$. Duas variáveis, x_4 e x_5, foram adicionadas para transformar restrições de desigualdade em igualdade. Suponha que

o seguinte tableau tenha sido encontrado durante a execução do método Simplex:

$$\left(\begin{array}{ccccc:c} \overset{x_1}{1} & \overset{x_2}{3} & \overset{x_3}{0} & \overset{x_4}{-1} & \overset{x_5}{0} & \overset{\mathbf{b}}{4} \\ 0 & -1 & 0 & -2 & 1 & 5 \\ 0 & 4 & 1 & 3 & 0 & 12 \end{array}\right)$$

A base é formada por x_1, x_3 e x_5, mas a *ordem* das colunas da base é x_1, x_5, x_3, porque as colunas da base são

$$\begin{pmatrix} \overset{x_1}{1} & \overset{x_3}{0} & \overset{x_5}{0} \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Se as reorganizarmos para formar a identidade, chegamos a

$$A_B = \begin{pmatrix} \overset{x_1}{1} & \overset{x_5}{0} & \overset{x_3}{0} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Assim, o vetor de custos deve ter as variáveis na mesma ordem:

$$\mathbf{c}_B^T = (\overset{x_1}{2}, \overset{x_5}{0}, \overset{x_3}{1}).$$

Isto porque usaremos não somente índices de colunas, mas também de linhas para facilitar o desenvolvimento mais adiante. A i-ésima variável básica tem sua coluna iqual a $\mathbf{e_i}$ (o i-ésimo vetor da base canônica).

Os coeficientes reduzidos de custo da base são zero. Para obter os coeficientes reduzidos de custo de x_2 e x_4, calculamos $\mathbf{z}^T = \mathbf{c}_B^T A_N$, e $\mathbf{r} = \mathbf{c} - \mathbf{z}$.

$$\mathbf{c}_B^T A_N = \begin{pmatrix} 2 & 0 & 1 \end{pmatrix} \begin{pmatrix} 3 & -1 \\ -1 & -2 \\ 4 & 3 \end{pmatrix} = \begin{pmatrix} 10 & 1 \end{pmatrix}.$$

Temos portanto

$$\begin{aligned} z_2 &= 10 \\ z_4 &= 1, \end{aligned}$$

e os coeficientes são

$$\begin{aligned} r_2 &= c_2 - z_2 = 3 - 10 = -7 \\ r_4 &= c_4 - z_4 = 0 - 1 = -1. \end{aligned}$$

O tableau completo tem os coeficientes reduzidos de custo das variáveis não básicas na última linha, e o valor do objetivo na última posição (com sinal trocado).

$$\left(\begin{array}{ccccc:c} \overset{x_1}{1} & \overset{x_2}{3} & \overset{x_3}{0} & \overset{x_4}{-1} & \overset{x_5}{0} & \overset{\mathbf{b}}{4} \\ 0 & -1 & 0 & -2 & 1 & 5 \\ 0 & 4 & 1 & 3 & 0 & 12 \\ \hdashline 0 & -7 & 0 & -1 & 0 & -20 \end{array}\right)$$

Os coeficientes reduzidos de custo são

$$\frac{\partial z}{\partial x_2} = c_2 - z_2 = -7, \quad \frac{\partial z}{\partial x_4} = c_4 - z_4 = -1.$$

Como os coeficientes reduzidos de custo são todos negativos (-7 e -1), a solução já é ótima: não há variáveis que possam ser incluidas na base aumentando o valor da solução. ◀

4.5 A operação de mudança de base

Outra maneira de visualizar o tableau Simplex:

$$\begin{pmatrix} A_B & A_N & \mathbf{b} \\ \mathbf{c}_B^T & \mathbf{c}_N^T & -z_0 \end{pmatrix}$$

Queremos percorrer diferentes bases do programa linear, sempre aumentando o valor da função objetivo. Manteremos as colunas da base sempre formando uma matriz identidade $m \times m$. Como já vimos antes, isto representa a descrição de cada uma das variáveis básicas em função das não básicas.

Suponha agora que queiramos incluir uma variável x_q na base, retirando x_p. Podemos escrever x_q em função das variáveis não básicas (inclusive x_p). Para isso basta operarmos as linhas do tableau, forçando a coluna $\mathbf{a_q}$ a ficar igual a $\mathbf{a_p}$. Para isso pode-se usar a operação definida a seguir.

A fim de simplificar as referências ao tableau, usaremos a_{in+1} para b_i e a_{m+1j} para r_j.

Proposição 4.4. *Seja um tableau Simplex representando o programa linear definido por A, $\mathbf{b}$ e $\mathbf{c}$. Seja $\mathbf{a_p}$ uma coluna da base e $\mathbf{a_q}$ uma coluna fora da base. Então a seguinte operação modifica a coluna $\mathbf{a_q}$ de forma que fique idêntica a $\mathbf{a_p}$, podendo substituí-la na base – e modifica também todas as colunas de A_N, inclusive $\mathbf{a_p}$, de maneira que as soluções de $A\mathbf{x} = \mathbf{b}$ sejam*

as mesmas. Para todo elemento a_{ij}*, inclusive na coluna* b *e na linha do objetivo, faça:*

$$a'_{pj} = \frac{a_{pj}}{a_{pq}}$$
$$a'_{ij} = a_{ij} - \frac{a_{pj}a_{iq}}{a_{pq}} \quad (i \neq p).$$

Demonstração. A coluna $\mathbf{a_p}$ tinha zeros exceto em a_{pp}, onde tinha um. Para $\mathbf{a_q}$, temos $a'_{pq} = a_{pq}/a_{pq} = 1$, e pode-se verificar que para todo $i \neq p$, $a'_{iq} = 0$. A operação não muda as soluções: a primeira é multiplicação de linha por escalar, e a segunda é soma de múltiplo de linha a outra. ■

4.6 Que variável entra na base?

Sabemos como retirar uma coluna $\mathbf{a_p}$ da base, trocando-a por outra coluna $\mathbf{a_q}$. Agora tratamos de como escolher a coluna (ou seja, a variável) que deve ser incluída na base.

Já sabemos que os coeficientes reduzidos de custo, dados por $r_j = c_j - z_j$, nos informam quanto uma unidade de cada variável não básica x_j aumenta a função objetivo, e podemos portanto escolher alguma variável com $r_j > 0$ (presumindo que estamos *maximizando* – se estivermos minimizando escolhemos $r_j < 0$), tendo assim a garantia de que melhoraremos a solução.

Este raciocínio é formalizado no teorema 4.5.

Teorema 4.5. *Seja* $\mathbf{x}$ *uma solução viável não degenerada para um problema de programação onde queiramos maximizar o objetivo, sendo* z *o valor de* $\mathbf{x}$*. Se o programa linear é limitado e existe* j *tal que* $(c_j - z_j) > 0$ *então há uma solução viável* $\mathbf{x}'$ *com valor* $z' > z$*, e* $\mathbf{x}'$ *pode ser obtida incluindo* a_j *na base.*

Demonstração. Temos $\mathbf{x} = (x_1, x_2, \ldots, x_m, 0, \ldots, 0)$. Suponha, sem perda de generalidade, que $j = m + 1$. Seja então

$$\mathbf{x}' = (x'_1, x'_2, \ldots, x'_m, x'_{m+1}, 0, \ldots, 0),$$

com $x'_{m+1} > 0$ uma solução viável. Temos

$$z' = z + \sum_{j>m} (c_j - z_j)x_j$$

Como nesta soma somente $x_{m+1} > 0$,

$$z' - z = (c_{m+1} - z_{m+1})x_{m+1}$$

Mas tanto $(c_{m+1} - z_{m+1})$ como x_{m+1} são positivos, portanto $z - z_0 > 0$ e

$$z' > z. \qquad \blacksquare$$

Os dois corolários a seguir são consequências simples do teorema 4.5.

Corolário 4.6. *Se $c_j - z_j \leq 0$ para todo j, a solução representada no tableau é ótima.*

Corolário 4.7. *Se $c_j - z_j = 0$ para alguma variável não básica x_j, então há mais de uma solução viável básica ótima (e consequentemente infinitas soluções ótimas).*

Proposição 4.8. *Se $a_{ij} \leq 0$ para todos i e j, o problema é ilimitado.*

Pode haver mais de um j tal que $(c_j - z_j) > 0$, e nem sempre o maior deles é a melhor opção. Precisamos de um critério de escolha. Dentre os critérios que podem ser usados merecem ser mencionados os seguintes:

- *Maior coeficiente reduzido de custo*, a regra proposta originalmente por George Dantzig.
- *Maior aumento absoluto no objetivo*. Verifica-se quanto uma unidade de cada variável x_j valeria, e multiplica-se por c_j. O maior valor é escolhido.
- *Aresta mais íngreme*, com o maior aumento no valor do objetivo para cada unidade de distância. Se a solução viável básica atual é $\mathbf{x}$ e estamos considerando outra, $\mathbf{x}'$, então a diferença entre os valores de objetivo é $\mathbf{c}^T\mathbf{x}' - \mathbf{c}^T\mathbf{x}$, ou $\mathbf{c}^T(\mathbf{x}' - \mathbf{x})$. A distância percorrida de $\mathbf{x}$ até $\mathbf{x}'$ é a norma de $\mathbf{x}' - \mathbf{x}$. Assim, escolhemos

$$\max_{\mathbf{x}'} \frac{\mathbf{c}^T(\mathbf{x}' - \mathbf{x})}{\|\mathbf{x}' - \mathbf{x}\|}.$$

- *Regra de Bland*: escolha a variável com menor índice. A única vantagem desta regra é a de garantir que não haverá ciclos.
- *Escolha aleatória*, que pode ser uma boa escolha por resultar em baixa probabilidade de ciclos, usualmente convergindo mais rápido que quando a regra de Bland é usada.

Estritamente falando, nenhuma das regras é melhor que as outras: pode-se construir exemplos onde cada uma delas leva a convergência mais lenta que alguma outra regra.

Exemplo 4.9. Suponha que em um problema com vetor de custos

$$\mathbf{c}^T = (1 \;\; 2 \;\; 3 \;\; 1)$$

tenhamos o seguinte tableau:

$$\left(\begin{array}{ccccccc|c} \overset{x_1}{-1} & \overset{x_2}{0} & \overset{x_3}{6} & \overset{x_4}{0} & \overset{x_5}{1} & \overset{x_6}{2} & \overset{x_7}{-1} & \mathbf{b} \\ 1/2 & 1 & -1 & 0 & 0 & -1/2 & 1/2 & 5 \\ 1/2 & 0 & -1 & 1 & 0 & -1/2 & -1/2 & 10 \\ \hline & & & & & & & -20 \end{array}\right)$$

A base é formada por x_5, x_2 e x_4:

$$A_B = \begin{pmatrix} \overset{x_5}{1} & \overset{x_2}{0} & \overset{x_4}{0} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

O vetor de custos das básicas é

$$\mathbf{c}_B^T = \begin{pmatrix} \overset{x_5}{0} & \overset{x_2}{2} & \overset{x_4}{1} \end{pmatrix}$$

As colunas não básicas de A são

$$A_N = \begin{pmatrix} \overset{x_1}{-1} & \overset{x_3}{6} & \overset{x_6}{2} & \overset{x_7}{-1} \\ 1/2 & -1 & -1/2 & 1/2 \\ 1/2 & -1 & -1/2 & -1/2 \end{pmatrix},$$

e portanto

$$\begin{aligned} \mathbf{z} &= \mathbf{c_B}^T A_N \\ &= (0, 2, 1) \begin{pmatrix} -1 & 6 & 2 & -1 \\ 1/2 & -1 & -1/2 & 1/2 \\ 1/2 & -1 & -1/2 & -1/2 \end{pmatrix} \\ &= \begin{pmatrix} \overset{x_1}{3/2} & \overset{x_3}{-3} & \overset{x_6}{-3/2} & \overset{x_7}{1/2} \end{pmatrix} \end{aligned}$$

Finalmente, temos

$$\begin{aligned} c_{N(1)} - z_1 &= 1 - (3/2) = -1/2 \\ c_{N(2)} - z_2 &= 3 - (-3) = 6 \\ c_{N(3)} - z_3 &= 0 - (-3/2) = 3/2 \\ c_{N(4)} - z_4 &= 0 - (1/2) = -1/2 \end{aligned}$$

Inserimos os coeficientes reduzidos de custo no tableau:

$$\left(\begin{array}{ccccccc|c} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & \mathbf{b} \\ -1 & 0 & 6 & 0 & 1 & 2 & -1 & 10 \\ 1/2 & 1 & -1 & 0 & 0 & -1/2 & 1/2 & 5 \\ 1/2 & 0 & -1 & 1 & 0 & -1/2 & -1/2 & 10 \\ \hline -1/2 & 0 & 6 & 0 & 0 & 3/2 & -1/2 & -20 \end{array}\right)$$

Dentre as não básicas, somente x_3 e x_6 tem coefciente reduzido de custo positivo, portanto é uma destas que deve entrar na base. ◀

4.7 Que variável sai da base?

Sabendo a coluna que entrará na base, nos resta escolher uma coluna da base para remover. Como estaremos retirando um valor que contribui para o objetivo, devemos escolher uma variável que, ao ser removida, removerá do objetivo o menor valor possível (para problemas de maximização; para minimização todo o raciocínio é simétrico).

4.7.1 Intuição

Esta seção passa a intuição que fundamenta o raciocínio da remoção de variáveis da base em um tableau Simplex. Não é demonstração formal, porque é feita com pouco rigor e apenas para $\mathbb{R}^2$, mas deve ajudar a compreender a explanação rigorosa na seção seguinte.

Podemos interpretar *todas* as variáveis em um problema como se fossem de folga. Como primeiro exemplo, temos um problema extremamente simples:

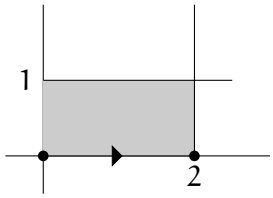
$$\begin{aligned} \max\ & x_1 + x_2 \\ \text{s.a.:}\ & x_1 \leq 2 \\ & x_2 \leq 1 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Podemos reescrever este problema explicitando cada uma das restrições de não-negatividade, apenas para maior clareza:

$$\begin{aligned} \max\ & x_1 + x_2 \\ \text{s.a.:}\ & x_1 \leq 2 \\ & x_2 \leq 1 \\ & x_1 \geq 0 \\ & x_2 \geq 0 \end{aligned}$$

Suponha que tenhamos começado com a solução viável básica $x_1 = x_2 = 0$. *a folga da restrição "$x_1 \geq 0$" é zero!*

Quando aumentamos o valor de x_1 para 2, estamos aumentando a folga desta restrição:

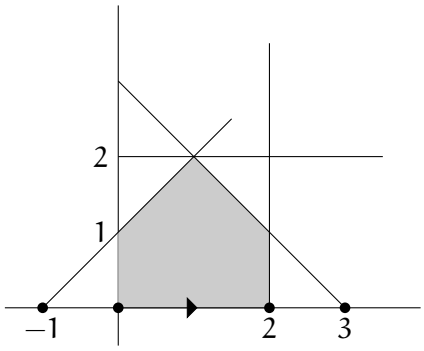


Temos portanto claro, agora, que toda vez que incluímos uma variável na base, podemos entender que estamos aumentando a folga de uma restrição.

Nos resta determinar quanto podemos avançar em uma dada direção. Considere o seguinte problema:

$$\begin{aligned} \max\ & x_1 + x_2 \\ \text{s.a.:}\ & x_1 + x_2 \leq 3 \\ & x_1 \leq 2 \\ & x_2 \leq 2 \\ & -x_1 + x_2 \leq 1 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Mostramos a seguir a região viável. Suponha que estejamos considerando o ponto extremo $(0,0)$, e que queiramos incluir x_1 na base. A figura mostra os pontos extremos onde podemos chegar mudando o valor de x_1.



O ponto $(0,0)$ toca nas restrições $x_1 \geq 0$ e $x_2 \geq 0$, portanto estas tem folga zero, e *as outras folgas estão na base*, com valor positivo.

Se retirarmos a folga da restrição $x_1 + x_2 \leq 3$, estaremos *aumentando o valor de* x_1 *até tocarmos nesta restrição* – mas então chegaremos ao ponto $(0, 3)$, que está fora da região viável.

Se removermos a folga da restrição $-x_1 + x_2 \leq 1$, estaríamos *diminuindo* o valor de x_1 (porque é a única maneira de chegar na reta $-x_1 + x_2 \leq 1$ saindo de $(0, 0)$. Além de cairmos fora da região viável, estaríamos indo no sentido oposto ao que queremos (iríamos na direção contrária à da derivada direcional $\partial z/\partial x_1$).

A opção correta é, portanto, retirar a folga da variável que representa o ponto mais próximo na direção em que estamos nos movendo: ou seja, *a variável cuja folga é a menor dentra as folgas positivas*. Em nosso exemplo, queremos aumentar x_1 *somente até chegar na restrição* $x_1 \leq 2$.

Sabendo que incluiremos a variável x_i na base, precisamos então calcular os valores das folgas para cada variável que poderíamos retirar. Isso é feito observando o tableau.

Suponha que queiramos incluir a variável x_q na base. Ela deixará de ser zero, e *a folga de alguma restrição passará a ser aumentada nesse valor*.

Assim, para cada possível variável x_p a sair da base, verificamos qual seria o valor de x_q se x_p saísse (basta determinar o pivô no tableau), e escolhemos o menor dos positivos. Procuramos portanto dentre todos os valores *positivos* b_i/a_{ij}, o menor.

Se a variável a entrar na base é x_q, então a variável a sair é x_p, onde

$$p = \arg\min_{i \leq m} \left\{ \frac{x_i}{a_{iq}} : a_{iq} > 0 \right\}.$$

4.7.2 Elaboração com rigor

Seja $\mathbf{x} = (x_1, x_2, \ldots, x_m, 0, \ldots, 0)$ uma solução viável básica.

$$\mathbf{a_1} x_1 + \mathbf{a_2} x_2 + \ldots + \mathbf{a_m} x_m = \mathbf{b} \tag{4.1}$$

Queremos introduzir $\mathbf{a_q}$ na base. Quem sai?

Observamos que, *como* A_B *é base para o espaço-coluna de* A, $\mathbf{a_q}$ pode ser descrita como combinação linear de colunas de A_B. Mais ainda, *a base que usamos é canônica* (é a matriz identidade). Assim, temos

$$a_{1q}\mathbf{a_1} + a_{2q}\mathbf{a_2} + \ldots + a_{mq}\mathbf{a_m} = \mathbf{a_q} \tag{4.2}$$

Exemplo 4.10. Suponha que a matriz de coeficientes seja

$$A = \begin{pmatrix} \overset{x_1}{1} & \overset{x_2}{0} & \overset{x_3}{2} & \overset{x_4}{1} \\ 0 & 1 & 5 & 7 \end{pmatrix},$$

e que a base seja formada pelas duas primeiras colunas.

$$A_B = \begin{pmatrix} \overset{x_1}{1} & \overset{x_2}{0} \\ 0 & 1 \end{pmatrix}$$

Queremos escrever a coluna $\mathbf{a_3}$ da matriz.

$$\begin{pmatrix} 2 \\ 5 \end{pmatrix} = 2\begin{pmatrix} 1 \\ 0 \end{pmatrix} + 5\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$ ◀

Agora escolha um $\varepsilon \geq 0$. Calculamos (4.1)$-\varepsilon$(4.2):

$$\begin{aligned} \mathbf{a_1}(x_1 - a_{1q}\varepsilon) + \ldots + \mathbf{a_m}(x_m - a_{mq}\varepsilon) &= \mathbf{b} - \varepsilon\mathbf{a_q} \\ \varepsilon\mathbf{a_q} + \mathbf{a_1}(x_1 - a_{1q}\varepsilon) + \ldots + \mathbf{a_m}(x_m - a_{mq}\varepsilon) &= \mathbf{b} \end{aligned}$$

e portanto

$$\varepsilon\mathbf{a_q} + \sum_{i \leq m} \mathbf{a_i}(x_i - \varepsilon a_{iq}) = \mathbf{b}$$

Temos agora $m + 1$ variáveis diferentes de zero, porque introduzimos a coluna $\mathbf{a_q}$ (e consequentemente a variável x_q). Deixamos de ter uma *base*, porque temos colunas LD. As variáveis do novo sistema são $x_i - \varepsilon a_{iq}$. Podemos remover uma das colunas tornando uma dessas variáveis zero, e faremos isto variando ε a partir do zero.

Para ε suficientemente pequeno mas não zero, $x_i - \varepsilon a_{iq}$ é uma solução viável, mas não básica (com $\varepsilon = 0$ temos a solução básica da qual partimos, $\mathbf{x}$). Queremos aumentar ε tanto quanto pudermos, mas sem tornar nenhum $x_i - \varepsilon a_{iq}$ negativo, porque assim tornaríamos a solução inviável.

Além disso, observamos que se $a_{iq} < 0$, não adianta tentar aumentar ε – aquela coluna não sairá da base.

Queremos então que, $\forall i \leq m$,

$$\begin{aligned} x_i - \varepsilon a_{iq} &\geq 0 \\ x_i &\geq \varepsilon a_{iq} \\ \frac{x_i}{a_{iq}} &\geq \varepsilon \end{aligned}$$

Assim, escolhemos para deixar a base a coluna $\mathbf{a_p}$ tal que p seja o índice que minimiza x_i/a_{iq}:

$$p = \arg\min_{i \leq m} \left\{ \frac{x_i}{a_{iq}} : a_{iq} > 0 \right\}$$

Se todos os a_{iq} forem negativos, o programa linear é ilimitado.

Exemplo 4.11. Suponha que $\mathbf{c}^T = (1, 1, 2)$, e considere o seguinte tableau.

$$\left(\begin{array}{ccccc|c} \overset{x_1}{1} & \overset{x_2}{3} & \overset{x_3}{0} & \overset{x_4}{-1} & \overset{x_5}{0} & \overset{b}{4} \\ 0 & -1 & 0 & 2 & 1 & 5 \\ 0 & -2 & 1 & 3 & 0 & 12 \\ \hline 0 & 3 & 0 & -7 & 0 & -20 \end{array}\right)$$

A coluna $\mathbf{a_2}$ entrará na base; verificaremos qual coluna deverá sair. Dividimos cada x_i pelos elementos de $\mathbf{a_4}$:

$$\begin{aligned} \frac{b_1}{a_{12}} &= 4 \div 3 = 4/3 \\ \frac{b_2}{a_{22}} &= 5 \div -1 = -5 \\ \frac{b_3}{a_{32}} &= 12 \div -2 = -6 \end{aligned}$$

Descartamos os valores negativos e determinamos que a *primeira* variável básica (x_1) deve sair da base. ◀

4.8 Método Simplex (algoritmo)

O algoritmo Simplex é mostrado a seguir em pseudocódigo. Nesta descrição o critério de seleção de variável a entrar na base não é explicitado. Ao escolher a coluna a sair da base, usamos b_i, já que por enquanto representamos a base como identidade, e nesta situação $x_i = b_i$.

construa o tableau de uma solução viável básica
enquanto existe $r_j > 0$:
    selecione q tal que $r_q > 0$ // x_q entrará na base
    calcule b_i/a_{iq} p/todo $a_{iq} \neq 0$ e $i \leq m$
    se todo b_i/a_{iq} é ≤ 0 : PARE -- problema ilimitado
    $p \leftarrow \min_{i \leq m} \left\{ \frac{b_i}{a_{iq}}, a_{iq} > 0 \right\}$
    faça x_q entrar na base, retirando x_p:
        $a'_{pj} \leftarrow \frac{a_{pj}}{a_{pq}}$
        $a'_{ij} \leftarrow a_{ij} - a_{pj}\frac{a_{iq}}{a_{pq}} \quad (i \neq p)$
retorne $\mathbf{x_B}$ // solução ótima

Exemplo 4.12. Considere o problema a seguir.

$$\begin{aligned} \max\ & 2x_1 + x_2 \\ \text{s.a.:}\ & x_1 - \frac{x_2}{2} \leq 3 \\ & 2x_1 + 3x_2 \leq 20 \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Adicionamos variáveis de folga para transformar as desigualdades em igualdades, obtendo

$$\begin{aligned} \max\ & 2x_1 + x_2 \\ \text{s.a.:}\ & x_1 - \frac{x_2}{2} + x_3 = 3 \\ & 2x_1 + 3x_2 + x_4 = 20 \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Montamos então o tableau Simplex.

$$\left(\begin{array}{cccc:c} \overset{x_1}{1} & \overset{x_2}{-1/2} & \overset{x_3}{1} & \overset{x_4}{0} & 3 \\ 2 & 3 & 0 & 1 & 20 \\ \hdashline 2 & 1 & 0 & 0 & 0 \end{array}\right)$$

Todas as variáveis não básicas tem coeficientes reduzidos positivos, portanto podemos escolher qualquer uma delas. Escolhemos x_1. Para determinar a coluna a sair, dividimos cada x_i básica pelos elementos correspondentes da coluna que entrará ($\mathbf{a_1}$):

$$\begin{aligned} 3/1 = 3 \quad &\Leftarrow \\ 20/2 = 10 \end{aligned}$$

O menor deles é o primeiro, portanto retiraremos então a coluna atualmente ocupada pela *primeira* variável básica (x_3).

$$\left(\begin{array}{cccc|c} \overset{\downarrow}{1} & -1/2 & \overset{\uparrow}{1} & 0 & 3 \\ 2 & 3 & 0 & 1 & 20 \\ \hline 2 & 1 & 0 & 0 & 0 \end{array}\right)$$

As setas indicam as colunas a entrar e sair da base. Usamos operações elementares nas linhas do tableau para tornar a primeira coluna igual à coluna $\mathbf{a_3}$: como a_{11} (que está marcado no tableau) já é um, só precisamos tornar a_{21} igual a zero. Então subtraímos da segunda linha duas vezes a primeira. O tableau resultante é

$$\left(\begin{array}{cccc|c} 1 & -1/2 & 1 & 0 & 3 \\ 0 & 4 & -2 & 1 & 14 \\ \hline 0 & 2 & -2 & 0 & -6 \end{array}\right)$$

Note que agora temos apenas uma variável com coeficiente reduzido positivo. Ela entrará na base. Para determinar a coluna a sair, calculamos

$$\begin{aligned} 3/(-1/2) = -6 \quad & (\leq 0) \\ 14/4 = 7/2 \quad & \Leftarrow \end{aligned}$$

A *segunda* variável não básica (x_3), portanto, sai da base.

$$\left(\begin{array}{cccc|c} 1 & \overset{\downarrow}{-1/2} & 1 & \overset{\uparrow}{0} & 3 \\ 0 & \boxed{4} & -2 & 1 & 14 \\ \hline 0 & 2 & -2 & 0 & -6 \end{array}\right)$$

O elemento marcado ($a_{22} = 4$) será transformado em um usando operações elementares. O tableau resultante é:

$$\left(\begin{array}{cccc|c} 1 & 0 & 3/4 & 1/8 & 19/4 \\ 0 & 1 & -1/2 & 1/4 & 7/2 \\ \hline 0 & 0 & -1 & -1/2 & -13 \end{array}\right)$$

Como agora os coeficientes reduzidos de custo são todos negativos, chegamos a uma solução ótima, com $x_1 = 19/4$, $x_2 = 7/2$, e valor do objetivo igual a 13. ◀

Exemplo 4.13. Mostramos agora que quando não seguimos o método de escolha da variável a sair da base, podemos chegar a uma solução inviável.

No exemplo 4.12, a primeira variável que escolhemos para entrar na base foi x_1. O tableau era

$$\left(\begin{array}{cccc|c} \overset{\downarrow}{\boxed{1}} & -1/2 & 1 & 0 & 3 \\ 2 & 3 & 0 & 1 & 20 \\ \hline 2 & 1 & 0 & 0 & 0 \end{array}\right) \begin{matrix} \\ \\ \\ \end{matrix} \begin{matrix} x_2 & x_3 & x_4 & \\ & & & \\ & & & \\ & & & \end{matrix}$$

Para escolher a variável a sair calculamos $3/1 = 1$ e $20/2 = 10$, devendo portanto escolher a primeira coluna da base. Se fizermos o contrário e escolhermos a segunda coluna, teremos que multiplicar a segunda linha por $1/2$, e depois subtraí-la da primeira. O resultado será

$$\begin{pmatrix} 0 & -2 & 1 & -1/2 & -7 \\ 1 & 3/2 & 0 & 1/2 & 10 \end{pmatrix} \begin{matrix} x_1 & x_2 & x_3 & x_4 & \\ & & & & \\ & & & & \end{matrix}$$

No entanto, esta solução é inviável, porque determina $x_3 = -7$. A variável x_3 é de folga, e se atribuirmos a ela valor negativo, estaríamos violando uma desigualdade. ◀

Teorema 4.14. *Se após a execução do método Simplex uma solução ótima tiver sido encontrara e houver coeficientes reduzidos de custo com valor zero, então há múltiplas (infinitas) soluções ótimas para o problema.*

Exemplo 4.15. O seguinte problema tem uma restrição ortogonal ao gradiente do objetivo:

$$\begin{aligned} \max\ & x_1 + 2x_2 \\ \text{s.a.:}\ & x_1 + 2x_2 \leq 20 \\ & x_1 \leq 10 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

O tableau final para este problema é

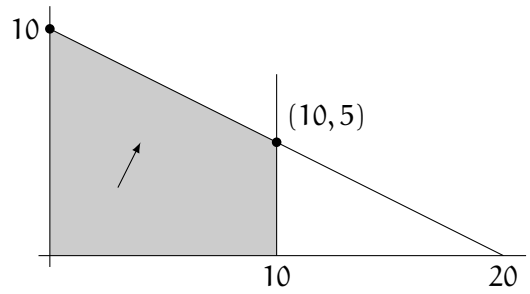
$$\left(\begin{array}{cccc|c} 0 & 1 & 1/2 & -1/2 & 5 \\ 1 & 0 & 0 & 1 & 10 \\ \hline 0 & 0 & -1 & 0 & -20 \end{array}\right) \begin{matrix} \\ \\ \\ \end{matrix} \begin{matrix} x_1 & x_2 & x_3 & x_4 & \\ & & & & \\ & & & & \\ & & & & \end{matrix}$$

e a solução ótima é $x_1 = 10, x_2 = 5$.

Temos zero no coeficiente reduzido de custo de x_4 (a folga da segunda restrição). Se a incluírmos na base, obtemos outro tableau,

$$\left(\begin{array}{cccc|c} 1/2 & 1 & 1/2 & 0 & 10 \\ 1 & 0 & 0 & 1 & 10 \\ \hline 0 & 0 & -1 & 0 & -20 \end{array}\right) \begin{matrix} \\ \\ \\ \end{matrix} \begin{matrix} x_1 & x_2 & x_3 & x_4 & \\ & & & & \\ & & & & \\ & & & & \end{matrix}$$

com outra solução ótima: $x_1 = 0, x_2 = 10$. Todas as combinações convexas das duas soluções ótimas são também ótimas. A figura a seguir mostra a região viável, as duas soluções e o gradiente do objetivo.



Como o gradiente é ortogonal à restrição $x_1 + 2x_2 \leq 20$, todos os pontos viáveis da reta $x_1 + 2x_2 = 20$ são ótimos. ◀

A partir da próxima seção, deixaremos de incluir os nomes das variáveis no tableau.

4.9 Obtendo uma solução viável básica inicial

Para problemas do tipo $\max \mathbf{c}^T\mathbf{x}$ s.a. $A\mathbf{x} \leq \mathbf{b}$, as variáveis de folga nos permitiram formar facilmente uma solução básica inicial. Nem todo problema terá uma variável de folga para cada restrição que nos permita obter de maneira simples uma solução inicial.

4.9.1 O método das duas fases

Considere agora o problema

$$\begin{aligned} \max: \ & \mathbf{c}^T\mathbf{x} \\ \text{s.a.:} \ & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned} \tag{4.3}$$

Uma maneira de obter uma solução viável básica é criar variáveis artificiais (não de folga – variáveis sem qualquer significado, cuja única razão de existir é permitir-nos encontrar uma solução inicial para o tableau Simplex). Veja por exemplo o problema a seguir.

$$\begin{aligned} \min: \ & \sum y_i \\ \text{s.a.:} \ & A\mathbf{x} + \mathbf{y} = \mathbf{b} \\ & \mathbf{x}, \mathbf{y} \geq \mathbf{0} \end{aligned} \tag{4.4}$$

Proposição 4.16. *O problema 4.3 é viável se e somente se 4.4 tem ótimo com* $\mathbf{y} = \mathbf{0}$.

Demonstração. Trivialmente, se 4.3 é viável 4.4 também tem ótimo com $\mathbf{y} = \mathbf{0}$. Se $\mathbf{y} = \mathbf{0}$, claramente podemos desprezar $\mathbf{y}$ obtendo solução viável para 4.3.

Se 4.4 tem ótimo com $\mathbf{y} > \mathbf{0}$, não há como diminuir $\mathbf{y}$ mantendo a solução viável: note que as variáveis y_i funcionam como variáveis de folga para o problema original, 4.3:

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \big[+ a_{1n+1}y_1 \big] = b_1,$$

e se houver algum $y_i > 0$, temos uma variável de folga sendo usada. Isso significa que o lado esquerdo de uma das restrições deve ser *estritamente* menor que o direito:

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n < b_1,$$

mas o problema original, 4.3, não tem estas variáveis de folga, e portanto é inviável. ■

Resolver 4.4 é fácil, com solução básica inicial $\mathbf{y} = \mathbf{b}$. A solução final, se houver, *não* terá variáveis y_i na base (porque o mínimo do objetivo se dá com $\mathbf{y} = \mathbf{0}$), e portanto serve para 4.3.

Exemplo 4.17. Temos agora o seguinte problema:

$$\begin{aligned} \max : & \, x_1 + 2x_2 \\ \text{s.a.:} & \, x_1 + x_2 \geq 1 \\ & \, x_1 + \frac{1}{2}x_2 \leq 1 \\ & \, \mathbf{x} \geq 0 \end{aligned}$$

Como a primeira restrição torna a origem inviável, podemos adicionar um vetor artificial $\mathbf{y}$, que minimizaremos:

$$\begin{aligned} \max : & \, -y_1 - y_2 \\ \text{s.a.:} & \, x_1 + x_2 + y_1 \geq 1 \\ & \, x_1 + \frac{1}{2}x_2 + y_2 \leq 1 \\ & \, \mathbf{x}, \mathbf{y} \geq 0 \end{aligned}$$

Passamos o problema para a forma padrão:

$$\begin{aligned} \max: & -y_1 - y_2 \\ \text{s.a.:} & \; x_1 + x_2 - x_3 + y_1 = 1 \\ & \; x_1 + \frac{1}{2}x_2 + x_4 + y_2 = 1 \\ & \; \mathbf{x}, \mathbf{y} \geq 0 \end{aligned}$$

e contruimos o tableau:

$$\left(\begin{array}{cccccc|c} 1 & 1 & -1 & 0 & 1 & 0 & 1 \\ 1 & 1/2 & 0 & 1 & 0 & 1 & 1 \\ \hline & & & & & & \end{array}\right)$$

No entanto, percebemos que só precisávamos da variável y_1, porque a coluna $(0,1)^T$, incluído para a variável y_2, não nos ajuda: já havia a coluna $(0,1)^T$ de x_4. Eliminamos y_2, portanto:

$$\begin{aligned} \max: & -y \\ \text{s.a.:} & \; x_1 + x_2 - x_3 + y_1 = 1 \\ & \; x_1 + \frac{1}{2}x_2 + x_4 = 1 \\ & \; \mathbf{x}, \mathbf{y} \geq 0 \end{aligned}$$

Contruimos novamente o tableau:

$$\left(\begin{array}{cccccc|c} 1 & 1 & -1 & 0 & 1 & 1 \\ 1 & 1/2 & 0 & 1 & 0 & 1 \\ \hline & & & & & \end{array}\right)$$

Calculamos os coeficientes reduzidos de custo:

$$\left(\begin{array}{cccccc|c} 1 & 1 & -1 & 0 & 1 & 1 \\ 1 & 1/2 & 0 & 1 & 0 & 1 \\ \hline 1 & 1 & -1 & 0 & 0 & -1 \end{array}\right)$$

Incluiremos x_2. Como

$$\begin{aligned} 1 \div 1 &= 1 \\ 1 \div 1/2 &= 2, \end{aligned}$$

removeremos y da base (o que é bom, porque estamos minimizando y, e seu valor passará a ser zero).

$$\left(\begin{array}{ccccc|c} 1 & 1 & -1 & 0 & 1 & 1 \\ 1/2 & 0 & 1/2 & 1 & -1/2 & 1/2 \\ \hline 0 & 0 & 0 & 0 & -1 & \end{array}\right)$$

E o tableau agora é ótimo. Temos a solução viável básica

$$\begin{aligned} x_1 &= 0 \\ x_2 &= 1 \\ x_3 &= 0 \\ x_4 &= 1/2 \\ y &= 0 \end{aligned}$$

Podemos aproveitar o tableau para resolver o problema original, e usamos a base (x_2, x_4).

$$\left(\begin{array}{cccc|c} 1 & 1 & -1 & 0 & 1 \\ 1/2 & 0 & 1/2 & 1 & 1/2 \\ \hline & & & & 0 \end{array}\right)$$

Agora temos qeu recalcular os coeficientes reduzidos de custo, porque nosso vetor $\mathbf{c}$ mudou:

$$\mathbf{c}^T = (1\ 2).$$

O tableau, com os coeficientes recalculados, é

$$\left(\begin{array}{cccc|c} 1 & 1 & -1 & 0 & 1 \\ 1/2 & 0 & 1/2 & 1 & 1/2 \\ \hline -1 & 0 & 2 & 0 & -2 \end{array}\right)$$

Neste momento estamos no ponto extremo $(x_1 = 0, x_2 = 1)$. Incluímos x_3 na base, removendo x_4.

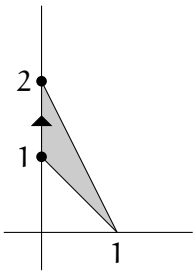
$$\left(\begin{array}{cccc|c} 2 & 1 & 0 & 2 & 2 \\ 1 & 0 & 1 & 2 & 1 \\ \hline -3 & 0 & 0 & -4 & -4 \end{array}\right)$$

Os coeficientes de custo são todos negativos, portanto temos a solução

ótima com valor 4 e

$$\begin{aligned} x_1 &= 0 \\ x_2 &= 2 \\ x_3 &= 1 \\ x_4 &= 0. \end{aligned}$$

A próxima figura mostra o caminho feito desde que começamos a fase II.



O tableau anterior nos colocava no ponto $(0,1)$, que foi o ponto encontrado na fase I. Após um passo do Simplex, mudamos para o ponto ótimo $(0,2)$. ◀

Exemplo 4.18. Temos agora o seguinte problema.

$$\begin{aligned} &\max x_1 + x_2 \\ &\text{s.a.} : x_1 + 2x_2 \leq 2 \\ &\qquad 3x_1 + 4x_2 \geq 12 \\ &\qquad \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Ao montarmos o tableau, percebemos que não há uma base óbvia que possamos usar:

$$\left(\begin{array}{cccc|c} 1 & 2 & 1 & 0 & 2 \\ 3 & 4 & 0 & -1 & 12 \\ \hline & & & & \end{array}\right)$$

Adicionamos uma variável extra y_1 (porque não precisamos de duas – podemos usar a folga da primeira linha), e minimizamos y, obtendo o tableau

$$\left(\begin{array}{ccccc|c} 1 & 2 & 1 & 0 & 0 & 2 \\ 3 & 4 & 0 & -1 & 1 & 12 \\ \hline & & & & & \end{array}\right)$$

Como estamos minimizando y, podemos simplesmente maximizar $-y$. O vetor $\mathbf{c}$ será $(0, 0, 0, -1)$.

$$\left(\begin{array}{cccccc} 1 & 2 & 1 & 0 & 0 & 2 \\ 3 & 4 & 0 & -1 & 1 & 12 \\ \hdashline 3 & 4 & 0 & -1 & 0 & \end{array}\right)$$

incluiremos x_1 na base. Verificamos quem deveria sair:

$$\begin{aligned} 2 \div 1 &= 2 \Leftarrow \\ 12 \div 3 &= 4 \end{aligned}$$

Retiramos a primeira básica, s_1.

$$\left(\begin{array}{cccccc} 1 & 2 & 1 & 0 & 0 & 2 \\ 0 & -2 & -3 & -1 & 1 & 6 \\ \hdashline 0 & -2 & -3 & -1 & 0 & \end{array}\right)$$

A solução para este problema é ótima, mas com $y_1 = 6$. Isto significa que não há solução para $A\mathbf{x} = \mathbf{b}$, e o problema original não tem solução viável. ◀

4.9.2 O método do M grande

O método do grande M realiza a mesma coisa que o método das duas fases de uma única vez.

$$\begin{aligned} \max: \ & \mathbf{c}^T\mathbf{x} \\ \text{s.a.:} \ & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned} \tag{4.5}$$

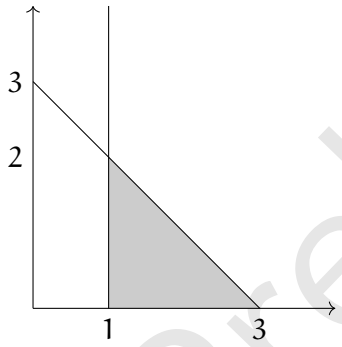
No objetivo, subtraímos do vetor $\mathbf{x}$ um outro vetor, $\mathbf{y}$, com coeficientes muito grandes. Multiplicaremos um valor grande M por um vetor de variáveis artificiais $\mathbf{y}$.

$$\begin{aligned} \max: \ & \mathbf{c}^T\mathbf{x} - M(\mathbf{1}^T\mathbf{y}) \\ \text{s.a.:} \ & A\mathbf{x} + \mathbf{y} = \mathbf{b} \\ & \mathbf{x}, \mathbf{y} \geq \mathbf{0} \end{aligned} \tag{4.6}$$

Exemplo 4.19. O problema a seguir tem uma restrição que nos impede de usar a origem como solução inicial básica.

$$\begin{aligned} \max\ & x_1 + 2x_2 \\ \text{s.a.:}\ & -x_1 \leq -1 \\ & x_1 + x_2 \leq 3 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

A representação gráfica deste problema é mostrada na próxima figura.



Observe que a origem não é viável, por isso não podemos usar $(0,0)$ como solução viável básica.

Usando o método do M grande com $M = 10$, obtemos um novo problema.

$$\begin{aligned} \max\ & x_1 + 2x_2 - 10(y_1 + y_2) \\ \text{s.a.:}\ & -x_1 + y_1 \leq -1 \\ & x_1 + x_2 + y_2 \leq 3 \\ & \mathbf{x}, \mathbf{y} \geq \mathbf{0} \end{aligned}$$

Observe que as variáveis y_i aparecem no objetivo com sinal negativo e multiplicadas pelo valor M, muito grande – assim, quando M for suficientemente grande, teremos a solução ótima com $\mathbf{y} = \mathbf{0}$, e o problema torna-se equivalente ao problema original, de onde partimos. ◀

O funcionamento do método do M grande é garantido pelas proposições a seguir.

Proposição 4.20. *Se 4.6 é ilimitado e 4.5 é viável, então 4.5 é ilimitado.*

Proposição 4.21. *Se 4.5 é viável e tem solução ótima finita, então existe um* $M > 0$ *tal que o tableau final do Simplex para 4.6 terá as variáveis artificiais fora da base.*

Proposição 4.22. *Se 4.5 é inviável então não existe* $M > 0$ *para o qual o tableau final do Simplex para 4.6 terá as variáveis artificiais fora da base.*

Exemplo 4.23.

$$\begin{aligned} \max: \ & 2x_1 + 5x_2 + 3x_3 + x_4 \\ \text{s.a.:} \ & 2x_1 - 2x_2 + x_3 + 2x_4 = 2 \\ & x_2 + x_3 - x_4 = 4 \\ & 4x_1 + 2x_2 - 8x_3 + 4x_4 = 8 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Usando o método do M grande, com $M = 500$, reescrevemos o problema da forma a seguir.

$$\begin{aligned} \min: \ & 2x_1 + 5x_2 + 3x_3 + x_4 - 500y_1 - 500y_2 - 500y_3 \\ \text{s.a.:} \ & 2x_1 - 2x_2 + x_3 + 2x_4 + y_1 = 2 \\ & x_2 + x_3 - x_4 + y_2 = 4 \\ & 4x_1 + 2x_2 - 8x_3 + 4x_4 + y_3 = 8 \\ & \mathbf{x}, \mathbf{y} \geq \mathbf{0} \end{aligned}$$

Obtemos a solução ótima $\mathbf{x} = (0, 56/9, 10/3, 50/9)^T$ e $\mathbf{y} = \mathbf{0}$. ◀

4.10 Minimização e desigualdades do tipo $\geq$

Até agora mantivemos o foco em problemas de maximização com desigualdades do tipo $\leq$. Era simples obter para esses problemas um tableau Simplex inicial, porque as variáveis de folga de das restrições formavam no tableau uma submatriz identidade. Em problemas de minimização, que usualmente tem restrições do tipo $\geq$, esta solução não seria possível, porque as variáveis de folga tem coeficiente -1, e o que surge no tableau é a identidade multiplicada por -1. Na última seção mostramos como lidar com esta situação, adicionando variáveis artificiais. Agora ilustraremos aquelas técnicas em problemas de minimização.

O simplex funciona da mesma forma em problemas de maximização e minimização. A única diferença na execução do algoritmo é que, *como queremos minimizar o objetivo, escolheremos os coeficientes reduzidos*

de custo com valor negativo ao escolher a coluna a entrar na base, e teremos uma solução ótima quando só houver coeficientes reduzidos de custo positivos.

Exemplo 4.24. Resolveremos o seguinte problema de minimização.

$$\begin{aligned} \min\ & 2x_1 + x_2 + x_3 \\ \text{s.a.:}\ & x_1 + x_2 \geq 2 \\ & x_2 + x_3 \geq 3 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Com as variáveis de folga, o problema é reescrito da seguinte maneira.

$$\begin{aligned} \min\ & 2x_1 + x_2 + x_3 \\ \text{s.a.:}\ & x_1 + x_2 - x_4 = 2 \\ & x_2 + x_3 - x_5 = 3 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Se montarmos o tableau Simplex, não teremos uma escolha simples de base inicial.

$$\left(\begin{array}{ccccc|c} 1 & 1 & 0 & -1 & 0 & 2 \\ 1 & 0 & 1 & 0 & -1 & 3 \\ \hline & & & & & \end{array}\right)$$

Não podemos usar as variáveis de folga, porque elas tem coeficientes negativos no tableau, e portanto seus valores são negativos ($x_4 = -2$, $x_5 = -3$). Também não podemos usar as colunas 2 e 3, que formam uma submatriz identidade (referentes às variáveis x_2 e x_3), porque a solução seria inviável: teríamos $x_2 = 2$, $x_3 = 3$, e $x_2 + x_3 = 5$, violando a segunda restrição.

Usamos portanto o método das duas fases. Minimizamos a função $y_1 + y_2$, e o tableau inicial é

$$\left(\begin{array}{ccccccc|c} 1 & 1 & 0 & -1 & 0 & 1 & 0 & 2 \\ 1 & 0 & 1 & 0 & -1 & 0 & 1 & 3 \\ \hline -2 & -1 & -1 & 1 & 1 & 0 & 0 & -5 \end{array}\right)$$

No cálculo dos coeficientes de custo, temos

$$\mathbf{z} = (1,1)\begin{pmatrix} 1 & 1 & 0 & -1 & 0 \\ 1 & 0 & 1 & 0 & -1 \end{pmatrix} = (2,1,1,-1,-1),$$

e portanto $\mathbf{c}_N^T - \mathbf{z}$ é $(-2,-1,-1,1,1)$. Incluiremos x_1 na base, e y_1 sairá. O próximo tableau é

$$\left(\begin{array}{ccccccc|c} 1 & 1 & 0 & -1 & 0 & 1 & 0 & 2 \\ 0 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \\ \hline 0 & 1 & -1 & -1 & -1 & 0 & 0 & -1 \end{array}\right)$$

$$\mathbf{z} = (0,1)\begin{pmatrix} 1 & 0 & -1 & 0 & 1 \\ -1 & 1 & 1 & -1 & -1 \end{pmatrix} = (-1,1,1,-1,-1),$$

e $\mathbf{c}_N^T - \mathbf{z}$ é $(1,-1,-1,-1,0)$. Escolhemos x_3 para entrar na base. A coluna a sair da base é a segunda (y_2). A parte superior do tableau não mudará. Apenas recalculamos os coeficientes reduzidos de custo.

$$\left(\begin{array}{ccccccc|c} 1 & 1 & 0 & -1 & 0 & 1 & 0 & 2 \\ 0 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

$$\mathbf{z} = (0,0), \quad A_N = (0,0,0,0,0).$$

Temos então $\mathbf{c}_N^T - \mathbf{z} = \mathbf{0}$, e a solução nos dá $\mathbf{y} = \mathbf{0}$ – o que significa que o problema original é viável. Retiramos $\mathbf{y}$ do tableau e mudamos para a função objetivo original, $\mathbf{c}^T = (2,1,1)$.

$$\left(\begin{array}{ccccc|c} 1 & 1 & 0 & -1 & 0 & 2 \\ 0 & -1 & 1 & 1 & -1 & 1 \\ \hline 0 & 0 & 0 & 1 & 1 & -3 \end{array}\right)$$

Ao calcularmos novamente os coeficientes reduzidos de custo, temos

$$\mathbf{z} = (2,1)\begin{pmatrix} 0 & -1 & 0 \\ 1 & 1 & -1 \end{pmatrix} = (1,-1,-1).$$

Assim, $\mathbf{c}_N^T - \mathbf{z} = (0,1,1)$. A solução já é ótima. ◀

4.11 Soluções degeneradas

Definição 4.25 (solução degenerada)**.** Seja um problema de programação linear com n variáveis e m restrições. Uma solução viável básica é *degenerada* se uma de suas variáveis básicas é igual a zero.

Um problema de programação linear é degenerado se uma de suas soluções viáveis básicas é degenerada. ⧫

Equivalentemente, um tableau representa uma solução degenerada se a coluna das constantes (**b**) contém algum valor igual a zero.

Exemplo 4.26. Considere o seguinte problema.

$$\begin{aligned}
&\max x_1 + x_2 + x_3 \\
&\text{s.a.} : x_1 + x_2 \leq 1 \\
&\qquad x_3 \leq 1 \\
&\qquad x_1 + x_3 \leq 2 \\
&\qquad \mathbf{x} \geq \mathbf{0}.
\end{aligned}$$

Quando usamos o método Simplex para resolvê-lo, chegamos ao seguinte tableau:

$$\left(\begin{array}{cccccc|c}
1 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & -1 & 0 & -1 & -1 & 1 & 0 \\
\hline
0 & 0 & 0 & -1 & -1 & 0 & -2
\end{array}\right)$$

com $x_1 = 1$, $x_2 = 0$ e $x_3 = 1$. Esta solução é degenerada porque x_2 é zero. ◀

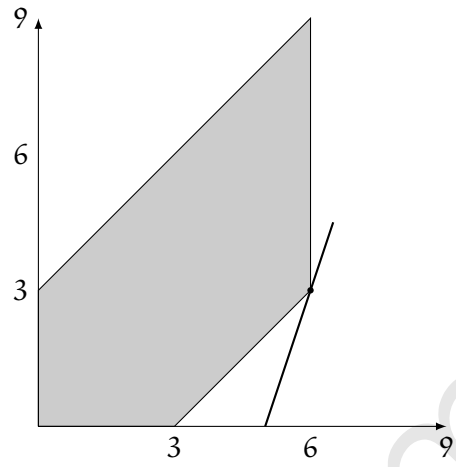
Soluções degeneradas acontecem quando uma restrição redundante toca o politopo, como ilustra o exemplo 4.27.

Exemplo 4.27. Temos a seguir um exemplo de problema degenerado.

$$\begin{aligned}
&\max x_1 + x_2 \\
&\text{s.a.:} x_1 - x_2 \leq 3 \\
&\qquad -x_1 + x_2 \leq 3 \\
&\qquad x_1 \leq 6 \\
&\qquad 3x_1 - x_2 \leq 15 \\
&\qquad \mathbf{x} \geq 0.
\end{aligned}$$

A última restrição é redundante, tocando apenas no ponto $(6, 3)$ do poliedro. Observe que esta restrição (a última) é combinação linear da primeira com a terceira:

$$\begin{aligned}
1\times\Big(x_1 - x_2 \leq 3\Big) \\
+2\times\Big(x_1 \leq 6\Big) \\
=3x_1 - x_2 \leq 15.
\end{aligned}$$



A seguir temos o mesmo problema, já na forma padrão.

$$\begin{aligned} \max\ & x_1 + x_2 \\ \text{s.a.:}\ & x_1 - x_2 + x_3 = 3 \\ & -x_1 + x_2 + x_4 = 3 \\ & x_1 + x_5 = 6 \\ & 3x_1 - x_2 + x_6 = 15 \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Montamos o tableau Simplex para o problema:

$$\left(\begin{array}{cccccc|c} 1 & -1 & 1 & 0 & 0 & 0 & 3 \\ -1 & 1 & 0 & 1 & 0 & 0 & 3 \\ 1 & 0 & 0 & 0 & 1 & 0 & 6 \\ 3 & -1 & 0 & 0 & 0 & 1 & 15 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

Podemos escolher tanto x_1 como x_2 para entrar na base. Escolhemos x_1. Para determinar a variável a sair, calculamos

$$\begin{aligned} 3 \div 1 &= 3 \Leftarrow \\ 3 \div -1 &= -3 \\ 6 \div 1 &= 6 \\ 15 \div 3 &= 5 \end{aligned}$$

e escolhemos a primeira variável não básica (x_3). O novo tableau é

$$\left(\begin{array}{cccccc|c} 1 & -1 & 1 & 0 & 0 & 0 & 3 \\ 0 & 0 & 1 & 1 & 0 & 0 & 6 \\ 0 & 1 & -1 & 0 & 1 & 0 & 3 \\ 0 & 2 & -3 & 0 & 0 & 1 & 6 \\ \hline 0 & 2 & -1 & 0 & 0 & 0 & -3 \end{array}\right)$$

Agora só podemos incluir x_2 na base, porque é a única não básica com coeficiente reduzido de custo acima de zero. Para determinar a variável a sair da base, novamente calculamos

$$\begin{aligned} 3 \div -1 &= -3 \\ 3 \div 0 &= \ldots \\ 3 \div 1 &= 3 \Leftarrow \\ 6 \div 2 &= 3 \Leftarrow \end{aligned}$$

Temos um empate entre x_5 e x_6. Estas são as variáveis de folga das duas restrições que tocam o ponto $(3, 6)$, uma delas redundante (lembre que a última restrição, de folga x_6, é combinação linear da primeira com a terceira – e a terceira tem folga x_3).

O fato de podermos retirar qualquer uma destas variáveis da base significa que podemos retirar a folga de qualquer uma das duas restrições para incluir x_2.

Escolhemos x_6 para sair da base (ou seja, retiramos a filga da última restrição, "encostando" na restrição), e o novo tableau é

$$\left(\begin{array}{cccccc|c} 1 & 0 & -1/2 & 0 & 0 & 1/2 & 6 \\ 0 & 0 & 1 & 1 & 0 & 0 & 6 \\ 0 & 0 & 1/2 & 0 & 1 & -1/2 & 0 \\ 0 & 1 & -3/2 & 0 & 0 & 1/2 & 3 \\ \hline 0 & 0 & 2 & 0 & 0 & -1 & -9 \end{array}\right)$$

O valor de x_5 é zero, e a solução é degenerada. A variável x_5 é a folga da terceira restrição. Ela é zero porque, ao "encostar" na última restrição, também tocamos na terceira.

Agora incluiremos x_3 (folga da primeira restrição), porque é nossa única

opção.

$$\begin{aligned} 6 \div -1/2 &= -12 \\ 6 \div 1 &= 6 \\ 0 \div 1/2 &= 0 \Leftarrow \\ 3 \div -3/2 &= -2 \end{aligned}$$

A variável a sair é x_5, e o novo tableau é

$$\left(\begin{array}{cccccc|c} 1 & 0 & 0 & 0 & 1 & 0 & 6 \\ 0 & 0 & 0 & 1 & -2 & 1 & 6 \\ 0 & 0 & 1 & 0 & 2 & -1 & 0 \\ 0 & 1 & 0 & 0 & 3 & -1 & 3 \\ \hline 0 & 0 & 0 & 0 & -4 & 1 & -9 \end{array}\right)$$

O valor de x_3 é zero, e temos outra solução degenerada. Observe que o valor do objetivo não mudou.

Incluiremos x_6. A variável a sair será x_4, e o tableau final é

$$\left(\begin{array}{cccccc|c} 1 & 0 & 0 & 0 & 1 & 0 & 6 \\ 0 & 0 & 0 & 1 & -2 & 1 & 6 \\ 0 & 0 & 1 & 1 & 0 & 0 & 6 \\ 0 & 1 & 0 & 1 & 1 & 0 & 9 \\ \hline 0 & 0 & 0 & -1 & -2 & 0 & -15 \end{array}\right)$$

Todos os coeficientes reduzidos de custo são negativos, portanto temos uma solução ótima, com $x_1 = 6$, $x_2 = 9$ e valor 15. ◀

Teorema 4.28. *Um problema de programação linear com uma restrição redundante que toca o politopo é degenerado.*

Teorema 4.29. *O uso da regra de Bland para escolher a coluna a entrar na base elimina a possibilidade de ciclos.*

Teorema 4.30. *Se a coluna a entrar na base for escolhida aleatoriamente, a probabilidade de ocorrência de ciclos tende a zero quando a quantidade de iterações tede a ∞.*

4.12 Método Simplex Revisado

O método Simplex mantém em memória – e percorre a cada passo – uma matriz com n colunas e m linhas. Quando n é muito maior que m, muitas

dessas colunas não são usadas durante a execução do algoritmo. Geometricamente, o caminho da solução inicial até a ótima não percorre todos os pontos extremos, mas um pequeno número deles – e isto é particularmente relevante quando a dimensão do politopo é alta. Cada vez que uma coluna entra na base, *todas* as colunas de A_N são atualizadas, mesmo que nunca venham a ser usadas. O método Simplex revisado foi desenvolvido como uma forma de evitar o armazenamento e processamento dessa informação desnecessária.

Antes de mais nada, listamos a informação de que o Simplex precisa para cada operação:

- *Coeficientes reduzidos de custo* para selecionar a coluna a entrar na base e para verificar otimalidade de soluções;
- *A coluna que entra na base e os valores de* $\mathbf{x}_B$ para selecionar a coluna a sair da base;
- *Os valores das variáveis básicas*, para que possamos reportar a solução ótima quando a encontrarmos.

Tudo mais é desnecessário.

Como já fizemos antes na seção 4.2, representaremos o tableau simplex dividindo A, $\mathbf{x}$ e $\mathbf{c}$ em duas partes cada um – uma parte relativa às variáveis básicas e outra relativa às não básicas. Como denotamos

$$\begin{aligned} A &= \begin{pmatrix} A_B & A_N \end{pmatrix} \\ c^T &= \begin{pmatrix} \mathbf{c}_B^T & \mathbf{c}_N^T \end{pmatrix} \\ x &= \begin{pmatrix} \mathbf{x}_B & \mathbf{x}_N \end{pmatrix}, \end{aligned}$$

a descrição de problemas de programação linear pode ser feita da seguinte maneira:

$$\begin{aligned} \max\ & \mathbf{c}_B^T x_B + \mathbf{c}_N^T x_N \\ \text{s.a.:}\ & A_B \mathbf{x}_B + A_N \mathbf{x}_N \leq \mathbf{b} \\ & \mathbf{x}_B, \mathbf{x}_N \geq 0. \end{aligned}$$

O tableau simplex é

$$\begin{pmatrix} A & \mathbf{b} \\ \mathbf{c}^T & 0 \end{pmatrix} = \begin{pmatrix} A_B & A_N & \mathbf{b} \\ \mathbf{c}_B^T & \mathbf{c}_N^T & 0 \end{pmatrix}$$

Usamos operações elementares para obter coeficientes 1 nas variáveis básicas, de forma que a base seja a identidade. Multiplicamos a parte superior por A_B^{-1}, obtendo

$$\left(\begin{array}{c|c|c} I & A_B^{-1}A_N & A_B^{-1}\mathbf{b} \\ \hline \mathbf{c}_B^T & \mathbf{c}_N^T & 0 \end{array}\right)$$

Finalmente, subtraímos da última linha $\mathbf{c}_B^T$ multiplicado pela parte superior:

$$\left(\begin{array}{c|c|c} I & A_B^{-1}A_N & A_B^{-1}\mathbf{b} \\ \hline 0 & \mathbf{c}_N^T - \mathbf{c}_B^T A_B^{-1}A_N & -\mathbf{c}_B^T A_B^{-1}\mathbf{b} \end{array}\right)$$

Observe na última linha que $\mathbf{r}_N^T = \mathbf{c}_N^T - \mathbf{c}_B^T A_B^{-1}A_N$ representa os coeficientes reduzidos de custo $(c_j - z_j)$. Já $-\mathbf{c}_B^T A_B^{-1}\mathbf{b}$ é igual a $-\mathbf{c}_B^T \mathbf{x}_B$, ou $-z_0$, e portanto o tableau é

$$\left(\begin{array}{c|c|c} I & A_B^{-1}A_N & A_B^{-1}\mathbf{b} \\ \hline 0 & \mathbf{r}_N^T & -z_0 \end{array}\right)$$

Temos no tableau original $n+1$ colunas, e cada vez que trocarmos uma variável da base, modificaremos todas elas.

O algoritmo para o método Simplex revisado guarda apenas A_B^{-1}, sem ter que guardar A_N, que pode ser muito grande. Podemos fazer Isso porque A_B^{-1} nos permite gerar qualquer informação do tableau que precisemos, inclusive os valores de c_j e x_j:

$$\left(\begin{array}{c|c} A_B^{-1} & A_B^{-1}\mathbf{b} \\ \hline -\mathbf{c}_B^T A_B^{-1} & -z_0 \end{array}\right)$$

Temos agora um tableau com $m+1$ colunas. Detalhamos a seguir as operações que o Simplex fará usando esta representação. Denotamos por a'_{iq} o i-ésimo elemento do vetor coluna $\mathbf{a}'_q$:

$$\mathbf{a}'_q = \begin{pmatrix} a_1 q \\ a_2 q \\ \vdots \\ a_m q \end{pmatrix}.$$

- *Coeficientes reduzidos de custo:* para a variável x_j, observamos que

$$c_j - z_j = c_j - \mathbf{c}_B^T A_B^{-1}\mathbf{a_j}.$$

- *A coluna da variável x_q a entrar na base:* basta escrevê-la usando a nova base, A_B^{-1}:

$$\mathbf{a}'_\mathbf{q} = A_B^{-1}\mathbf{a}_\mathbf{q}$$

- *Atualização do tableau:* para atualizar todos os valores, efetuamos pivoteamento em a'_{pq} (o elemento na p-ésima linha do vetor coluna $\mathbf{a}_\mathbf{q}$).

A seguir está o método Simplex revisado, em pseudocódigo.

```
calcule A_B^{-1}
repita:
    λ^T ← c_B^T A_B^{-1}
    r_N^T ← c_N − λ^T A_B^{-1}
    se r_N^T ≤ 0 PARE: x_B ótimo
    q ← escolhe_coluna_a_entrar()
    a'_q ← A_B^{-1} a_q
    se a'_q ≤ 0 PARE: problema ilimitado
    calcule b_i/a'_iq e determine a_p
    atualize A_B^{-1} e a solução A_B^{-1} b (pivoteamento)
```

Quando usamos o método Simplex revisado, o tableau pode ser representado da seguinte maneira:

$$\mathbf{v}\left(A_b^{-1} \;\vdots\; \mathbf{b}\right)$$

onde $\mathbf{v}$ é a sequência de índices das variáveis básicas.

Exemplo 4.31. Para o problema

$$\begin{aligned} \max\ & x_1 + 2x_2 + 1x_3 \\ \text{s.a.:}\ & 4x_1 + 6x_2 + 3x_3 \leq 60 \\ & x_1 + x_2 + x_3 \leq 10 \\ & 2x_1 + x_2 + 2x_3 \leq 40 \\ & \mathbf{x} \geq 0 \end{aligned}$$

o tableau inicial é

$$\begin{matrix} 4 \\ 5 \\ 6 \end{matrix}\begin{pmatrix} 1 & 0 & 0 & \vdots & 60 \\ 0 & 1 & 0 & \vdots & 10 \\ 0 & 0 & 1 & \vdots & 40 \end{pmatrix}$$

As variáveis na base são as de folga, x_4, x_5, x_6. A identidade é a inversa da base, e o vetor $\mathbf{b}$ é o mesmo da descrição do problema.

O tableau final para o mesmo problema é

$$\begin{array}{c} 2 \\ 1 \\ 6 \end{array} \left(\begin{array}{ccc|c} 1/2 & -2 & 0 & 10 \\ -1/2 & 3 & 0 & 0 \\ 1/2 & -4 & 1 & 30 \end{array}\right)$$

Neste tableau, as variáveis básicas são $(2, 1, 6)$ (nesta ordem). A inversa da base é a submatriz do lado esquerdo, e $\mathbf{b}$ é $(10, 0, 30)^T$.

Note que as colunas mostradas são as mesmas que estariam no tableau do é o mesmo do problema originalmétodo Simplex (não revisado), nas é o mesmo do problema originalposições que usamos inicialmente. A é o mesmo do problema originalseguir mostramos o tableau final em como seria descrito usando o método Simplex não revisado.

$$\left(\begin{array}{ccc|ccc|c} 0 & 1 & -1/2 & 1/2 & -2 & 0 & 10 \\ 1 & 0 & 2/3 & -1/2 & 3 & 0 & 0 \\ 0 & 0 & -1/2 & 1/2 & -4 & 1 & 30 \end{array}\right)$$

A inversa da base que representamos no tableau final do Simplex revisado também aparece nas colunas $4, 5, 6$ do tableau Simplex (não revisado), como destacado na figura. ◀

No exemplo a seguir resolvemos um problema usando o métoro Simplex revisado.

Exemplo 4.32. Resolveremos o problema

$$\begin{aligned} \max\ & x_1 + x_2 \\ \text{s.a.:}\ & 2x_1 + x_2 \leq 40 \\ & x_1 + 2x_2 \leq 50 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

$$\begin{array}{c} 3 \\ 4 \end{array} \left(\begin{array}{cc|c} 1 & 0 & 40 \\ 0 & 1 & 50 \end{array}\right)$$

$$\begin{aligned} \lambda^T &= c_B^T A_B^{-1} = (0,0)A_B^{-1} = (0,0) \\ \mathbf{r_N} &= \mathbf{c_N}^T - \lambda^T A_N = \mathbf{c_N}^T \\ &= (1,1). \end{aligned}$$

Podemos incluir qualquer uma das variáveis. Escolhemos x_1. Geramos a coluna de x_1:

$$a_1' = A_B^{-1} a_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}.$$

Continuamos,

$$\begin{array}{c} 3 \\ 4 \end{array} \left(\begin{array}{cc|c} 1 & 0 & 40 \\ 0 & 1 & 50 \end{array}\right) \begin{pmatrix} 2 \\ 1 \end{pmatrix} \qquad \begin{array}{l} 40 \div 2 = 20 \Leftarrow \\ 50 \div 1 = 50 \end{array}$$

A primeira variável da base (x_3) sairá. Neste mesmo tableau, trocamos o índice de x_3 pelo de x_1, e pivoteamos para transformar a coluna de x_1 em $(0,1)^T$.

$$\begin{array}{c} 1 \\ 4 \end{array} \left(\begin{array}{cc|c} 1/2 & 0 & 20 \\ -1/2 & 1 & 30 \end{array}\right) \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad \begin{array}{l} L_1 \div 2 \\ L_2 - (1/2)L_1 \end{array}$$

Agora descartamos a coluna a_1, e calculamos os coeficientes reduzidos de custo.

$$\begin{array}{c} 1 \\ 4 \end{array} \left(\begin{array}{cc|c} 1/2 & 0 & 20 \\ -1/2 & 1 & 30 \end{array}\right) \qquad \begin{array}{l} \lambda^T = c_B^T A_B^{-1} = (1,0) A_B^{-1} = (1/2, 0) \\ \mathbf{r_N} = \mathbf{c_N}^T - \lambda^T A_N \\ \quad = (1/2, -1/2). \end{array}$$

Devemos incluir a primeira não básica, x_2. Geramos sua coluna:

$$a_2' = A_B^{-1} a_2 = \begin{pmatrix} 1/2 & 0 \\ -1/2 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 1/2 \\ 3/2 \end{pmatrix}$$

Escrevemos a coluna ao lado do tableau, e verificamos que coluna deve sair:

$$\begin{array}{c} 1 \\ 4 \end{array} \left(\begin{array}{cc|c} 1/2 & 0 & 20 \\ -1/2 & 1 & 30 \end{array}\right) \begin{pmatrix} 1/2 \\ 3/2 \end{pmatrix} \qquad \begin{array}{l} 20 \div 1/2 = 40 \\ 30 \div 3/2 = 20 \Leftarrow \end{array}$$

A segunda variável da base, x_4, sairá. Pivoteamos para transformar a_2 em $(0,1)$.

$$\begin{array}{c} 1 \\ 4 \end{array}\left(\begin{array}{cc|c} 2/3 & -1/3 & 10 \\ -1/3 & 2/3 & 20 \end{array}\right)\begin{pmatrix} 0 \\ 1 \end{pmatrix} \qquad \begin{array}{l} L_2 \times 2/3 \\ L_1 - (1/3)L_2 \end{array}$$

Calculamos os coeficientes reduzidos de custo.

$$\begin{array}{c} 1 \\ 2 \end{array}\left(\begin{array}{cc|c} 2/3 & -1/3 & 10 \\ -1/3 & 2/3 & 20 \end{array}\right) \qquad \begin{aligned} \lambda^T &= c_B^T A_B^{-1} = (1,1)A_B^{-1} \\ &= (1/3, 1/3) \\ \mathbf{r_N} &= \mathbf{c_N}^T - \lambda^T A_N \\ &= (0,0) - (1/3, 1/3) \\ &= (-1/3, -1/3). \end{aligned}$$

Temos finalmente a solução ótima, com

$$\begin{aligned} x_1 &= 10 \\ x_2 &= 20. \end{aligned}$$ ◀

É comum que implementações não armazenem A_B^{-1} explicitamente, mas sim por sua fatoração LU. A fatoração LU de A_B^{-1} é apenas atualizada, sendo recalculada apenas de tempos em tempos. Isso traz um equilíbrio entre eficiência e precisão numérica.

Exercícios

Ex. 29 — Resolva os problemas a seguir usando o método Simplex. Resolva cada um usando a regra de Bland e também alguma outra regra.

(i) $$\begin{aligned} \max\ & x_1 - x_2 + 3x_3 \\ \text{s.a.}: & \\ & x_1 - x_2 \leq 4 \\ & 2x_1 + x_3 \leq 6 \\ & 3x_2 + \frac{1}{2}x_3 \leq 5 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

(ii) $$\begin{aligned} \max\ & x_1 + 2x_2 \\ \text{s.a.}: & \\ & x_1 + 2x_2 \leq 1 \\ & x_1 \leq 1 \\ & x_2 \leq 1 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

(iii) $\max x_1 + x_2 - 2x_3$
$\text{s.a.} : x_1 + x_2 \leq 5$
$x_2 + 2x_3 \leq 5$
$2x_1 - x_2 + x_3 \leq 9$
$\mathbf{x} \geq \mathbf{0}$

(iv) $\max 3x_1 + 4x_2 + 2x_3$
$\text{s.a.} : x_1 - x_2 \leq 8$
$x_1 + x_2 + x_3 = 11/2$
$-x_1 + 2x_2 \leq 10$
$\mathbf{x} \geq \mathbf{0}$

(v) $\max x_1 + x_2$
$\text{s.a.} : -x_1 + x_2 \leq 1/2$
$-\frac{2}{3}x_1 + x_2 \leq 2/3$
$x_2 \leq 3/2$
$\mathbf{x} \geq \mathbf{0}$

(vi) $\max x_1 - 2x_2 + x_3 - 3x_4 + x_5$
$\text{s.a.} : 2x_1 + x_3 - x_4 + x_5 \leq 22$
$-3x_2 + x_3 \leq 10$
$\mathbf{x} \geq \mathbf{0}$

(vii) $\max x_1 - x_2$
$\text{s.a.} : x_1 + x_2 \leq 4$
$x_1 - x_2 \leq 1$
$-2x_1 + 3x_2 \leq 2$
$\mathbf{x} \geq \mathbf{0}$

(viii) $\max x_1 - 2x_2 + x_3$
$\text{s.a.} : x_1 + x_2 \leq 10$
$x_1 - x_2 \leq 1$
$x_1 - 2x_2 \leq 6$
$\mathbf{x} \geq \mathbf{0}$

Ex. 30 — Coloque os problemas na forma padrão para que possam ser resolvidos pelo método Simplex, e depois use o Simplex para resolvê-los, mostrando os tableaux intermediários.

(i) $\max 2x_1 + x_2 + 3x_3$
$\text{s.a.} : x_1 - x_2 = 5$
$2x_1 + x_3 = 10$
$x_2 + 2x_3 \leq 9$
$\mathbf{x} \geq \mathbf{0}$

(ii) $\max 3x_1 + 2x_2$
$\text{s.a.} : x_1 \geq 1$
$x_1 \leq 2$
$x_2 \geq 1$
$x_2 \leq 2$
$\mathbf{x} \geq \mathbf{0}$

$$\begin{array}{rl} (iii) & \max 2x_1 + x_2 - x_3 \\ & \text{s.a.} : x_1 + x_2 \leq 3 \\ & \quad 2x_2 + x_3 = 5 \\ & \quad x_1 - 3x_3 = 4 \\ & \quad \mathbf{x} \geq \mathbf{0} \end{array} \qquad \begin{array}{rl} (iv) & \max 2x_1 + x_2 \\ & \text{s.a.} : x_1 + x_2 \geq 3 \\ & \quad 2x_2 + x_3 = 5 \\ & \quad \mathbf{x} \geq \mathbf{0} \end{array}$$

$$\begin{array}{rl} (v) & \max x_1 + x_2 - x_3 \\ & \text{s.a.} : x_1 + x_2 \leq 7 \\ & \quad x_1 - x_2 \leq 6 \\ & \quad 2x_1 + 2x_2 \geq 6 \\ & \quad x_1 + x_3 = 7 \\ & \quad \mathbf{x} \geq \mathbf{0} \end{array}$$

Ex. 31 — Coloque os problemas na forma padrão para que possam ser resolvidos pelo método Simplex, e depois use o Simplex para resolvê-los, mostrando os tableaux intermediários. Use o método das duas fases em dois problemas, e o do M grande em dois outros.

$$\begin{array}{rl} (i) & \min 2x_1 + 4x_2 + 8x_3 \\ & \text{s.a.} : x_1 + x_2 = 5 \\ & \quad x_1 + x_3 \geq 4 \\ & \quad 3x_2 + x_3 \geq 5 \\ & \quad \mathbf{x} \geq \mathbf{0} \end{array} \qquad \begin{array}{rl} (ii) & \min 2x_1 - x_2 \\ & \text{s.a.} : x_1 \geq 2 \\ & \quad x_1 \leq 4 \\ & \quad x_2 \geq 2 \\ & \quad x_2 \leq 4 \\ & \quad \mathbf{x} \geq \mathbf{0} \end{array}$$

$$\begin{array}{rl} (iii) & \min x_1 - x_2 + 2x_3 \\ & \text{s.a.} : x_1 + x_2 \geq 5 \\ & \quad x_2 + 3x_3 = 8 \\ & \quad x_1 - x_3 \geq 4 \\ & \quad \mathbf{x} \geq \mathbf{0} \end{array} \qquad \begin{array}{rl} (iv) & \min 8x_1 + 3x_2 \\ & \text{s.a.} : 5x_1 - x_2 = 3 \\ & \quad x_2 + 3x_3 = 5 \\ & \quad \mathbf{x} \geq \mathbf{0} \end{array}$$

Ex. 32 — Implemente o método Simplex: primeiro, apenas para restrições na forma de desigualdade, $A\mathbf{x} \leq \mathbf{b}$. Depois permitindo igualdades, usando a técnica descrita neste Capítulo para obter soluções viáveis básicas iniciais. Posteriormente, experimente com diferentes métodos para escolher a coluna a entrar na base.

Ex. 33 — Demonstre rigorosamente os Corolários 4.6 e 4.7.

Ex. 34 — Demonstre a Proposição 4.8. É realmente necessário verificar *todos* os a_{ij} em A? Explique.

Ex. 35 — Seja $(x_1, x_2, \ldots, x_n)$ uma solução viável básica. Prove que

$$|x_j| \leq m!\alpha^{m-1}\beta,$$

onde

$$\alpha = \max_{i,j}\{|a_{i,j}|\}$$
$$\beta = \max_{j\leq m}\{|b_j|\}$$

Ex. 36 — Considere o problema

$$\max \mathbf{c}^T\mathbf{x} \text{ sujeito a } A\mathbf{x} = \mathbf{b},$$

com solução ótima de valor v. Suponha que o problema

$$\min \mathbf{c}^T\mathbf{x}, \text{ sujeito a } A\mathbf{x} = \mathbf{b}$$

tenha ótimo com o mesmo valor, v. Pode-se concluir que existe um único ponto ótimo para os dois? Como é, geometricamente, a região viável?

Ex. 37 — Prove que quando usamos o método Simplex em um problema não degenerado, uma coluna que sai da base não pode volar a ser básica na iteração seguinte.

Ex. 38 — Prove que se um problema tem região viável ilimitada, existe para ele uma função objetivo que não permite resolver o problema, e uma que permite obter uma solução ótima.

Ex. 39 — O que acontece quando a mesma solução viável básica pode ser descrita por mais de uma base?

Ex. 40 — O que acontece quando usamos o método do M grande em um problema inviável? E em um problema ilimitado?

Ex. 41 — Suponha que tenhamos usado o método das duas fases para resolver um problema. Se, após a fase I, uma variável artificial sai da base, ela pode novamente entrar?

Ex. 42 — Resolva, usando o método Simplex revisado:

$$\begin{aligned} \max\ & x_1 - x_2 + 2x_3 + 4x_4 + x_5 + 2x_6 + x_7 \\ \text{s.a.:}\ & x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \leq 50 \\ & 2x_1 - x_2 + 3x_3 + 2x_4 - x_5 - x_6 + 2x_7 \leq 40 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Capítulo 5

Dualidade

A todo problema de maximização pode-se associar um problema de minimização, chamado de *dual* do problema. Neste Capítulo estudamos a dualidade em programas lineares.

Definição 5.1 (Primal e dual)**.** Considere o seguinte programa linear:

$$\begin{aligned} \max\ & \mathbf{c}^T\mathbf{x} \\ \text{s.a.:}\ & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Damos o nome a este P.L. de *primal*, e dizemos que o P.L. a seguir é seu *dual*.

$$\begin{aligned} \min\ & \mathbf{b}^T\mathbf{y} \\ \text{s.a.:}\ & A^T\mathbf{y} \geq \mathbf{c} \\ & \mathbf{y} \geq \mathbf{0}. \end{aligned}$$

♦

Se um problema tem restrições em forma de igualdade, podemos construir seu dual primeiro passando o problema para a forma de desigualdade, observando que a restrição

$$a_{i1}x_1 + \ldots + a_{in}x_n = b_i$$

pode ser reescrita como

$$\begin{aligned} a_{i1}x_1 + \ldots + a_{in}x_n &\leq b_i \\ -a_{i1}x_1 - \ldots - a_{in}x_n &\leq -b_i. \end{aligned}$$

Exemplo 5.2. Determinaremos o dual do seguinte problema.

$$\begin{aligned} \max\ & x_1 + 2x_2 + x_3 \\ \text{s.a.:}\ & x_1 - 3x_2 = 5 \\ & -x_1 + 2x_2 - x_3 \leq 3 \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

A primeira restrição é uma igualdade, portanto a transformamos em duas desigualdades,

$$x_1 - 3x_2 = 5 \quad \Longrightarrow \quad \begin{cases} x_1 - 3x_2 & \leq 5 \\ -x_1 + 3x_2 & \leq -5, \end{cases}$$

obtendo

$$\begin{aligned} \max\ & x_1 + 2x_2 + x_3 \\ \text{s.a.:}\ & x_1 - 3x_2 \leq 5 \\ & -x_1 + 3x_2 \leq -5 \\ & -x_1 + 2x_2 - x_3 \leq 3 \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Agora podemos obter o dual do problema:

$$\begin{aligned} \min\ & 5y_1 - 5y_2 + 3y_3 \\ \text{s.a.:}\ & y_1 - y_2 - y_3 \geq 1 \\ & -3y_1 + 3y_2 + 2y_3 \geq 2 \\ & -y_3 \geq 1 \\ & \mathbf{y} \geq \mathbf{0}. \end{aligned}$$

No entanto, este programa linear tem tres variáveis, e o primal do qual partimos originalmente tem duas restrições apenas. ◀

O Exercício 49 pede a demonstração do Teorema 5.3, que consiste na formalização do que foi visto no exemplo anterior.

Teorema 5.3. *Considere um programa linear de* maximização. *É possível obter um programa linear dual a este, mas* com o número de variáveis do dual exatamente igual ao número de restrições do primal. *Se a* i*-ésima restrição do primal é uma*

- igualdade*: então a variável correspondente* y_i *no dual será livre (ou seja, poderá assumir valores negativos)*
- desigualdade $\geq$*: então a variável* y_i *do dual assumirá valores* ≤ 0.

5.1 Interpretações do dual

Há duas interpretações importantes para o dual de um programa linear: a operacional e a econômica.

5.1.1 Interpretação operacional

Os coeficientes reduzidos de custo (na última linha do tableau Simplex) representam, para cada variável não básica, o quanto poderíamos melhorar a função objetivo se ela fosse incluída na base. Esses coeficientes mudam à medida que o Simplex tenta encontrar soluções *que se mantenham viáveis, mas mais próximas do ótimo*. O vetor $\mathbf{b}$ representa o limite da viabilidade – que não muda durante a execução do Simplex.

No problema dual, a função a ser otimizada é $\mathbf{b}^T\mathbf{y}$ – ou seja, procuramos otimizar a distância até a viabilidade. Já as restrições são da forma $A\mathbf{y} \gtreqless \mathbf{c}$ – e como $\mathbf{c}$ não muda, estamos mantendo a condição de otimalidade. Assim, o dual de um programa linear representa o problema de, *dada uma solução qualquer que tenha valor ótimo, encontrar aquela mais próxima possível da viabilidade, mantendo o mesmo valor*.

5.1.2 Interpretação econômica

Considere o problema de mistura a seguir: uma empresa produz refeições prontas para companhias aéreas. Em cada refeição podem ser usadas quantidades diferentes de alguns ingredientes.

Ao resolver o problema primal, maximizamos o lucro total obtido com a produção das refeições escolhendo quanto usar de cada ingrediente. Resolvendo o problema dual, minimizamos o custo de cada recurso (e para isso diminuímos a quantidade usada, claro), de forma a manter o valor ótimo de lucro.

Em outras palavras, resolver o primal significa partir de quantidades de recursos menor que a ótima e aumentá-las até chegar ao ótimo, nunca usando mais recursos que o possível; resolver o dual significa partir de uma quantidade de recursos maior que o disponível, e diminuí-las (diminuindo o custo) até chegar ao mínimo possível.

5.2 Lema de Farkas

Nesta seção tratamos do Lema de Farkas. Este lema é usado em diversas ocasiões no desenvolvimetno de tópicos relacionados à dualidade. Neste capítulo, ele será usado na demonstração de um teorema que relaciona

a viabilidade do dual com a característica de limitação do primal (Teorema 5.16).

Para uma melhor compreensão deste Lema, usaremos as definições de combinaçõs positivas e cones gerados por vetores.

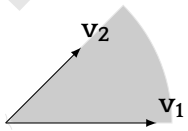
Definição 5.4 (combinação positiva)**.** Uma *combinação positiva* de um conjunto de vetores é uma combinação linear destes vetores, tendo coeficientes não negativos. ♦

Definição 5.5 (cone convexo)**.** O *cone convexo* gerado por um conjunto de vetores é o conjunto de todas as combinações positivas daquele conjunto. ♦

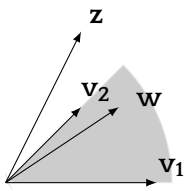
Exemplo 5.6. Seja $V = \{(2,0)^T, (1,1)^T\}$. O cone gerado por estes dois vetores é

$$\begin{aligned}\left\{a(2,0) + b(1,1) : \quad a,b \geq 0\right\} &= \left\{(2a,0)^T + (b,b)^T : \quad a,b \geq 0\right\} \\ &= \left\{(2a+b,b)^T : \quad a,b \geq 0\right\} \\ &= \left\{(x,y)^T : \quad x > y \geq 0\right\}.\end{aligned}$$

Os dois vetores são mostrados na figura a seguir; o cone formado por eles é composto de todos os vetores na área sombreada.



Observe que para descrever um vetor de $\mathbb{R}^2$ fora do cone como combinação linear de v_1 e v_2, teríamos que usar coeficientes negativos:



O vetor $\mathbf{w} = (1.5, 1)$ está contido no cone, e sua descrição é

$$\mathbf{w} = \frac{1}{4}(2,0)^T + (1,1)^T,$$

com coeficientes positivos $1/4, 1$. Já o vetor fora do cone, $(1,2)^T$ é

$$\mathbf{z} = -\frac{1}{2}(2,0)^T + 2(1,1)^T,$$

com um coeficiente negativo. ◀

Relembramos que a multiplicação de uma matriz por um vetor coluna descreve a combinação linear das colunas com os coeficientes dados no vetor:

$$\begin{aligned} A\mathbf{x} &= \left(\mathbf{a}^1, \mathbf{a}^2, \ldots, \mathbf{a}^n\right) \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \\ &= x_1\mathbf{a}^1 + x_2\mathbf{a}^2 + \cdots + x_n\mathbf{a}^n. \end{aligned}$$

Assim, o cone gerado pelas colunas de A é o conjunto de combinações positivas de suas colunas, ou

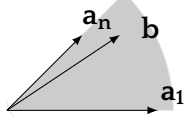
$$\left\{A\mathbf{x}, \quad \mathbf{x} \geq 0\right\}.$$

O Lema de Farkas (Lema 5.7) representa a essência da dualidade em programação linear. O Lema diz, geometricamente, que dados n vetores, um vetor qualquer x pode estar dentro do cone (e neste caso será combinação *positiva* dos outros) ou fora do cone (quando é combinação *não positiva* dos outros) – e nunca ambos ao mesmo tempo.

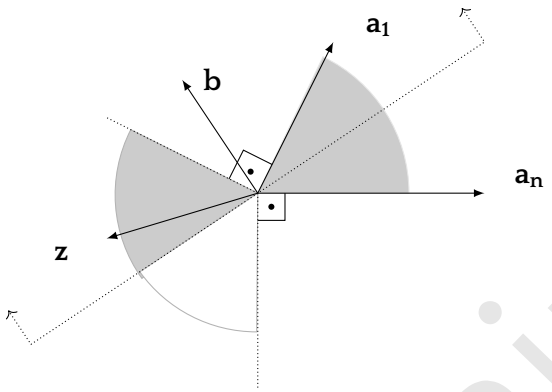
Em essencia, o Lema de Farkas diz que o vetor $\mathbf{b}$ pode pertencer ao cone gerado por A ou não.

Seja S a revião viável, definida por um programa linear $A\mathbf{x} = b$. Então *exatamente* uma das duas situações ocorre:

i) $\mathbf{b}$ pertence ao cone gerado por A, portanto temos $A\mathbf{x} = \mathbf{b}$, com $\mathbf{x} \geq \mathbf{0}$ (porque para pertencer ao cone deve ser combinação positiva das colunas de A).



ii) $\mathbf{b}$ não pertence ao cone gerado por A. Neste caso, deve haver algum $\mathbf{z}$, *formando ângulo menor que* 90^o *com* $\mathbf{b}$, mas formando mais de 90^o com o cone definido por A.



Mostramos na figura apenas a_1 e a_n; os outros vetores a_i estão entre eles; o semiespaço determinado pela linha leve consiste dos vetores com ângulo menor que 90º com $\mathbf{b}$; o cone à esquerda e abaixo é formado pelos vetores com ângulo maior ou igual que 90º com o cone de A.

Assim, existe $\mathbf{z}$ tal que

- o ângulo entre colunas de A e $\mathbf{z}$ é maior que 90º, logo $A^T\mathbf{z} \leq \mathbf{0}$;
- o ângulo entre b e z é menor que 90º, logo $\mathbf{b}^T\mathbf{z} > 0$.

Se tomarmos $\mathbf{y} = -\mathbf{z}$, teremos então $A^T\mathbf{y} \geq \mathbf{0}$ para algum $\mathbf{y}$ tal que $\mathbf{b}^T\mathbf{y} < 0$.

Embora a discussão geométrica até este ponto possa ser convincente para o caso em que tratamos de vetores em $\mathbb{R}^2$, enunciamos e demonstramos algebricamente a seguir o Lema de Farkas.

Lema 5.7 (de Farkas). *Sejam A uma matriz e $\mathbf{b}$ um vetor, sendo que o número de linhas de A é igual à quantidade de elementos de $\mathbf{b}$. Então exatamente um dos dois sistemas a seguir tem solução.*

i) $A\mathbf{x} = \mathbf{b}$, *para algum* $\mathbf{x} \geq \mathbf{0}$.

ii) $\mathbf{y}^T A \geq \mathbf{0}$ *para algum* $\mathbf{y}$ *tal que* $\mathbf{b}^T\mathbf{y} < 0$.

Esta demonstração está incompleta. Outra demonstração diferente será usada em uma futura versão do texto.

Demonstração. (i $\Rightarrow$ não ii) Suponha que (i) valha, e $A\mathbf{x} = \mathbf{b}$ tenha solução positiva. Se existe $\mathbf{y}$ tal que $\mathbf{y}^T A \geq \mathbf{0}$, então

$$\begin{aligned} \mathbf{y}^T A &\geq \mathbf{0}^T \\ \mathbf{y}^T A\mathbf{x} \geq \mathbf{0}^T\mathbf{x} &= 0 && \text{(porque } \mathbf{x}, A\mathbf{y} \geq \mathbf{0}) \\ \mathbf{y}^T\mathbf{b} &\geq 0 && (A\mathbf{x} = \mathbf{b}) \end{aligned}$$

e o sistema (ii) não pode ter solução.

(não i $\Rightarrow$ ii) – esta parte é omitida por ora. ■

Fica claro, do enunciado do Lema de Farkas, que ele tem forte relação com o conceito de dualidade – o produto $A^T\mathbf{y}$, com $\mathbf{y}$ não restrito a positivos, é parte da descrição do dual de $A\mathbf{x} = \mathbf{b}$.

Exemplo 5.8. Sejam

$$A = \begin{pmatrix} 1 & 3 & 2 \\ 0 & -1 & 4 \\ 2 & 1 & 5 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 3 \\ 5 \\ 1 \end{pmatrix}.$$

O sistema $A\mathbf{x} = \mathbf{b}$ só tem a solução $\mathbf{x} = (3, -5, 30/19)^T$, portanto o sistema $A^T\mathbf{y} \geq \mathbf{0}$ deve ter solução para algum $\mathbf{y}$ com $\mathbf{b}^T\mathbf{y} < 0$. E realmente, a solução $\mathbf{y} = (2, -3, 4)^T$ nos dá

$$A^T\mathbf{y} = (10, 13, 12)^T, \mathbf{yb} = 3(2) + 5(-3) + 1(4) = -5. \quad ◀$$

Exemplo 5.9. O Lema de Farkas vale para quaisquer matrizes, não apenas quadradas. Por exemplo, sejam

$$A = \begin{pmatrix} 1 & 3 & 1 \\ 0 & 1 & 2 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 4 \\ 1 \end{pmatrix}.$$

As soluções deste sistema sào da forma

$$x_1 - 2x_3 = 1,$$

o que inclui a solução $\mathbf{x} = 5, 2)^T$. O sistema $A^T\mathbf{y} \geq \mathbf{0}$ portanto não pode ter solução com $\mathbf{y}^T\mathbf{b} < 0$. Realmente,

$$\begin{pmatrix} 1 & 0 \\ 3 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

só tem a solução trivial, com $\mathbf{y} = \mathbf{0}$. ◀

5.3 Teoremas de dualidade

As soluções ótimas para o primal e o dual tem o mesmo valor, como mostraremos nos próximos teoremas.

Teorema 5.10 (dualidade fraca)**.** *Sejam* $\mathbf{x}$ *e* $\mathbf{y}$ *soluções para o primal e o dual de um programa linear. Então* $\mathbf{c}^T\mathbf{x} \leq \mathbf{b}^T\mathbf{y}$*.*

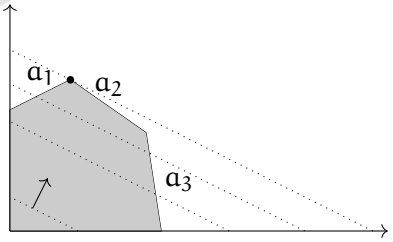
Demonstração.

$$\begin{aligned} \mathbf{c} &\leq A^T\mathbf{y} \\ \mathbf{c}^T &\leq \mathbf{y}^T A \\ \mathbf{c}^T\mathbf{x} &\leq \mathbf{y}^T A\mathbf{x} \\ &\leq \mathbf{y}^T\mathbf{b} && \text{(somente porque } \mathbf{x}, \mathbf{y}, \mathbf{b} \geq 0\text{)} \\ &= \mathbf{b}^T\mathbf{y} && \blacksquare \end{aligned}$$

Corolário 5.11. *Se* $\mathbf{x_0}$ *e* $\mathbf{y_0}$ *são soluções para o primal e o dual, e* $\mathbf{c}^T\mathbf{x_0} = \mathbf{b}^T\mathbf{y_0}$ *então ambas são soluções ótimas.*

Demonstração. Seja $\mathbf{x}$ solução viável para o primal. Então $\mathbf{c}^T\mathbf{x} \leq \mathbf{b}^T\mathbf{y}_0 = \mathbf{c}^T\mathbf{x}_0$. ■

As restrições de um programa linear $\max \mathbf{c}^T\mathbf{x}$, s.a $A\mathbf{x} \leq \mathbf{b}$ são da forma $a_i\mathbf{x} \leq b_i$, onde a_i é a i-ésima linha de A. Estas restrições podem ser visualizadas como hiperplanos, cada um definindo um semiespaço. A solução ótima está exatamente na interseção das restrições.



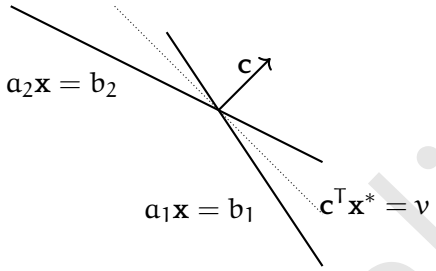
Na figura anterior, a solução ótima é a interseção das restrições $a_1\mathbf{x} = b_1$ e $a_2\mathbf{x} = b_2$, e a_3 é redundante.

Lema 5.12. *Se o ponto ótimo de um programa linear não pertence ao hiperplano definido por uma das restrições, ela é redundante e pode ser removida sem que a solução ótima mude.*

Teorema 5.13 (dualidade forte)**.** *Se o primal ou o dual de um programa linear tem solução ótima, o outro também tem, e os valores das soluções ótimas são iguais.*

Demonstração. Sejam $\max \mathbf{c}^T\mathbf{x}$ s.a $A\mathbf{x} \leq \mathbf{b}$ o primal de um programa linear e $\min \mathbf{b}^T\mathbf{y}$ s.a $A^T\mathbf{y} \leq \mathbf{c}$ seu dual. Suponha que o primal tem solução ótima $\mathbf{x}^*$, com valor $v = \mathbf{c}^T\mathbf{x}^*$.

O hiperplano $\mathbf{c}^T\mathbf{x}^* = v$ toca no poliedro apenas no ponto ótimo $\mathbf{x}^*$ (ou nos pontos ótimos, se houver mais de um).



Aqui denotamos por a_i a i-ésima linha de A. Sejam $a_1\mathbf{x} \leq b_1$, $a_2\mathbf{x} \leq b_2$, ... as restrições do primal, e considere os hiperplanos $a_1\mathbf{x} = b_1$, $a_2\mathbf{x} = b_2$, que definem a borda do poliedro. Já o hiperplano $\mathbf{c}^T\mathbf{x}^* = v$, ortogonal a $\mathbf{c}$ e que toca o poliedro em $\mathbf{x}^*$, é claramente uma combinação linear não negativa dos hiperplanos definidos pelos $a_i\mathbf{x} = b_i$. Temos portanto

$$\begin{array}{rcl} \lambda_1(a_1\mathbf{x}) & & \lambda_1 b_1 \\ +\lambda_2(a_2\mathbf{x}) & & +\lambda_2 b_2 \\ +\lambda_3(a_3\mathbf{x}) & = & +\lambda_3 b_3 \\ \vdots & & \vdots \\ \sum \lambda_i a_i\mathbf{x} & = & \sum \lambda_i b_i \end{array}$$

E consequentemente,

$$\begin{aligned} \mathbf{c}^T &= \sum \lambda_i a_i \\ v &= \sum \lambda_i b_i \end{aligned}$$

Então,

$$\max\left\{\mathbf{c}^T\mathbf{x} | A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\right\} = v = \sum \lambda_i b_i,$$

mas temos também

$$\sum \lambda_i b_i \geq \min\left\{\mathbf{b}^T\mathbf{y} | A^T\mathbf{y} \geq \mathbf{c}, \mathbf{y} \geq \mathbf{0}\right\},$$

e o valor máximo para o primal é maior ou igual que o valor mínimo para o dual.

Mostramos que $\mathbf{c}^T\mathbf{x} \geq \mathbf{b}^T\mathbf{y}$, mas também sabemos pelo teorema 5.10 que $\mathbf{c}^T\mathbf{x} \leq \mathbf{b}^T\mathbf{y}$, e portanto os dois valores devem ser iguais. ■

Exemplo 5.14. Temos a seguir um par primal/dual de problemas:

$$\begin{array}{ll} \max x_1 + 3x_2 & \min 6y_1 + 9y_2 \\ \text{s.a.: } x_1 + x_2 \leq 6 & \text{s.a.: } y_1 + 2y_2 \geq 1 \\ \quad 2x_1 - x_2 \leq 9 & \quad y_1 - y_2 \geq 3 \\ \quad \mathbf{x} \geq 0 & \quad \mathbf{y} \geq 0 \end{array}$$

A solução ótima para o *primal* é $x_1 = 0, x_2 = 6$, com valor 18. A solução ótima para o *dual* é $y_1 = 3, y_2 = 0$, com valor 18. ◀

Exemplo 5.15 (Multiplicadores Simplex)**. este exemplo está no llugar errado...** Considere o exemplo 5.14. Se usarmos os valores ótimos do dual como multiplicadores para as restrições do primal, temos

$$\begin{aligned} &3\Big(x_1 + x_2 \leq 6\Big) \\ &+0\Big(2x_2 - x_2 \leq 9\Big) \\ =&3x_1 + 3x_2 \leq 18 \\ \Rightarrow&x_1 + 3x_2 \leq 18, && \text{(porque } x_1 \geq 0\text{)} \end{aligned}$$

confirmando a otimalidade da solução: se obtivemos esta desigualdade somente fazendo combinação linear de restrições, ela é válida – mas ela diz exatamente que *a função objetivo,* $x_1 + 3x + 2$*, não pode ter valor maior que* 18.

O mesmo vale se multiplicarmos as restrições do dual pelos valores ótimos do primal:

$$\begin{aligned} &0\Big(y_1 + 2y_2 \geq 1\Big) \\ &+6\Big(y_1 - y_2 \geq 3\Big) \\ =&6y_1 - 6y_2 \geq 18 \\ \Rightarrow&6y_1 + 9y_2 \geq 18 && \text{(porque } y_2 \geq 0\text{)} \end{aligned}$$

O que a última desigualdade nos diz é que a função objetivo do dual não pode ter valor menor que 18. ◀

Teorema 5.16. *Se o primal de um programa linear é viável e seu dual não é então o primal é ilimitado.*

Demonstração. Seja $\mathbf{x}$ viável para o primal. Como o dual é inviável, o sistema a seguir não tem solução.

$$\begin{aligned} A^T\mathbf{y} &\geq \mathbf{c} \\ \mathbf{y} &\geq 0 \end{aligned}$$

Como não há solução para este sistema, pelo Lema de Farkas deve existir solução para

$$\begin{aligned} A\mathbf{z} &\geq 0 \\ \mathbf{c}^T\mathbf{z} &> 0 \\ \mathbf{z} &\geq 0 \end{aligned}$$

Tomemos um $\mathbf{z}$, solução do sistema acima. Observamos que $\mathbf{z}+\omega$ é viável para o primal se $\omega \geq 0$:

$$A(\mathbf{x} + \omega\mathbf{z}) = A\mathbf{x} + \omega A\mathbf{z}$$

e como $\omega A\mathbf{z} \leq 0$, $\mathbf{x} + \omega\mathbf{z} \leq b$. Mas temos também

$$\mathbf{c}^T(\mathbf{x} + \omega\mathbf{z}) = \mathbf{c}^T\mathbf{x} + \omega\mathbf{c}^T\mathbf{z},$$

e como $\omega > 0$, $\mathbf{c}^T\mathbf{z} > 0$, temos que $\mathbf{c}^T(\mathbf{x} + \omega\mathbf{z}) \to \infty$ quando $\omega \to \infty$, e $\mathbf{x}$ não é ótimo. Mais ainda, para qualquer $\mathbf{x}$ supostamente ótimo podemos determinar $\mathbf{x}'$ dando valor maior para o objetivo. ■

Corolário 5.17. *Se o primal de um programa linear é viável e* $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$ *é limitada por cima então o dual é viável.*

Se o dual de um programa linear é viável e $g(\mathbf{y}) = \mathbf{b}^T\mathbf{y}$ *é limitada por baixo então o primal é viável.*

Se tanto o primal como o dual forem viáveis, tanto f *como* g *são limitadas,* f *por cima e* g *por baixo.*

Teorema 5.18. *Se tanto o primal como o dual são viáveis, ambos tem (a mesma) solução ótima.*

Observamos também que o tableau simplex para o dual é exatamente o tableau transposto do primal:

$$\begin{pmatrix} A_B & A_N & \mathbf{b} \\ \mathbf{c}_B^T & \mathbf{c}_N^T & \end{pmatrix} \longrightarrow \begin{pmatrix} A_B^T & \mathbf{c}_B \\ A_N^T & \mathbf{c}_N \\ \mathbf{b}^T & \end{pmatrix}$$

A seguir enunciamos o Teorema das folgas complementares, que usaremos no desenvolvimento dos algoritmos Simplex-dual e primal-dual. Informalmente, o Teorema determina que quando há solução ótima para um par de problemas duais, se a i-ésima variável da solução do primal é não zero, a solução do dual torna sua i-ésima linha uma igualdade (sem folga); e quando a i-ésima linha do dual é desigualdade estrita, a i-ésima variável do primal tem valor zero. Ou seja, se $x_j \neq 0$, então a j-ésima linha do dual é satisfeita sem folga, $(\mathbf{a_j})^T\mathbf{y} = c_j$.

Teorema 5.19 (das folgas complementares)**.** *Sejam um programa linear e seu dual,*

$$\begin{aligned} &\max \mathbf{c}^T\mathbf{x}, \quad \textit{s.a.: } A\mathbf{x} \leq \mathbf{b} \\ &\min \mathbf{b}^T\mathbf{y}, \quad \textit{s.a.: } A^T\mathbf{y} \geq \mathbf{c} \end{aligned}$$

Duas soluções $\mathbf{x}$ *e* $\mathbf{y}$ *para o primal e dual são ótimas se e somente se*

$$\begin{aligned} \mathbf{y}^T(A\mathbf{x} - \mathbf{b}) &= \mathbf{0} \\ \mathbf{x}^T(\mathbf{c} - A^T\mathbf{y}) &= \mathbf{0}. \end{aligned}$$

Demonstração. Sejam $\mathbf{x}$ e $\mathbf{y}$ soluções viáveis para o primal e dual. Temos

$$\begin{aligned} A\mathbf{x} &\leq \mathbf{b} \\ \mathbf{b} - A\mathbf{x} &\geq \mathbf{0} \\ \mathbf{y}^T(\mathbf{b} - A\mathbf{x}) &\geq \mathbf{0}, \end{aligned}$$

porque $\mathbf{y}$ é viável ($\mathbf{y} \geq \mathbf{0}$). Similarmente,

$$\begin{aligned} A^T\mathbf{y} &\geq \mathbf{c} \\ A^T\mathbf{y} - \mathbf{c} &\geq \mathbf{0} \\ \mathbf{x}^T(A^T\mathbf{y} - \mathbf{c}) &\geq \mathbf{0}. \end{aligned}$$

Sejam $p = \mathbf{y}^T(\mathbf{b} - A\mathbf{x})$ e $q = \mathbf{x}^T\left(A^T\mathbf{y} - \mathbf{c}\right)$. Então,

$$\begin{aligned} p + q &= \mathbf{y}^T(\mathbf{b} - A\mathbf{x}) + \mathbf{x}^T\left(A^T\mathbf{y} - \mathbf{c}\right) \\ &= \mathbf{y}^T\mathbf{b} + \left(-\mathbf{y}^TA\mathbf{x} + \mathbf{x}^TA^T\mathbf{y}\right) - \mathbf{x}^T\mathbf{c} \\ &= \mathbf{y}^T\mathbf{b} - \mathbf{x}^T\mathbf{c} \\ &\geq 0, \end{aligned}$$

porque $\mathbf{y}^T\mathbf{b} \geq \mathbf{x}^T\mathbf{c}$, pelo Teorema 5.10.

($\Rightarrow$) Se as duas soluções forem ótimas, temos $p + q \geq 0$ e também $p = q$, e necessariamente $p = q = 0$. Portanto,

$$\mathbf{y}^T(\mathbf{b} - A\mathbf{x}) = 0$$
$$\mathbf{x}^T\left(A^T\mathbf{y} - \mathbf{c}\right) = 0$$

($\Leftarrow$) Suponha $p = q = 0$. Então

$$\mathbf{y}^T(\mathbf{b} - A\mathbf{x}) + \mathbf{x}^T\left(A^T\mathbf{y} - \mathbf{c}\right) = 0$$
$$\mathbf{y}^T\mathbf{b} - \mathbf{x}^T\mathbf{c} = 0$$
$$\mathbf{y}^T\mathbf{b} = \mathbf{x}^T\mathbf{c}$$

e, como as soluções tem o mesmo valor objetivo e são viáveis para o primal e para o dual, são ambas ótimas. ■

5.4 Algoritmo simplex dual

Considere o primal e o dual de um programa linear, como os que são dados a seguir.

$$\begin{aligned} \text{maximize } & \mathbf{c}^T\mathbf{x} \\ \text{s.a.: } & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

$$\begin{aligned} \text{minimize } & \mathbf{b}^T\mathbf{y} \\ \text{s.a.: } & A^T\mathbf{y} \geq \mathbf{c} \\ & \mathbf{y} \geq \mathbf{0}. \end{aligned}$$

O método Simplex, quando aplicado ao primal, procura por várias soluções *viáveis* (sempre obedecendo as restrições) e básicas, cada uma com valor objetivo maior, até encontrar a solução ótima. Observando a formulação do dual, podemos imaginar que se executarmos nele o mesmo algoritmo, o que sempre se mantém é a condição de otimalidade (porque $\mathbf{c}$ agora está no lugar de $\mathbf{b}$), e que o que buscamos é uma solução cada vez mais próxima da viabilidade. Isto é o que faz o algoritmo dual-simplex.

Suponha que tenhamos uma solução básica para o primal que tenha valor melhor que qualquer ponto do politopo, mas que não seja viável. Representamos esta solução pela base A_B, tal que $\mathbf{x}_B = A_B^{-1}\mathbf{b}$.

Manteremos o tempo todo a condição de viabilidade dual: a solução sempre será ótima para o primal,

$$\mathbf{c_N - z \leq 0.}$$

Começamos com uma solução que é inviável para o primal (há algum $b_i < 0$). Se em alguma iteração tivermos conseguido viabilidade primal (todos os $b_i \geq 0$), então teremos uma solução ótima e viável para o primal.

5.4.1 Quem sai?

Queremos trazer o sistema para a viabilidade, e sabemos que há elementos $b_p < 0$. Podemos portanto fazer pivoteamento em uma coluna a_{pq}, com $a_{pq} < 0$, e q fora da base. Isto resultará em um novo valor de b_p maior que zero:

$$\cdots \overbrace{-a_{pq}} x_q + \cdots = \overbrace{-b_p}$$
$$\cdots + 1x_q + \cdots = \frac{b_p}{a_{pq}}$$

5.4.2 Quem entra?

Dentre todos os x_q, escolhemos aquele que mantém a viabilidade. Usando raciocínio semelhante ao que usamos para determinar a coluna a sair da base, temos

$$q = \arg\min_{j \leq m} \left\{ \frac{c_j - z_j}{a_{rj}} : a_{pj} < 0 \right\}$$

Como todos os $c_j - z_j$ são menores ou iguais a zero, estaremos minimizando em um conjunto limitado por baixo.

O exercício 52 pede o desenvolvimento dos detalhes.

5.4.3 Base inicial

Já descrevemos completamente os passos do algoritmo Simplex-dual. Ele exige, no entanto, que iniciemos com uma solução *ótima mas inviavel* para o primal. Descrevemos aqui o método da restrição artificial para obter uma base inicial para o Simplex-dual.

i) Adicione a restrição

$$\sum_{j \geq m} x_j \leq M,$$

onde M é muito grande (as variáveis somadas são as não-básicas, $j \geq m$). Esta restrição incluirá na base uma nova variável de folga, x_{n+1}.

$$\left(\begin{array}{cc|c} A_B & A_N & \mathbf{b} \end{array}\right) \longrightarrow \left(\begin{array}{cc|c|c} \mathbf{0}^T & \mathbf{1}^T & 1 & M \\ \hline A_B & A_N & \mathbf{0} & \mathbf{b} \end{array}\right)$$

ii) Na primeira iteração, removemos a variável x_{n+1}, incluindo a coluna com maior coeficiente reduzido de custo.

iii) Seguimos usando o algoritmo Simplex-dual.

O exercício 53 pede a demonstração da Proposição 5.20.

Proposição 5.20. *Partindo de um tableau Simplex com base* viável para o primal *de um problema, o método da restrição artificial resulta em um tableau* viável para o dual *do mesmo problema (ou seja, ótimo para o primal), mas não necessariamente viável para o primal.*

Exemplo 5.21. Usaremos como exemplo o seguinte problema.

$$\begin{aligned} \max\ & 2x_1 + x_2 \\ \text{s.a.:}\ & x_1 - x_2 \leq 10 \\ & 3x_1 - 2x_2 \leq 15 \\ & -x_1 + 5x_2 \leq 20 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

O tableau inicial para o primal seria

$$\left(\begin{array}{rrrrr|r} 1 & -1 & 1 & 0 & 0 & 10 \\ 3 & -2 & 0 & 1 & 0 & 15 \\ -1 & 5 & 0 & 0 & 1 & 20 \\ \hline 2 & 1 & 0 & 0 & 0 & 0 \end{array}\right)$$

com base x_3, x_4, x_5. Adicionamos portanto a restrição

$$x_1 + x_2 \leq M$$

e o novo tableau é

$$\left(\begin{array}{rrrrrr|r} 1 & 1 & 0 & 0 & 0 & 1 & M \\ 1 & -1 & 1 & 0 & 0 & 0 & 10 \\ 3 & -2 & 0 & 1 & 0 & 0 & 15 \\ -1 & 5 & 0 & 0 & 1 & 0 & 20 \\ \hline 2 & 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

Agora removemos x_6 da base, incluindo x_1 (que tem o maior coeficiente reduzido de custo).

$$\left(\begin{array}{cccccc|c} 1 & 1 & 0 & 0 & 0 & 1 & M \\ 0 & -2 & 1 & 0 & 0 & -1 & 10-M \\ 0 & -5 & 0 & 1 & 0 & -3 & 15-3M \\ 0 & 6 & 0 & 0 & 1 & 1 & 20+M \\ \hline & -1 & 0 & 0 & 0 & -2 & -M \end{array}\right)$$

Agora o tableau é ótimo para o primal: os coeficientes reduzidos de custo são -1 e -2. Como ainda é inviável para o primal, podemos usar o algoritmo Simplex-dual a partir daqui. ◀

Após resolver o problema usando o algoritmo Simplex dual, o resultado final poderá ser:

i) x_{n+1} na base: a solução é ótima

ii) x_{n+1} fora da base, com coeficiente reduzido de custo zero: a solução é ótima, e o M escolhido é justo

iii) x_{n+1} fora da base, com coeficiente reduzido de custo negativo: o M está limitando a solução, e o coeficiente reduzido de custo negativo significa que se incluirmos esta folga na base, o objetivo decairá. O problema primal é ilimitado.

5.4.4 Resumo e exemplos

O algoritmo dual-Simplex é mostrado a seguir.

seja A_B base primal ótima ($c_j - z_j < 0 \quad \forall j \notin B$)
calcule $\mathbf{x}_B = A_B^{-1}\mathbf{b}$
enquanto $\mathbf{x}_B$ tem elementos negativos:
    $p \leftarrow \arg\min_i\{x_i : x_i < 0\}$
    se $a_{pj} \geq 0$ todo j:
        PARE -- primal ilimitado
    escolha j tal que $a_{pj} < 0$
    $q \leftarrow \arg\max_{j \leq m} \left\{ \frac{c_j - z_j}{a_{rj}} : a_{pj} < 0 \right\}$
    retire a_p da base e inclua a_q
PARE -- $\mathbf{x}_B$ é viável e ótima

Exemplo 5.22. Resolveremos o seguinte problema usando o algoritmo dual-Simplex.

$$\begin{aligned} \max\ & x_1 + 4x_2 \\ \text{s.a.:}\ & 2x_1 + x_2 \leq 40 \\ & x_1 + 2x_2 \leq 50 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

O tableau inicial é

$$\left(\begin{array}{cccc|c} 2 & 1 & 1 & 0 & 40 \\ 1 & 2 & 0 & 1 & 50 \end{array}\right)$$

Observamos que o tableau é viável para o primal, e não serve para o método simplex dual. Adicionamos a restrição

$$x_1 + x_2 \leq M$$

e o novo tableau é

$$\left(\begin{array}{ccccc|c} 1 & 1 & 0 & 0 & 1 & M \\ 2 & 1 & 1 & 0 & 0 & 40 \\ 1 & 2 & 0 & 1 & 0 & 500 \\ \hline 1 & 4 & 0 & 0 & 0 & 0 \end{array}\right) \quad \begin{array}{l} \mathbf{z} = \mathbf{c}_B^T A_N = (0\,0\,0) A_N = (0\,0) \\ \mathbf{c_N} - \mathbf{z} = (1\,4) \end{array}$$

a nova restrição (x_5) sairá.

$$\left(\begin{array}{ccccc|c} 1 & 1 & 0 & 0 & 1 & M \\ 1 & 0 & 1 & 0 & -1 & 40 - M \\ -1 & 0 & 0 & 1 & -2 & 50 - 2M \\ \hline -3 & 0 & 0 & 0 & -4 & -4M \end{array}\right)$$

Percebemos agora que o tableau já é viável para o dual. Escolhemos $M = 50$.

$$\left(\begin{array}{ccccc|c} 1 & 1 & 0 & 0 & 1 & 50 \\ 1 & 0 & 1 & 0 & -1 & -10 \\ -1 & 0 & 0 & 1 & -2 & -50 \\ \hline -3 & 0 & 0 & 0 & -4 & -200 \end{array}\right)$$

Removeremos a terceira variável da base, x_4, e incluiremos x_5.

$$\left(\begin{array}{ccccc|c} 1/2 & 1 & 0 & 1/2 & 0 & 25 \\ 3/2 & 0 & 1 & -1/2 & 0 & 15 \\ 1/2 & 0 & 0 & -1/2 & 1 & 25 \\ \hline -1 & 0 & 0 & -2 & 0 & -100 \end{array}\right)$$

Como todos os b_i são positivos, temos a solução ótima, com $x = 0$ e $x_2 = 25$. ◀

Exemplo 5.23. Resolveremos agora o problema a seguir usando o algoritmo dual-Simplex.

$$\begin{aligned} \max\ & x_1 + 2x_2 \\ \text{s.a.:}\ & x_1 + 2x_2 \leq 6 \\ & x_1 - x_2 \leq 2 \\ & x_2 \leq 2 \\ & x_1 \leq 3 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Um tableau inicial para o primal é mostrado, apenas para referência, a seguir (ele não será usado).

$$\left(\begin{array}{cccccc|c} 1 & 2 & 1 & 0 & 0 & 0 & 6 \\ 1 & -1 & 0 & 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 & 1 & 0 & 2 \\ 1 & 0 & 0 & 0 & 0 & 1 & 3 \end{array}\right)$$

Começamos com a base (x_1, x_4, x_5, x_6), mostrada no tableau a seguir.

$$\left(\begin{array}{cccccc|c} 1 & 2 & 1 & 0 & 0 & 0 & 6 \\ 0 & -3 & -1 & 1 & 0 & 0 & -4 \\ 0 & 1 & 0 & 0 & 1 & 0 & 2 \\ 0 & -2 & -1 & 0 & 0 & 1 & -3 \\ \hline 0 & 0 & -1 & 0 & 0 & 0 & \end{array}\right)$$

Esta solução é evidentemente inviável ($x_4 = -4, x_6 = -3$), mas ótima (os coeficientes reduzidos de custo são negativos).

Escolhemos primeiro a coluna a *sair*: o menor dos elementos negativos no lado direito é -4, na segunda linha, portanto a *segunda* coluna da base (x_4) sairá.

$$\begin{aligned} 0 \div -3 &= 0 \\ -1 \div -1 &= 1 \end{aligned}$$

A coluna de x_2 entrará na base.

$$\left(\begin{array}{cccccc|c} 1 & 0 & 1/3 & 2/3 & 0 & 0 & 10/3 \\ 0 & 1 & 1/3 & -1/3 & 0 & 0 & 4/3 \\ 0 & 0 & -1/3 & 1/3 & 1 & 0 & 2/3 \\ 0 & 0 & -1/3 & -2/3 & 0 & 1 & -1/3 \\ \hline 0 & 0 & -1 & 0 & 0 & 0 & \end{array}\right)$$

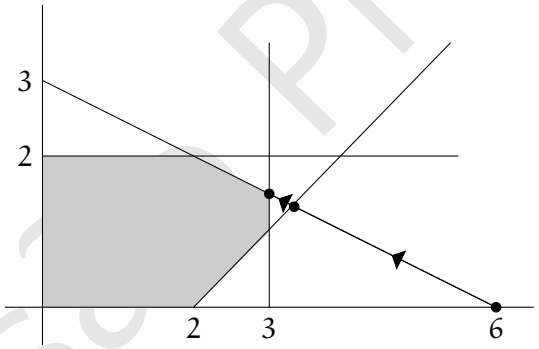
Ainda precisamos remover a variável x_6 da base, porque ela tem valor $-1/3$.

$$-1 \div -\frac{1}{3} = 3$$
$$0 \div -\frac{2}{3} = 0$$

A coluna de x_4 entra na base.

$$\left(\begin{array}{cccccc|c} 1 & 0 & 0 & 0 & 0 & 1 & 3 \\ 0 & 1 & 1/2 & 0 & 0 & -1/2 & 3/2 \\ 0 & 0 & -1/2 & 0 & 1 & 1/2 & 1/2 \\ 0 & 0 & 1/2 & 1 & 0 & -3/2 & 1/2 \\ \hline 0 & 0 & -1 & 0 & 0 & 0 & \end{array}\right)$$

A figura a seguir mostra o caminho feito pelo algoritmo dual-Simplex. Partimos de uma solução básica não viável (interseção de restrições fora da região viável) e mudamos a base, sempre diminuindo a distâmcia até a viabilidade.



◀

Exemplo 5.24. Este exemplo mostra a aplicação do algoritmo Simplex-dual em um problema ilimitado.

$$\begin{aligned} \max\ & x_1 + x_2 \\ \text{s.a.:}\ & x_1 \leq 1 \\ & x_1 - x_2 \leq 2 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Usaremos a restrição artificial

$$x_1 + x_2 \leq M,$$

e nosso tableau inicial é mostrado a seguir.

$$\left(\begin{array}{ccccc|c} 1 & 1 & 0 & 0 & 1 & M \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & -1 & 0 & 1 & 0 & 2 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 \end{array}\right)$$

Retiramos x_{n+1} da base, incluíndo x_1.

$$\left(\begin{array}{ccccc|c} 1 & 1 & 0 & 0 & 1 & M \\ 0 & -1 & 1 & 0 & -1 & 1-M \\ 0 & -2 & 0 & 1 & -1 & 2-M \\ \hline 0 & 0 & 0 & 0 & -1 & -M \end{array}\right)$$

Note que agora a solução é inviável mas ótima para o primal. Retiramos a segunda variável da base. Quem entra é x_2.

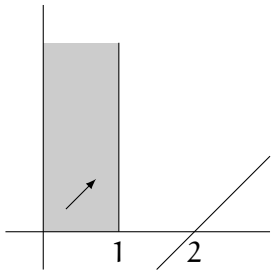
$$\left(\begin{array}{ccccc|c} 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & -1 & 0 & 1 & M-1 \\ 0 & 0 & -2 & 1 & 1 & M \\ \hline 0 & 0 & 0 & 0 & -1 & -M \end{array}\right)$$

Observamos que temos um tableau teoricamente ótimo, porque todos os b_i são positivos. No entanto, a variável artificial x_{n+1} (que foi incluída para obtermos uma solução primal-inviável) está fora da base. Isto significa que a restrição

$$x_1 + x_2 \leq M$$

está sem folga – temos $x_1 + x_2 = M$, para M tão grande quanto queiramos – e a solução pode ter valor menor se incluírmos esta folga na base (porque o coeficiente de custo de x_{n+1} é -1. Concluimos que o problema é ilimitado.

A figura a seguir ilustra graficamente o problema – a seta indica o gradiente da função objetivo.



A região é ilimitada na direção de x_2 (o que também se pode facilmente verificar inspecionando a descrição do problema). ◄

Notas

A demonstração do teorema forte da dualidade (Teorema 5.13) dada neste texto é semelhante àquela apresentada por Alexander Schrijver [Sch98].

Exercícios

Ex. 43 — Mostre o dual dos problemas a seguir.

$$\begin{aligned} (\mathrm{i}) \quad & \max x_1 + x_2 + 3x_3 \\ & \text{s.a.} : x_1 - 2x_2 = 5 \\ & \quad x_1 + x_3 \geq 10 \\ & \quad 3x_2 + 4x_3 \leq 9 \end{aligned} \qquad \begin{aligned} (\mathrm{ii}) \quad & \min 3x_1 + 2x_2 \\ & \text{s.a.} : x_1 + x_2 \geq 1 \\ & \quad x_1 - x_2 \leq 2 \end{aligned}$$

$$\begin{aligned} (\mathrm{iii}) \quad & \max 2x_1 + 2x_2 - 3x_3 \\ & \text{s.a.} : 2x_1 + 3x_2 \leq 3 \\ & \quad 5x_2 + 5x_3 \leq 5 \\ & \quad 9x_1 + 2x_3 \leq 4 \end{aligned} \qquad \begin{aligned} (\mathrm{iv}) \quad & \min 2x_1 + 5x_2 \\ & \text{s.a.} : 2x_1 + 7x_2 \geq 5 \\ & \quad 2x_2 - x_3 = 1 \end{aligned}$$

Ex. 44 — Resolva os problemas do exercício 43 usando o método Simplex-dual.

Ex. 45 — Mostre que o problema

$$\begin{aligned} & \max \mathbf{c}^T \mathbf{x} \\ & \text{s.a:} A\mathbf{x} = \mathbf{b} \\ & \quad \mathbf{x} \geq \mathbf{0} \end{aligned}$$

tem solução se e somente se $\mathbf{c}^T$ é combinação linear das linhas de A.

Ex. 46 — Implemente o algoritmo Simplex dual. Inicialmente represente o tableau inteiro; depois, tente usar as idéias do Simplex revisado, mas com o algoritmo dual.

Ex. 47 — Antes do Lema 5.12, mencionamos que uma condição necessária para que um programa linear tenha solução ótima é que o gradiente

do objetivo possa ser expresso como combinação linear não negativa das restrições. Demonstre rigorosamente isto.

Ex. 48 — Prove o Lema 5.12.

Ex. 49 — Prove o Teorema 5.3.

Ex. 50 — Se a solução ótima do dual é degenerada, o que se pode dizer sobre a solução ótima do primal?

Ex. 51 — Suponha que eu tenha resolvido o dual de um problema, e encontrei uma solução degenerada, com k variáveis básicas assumindo valor zero. O que posso dizer sobre o conjunto de soluções ótimas do primal?

Ex. 52 — Desenvolva os detalhes da seção 5.4.2 – conforme mencionado ali, trata-se de raciocínio semelhante ao usado para escolha da coluna a sair no algoritmo Simplex primal.

Ex. 53 — Prove a Proposição 5.20.

Capítulo 6

Análise de Sensibilidade

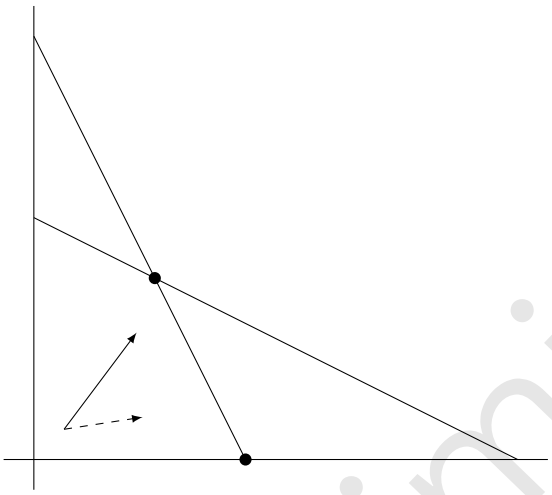
Suponha que tenhamos encontrado a solução $\mathbf{x}^*$, ótima para o programa linear

$$\max \mathbf{c}^T\mathbf{x} \text{ s.a.: } A\mathbf{x} \leq \mathbf{b}.$$

Verificaremos o quanto podemos mudar no problema sem mudar sua solução ótima, que já encontramos. A isso damos o nome de *análise de sensibilidade*.

6.1 Mudanças no objetivo

Mudar coeficientes no vetor que define a função objetivo terá um único efeito importante: o gradiente mudará de direção. Se o ângulo for suficientemente grande, a solução ótima pode mudar. Na próxima figura, as duas setas representam dois possíveis gradientes da função objetivo, e duas soluções ótimas diferentes – uma para cada um dos objetivos diferentes.



Teorema 6.1. *Suponha que um valor Δ tenha sido somado ao coeficiente c_k. Se x_k estiver na base, a solução ótima será a mesma, $\mathbf{x}^*$, do problema original se, para toda variável x_j* fora *da base,*

$$\Delta \geq \frac{(c_j - z_j)}{a_{kj}} \quad \textit{se } a_{kj} > 0$$
$$\Delta \leq \frac{(c_j - z_j)}{a_{kj}} \quad \textit{se } a_{kj} < 0$$

Se x_k não estiver na base, a solução continuará sendo ótima se

$$\Delta \leq -(c_k - z_k).$$

Demonstração. Como a descrição da região viável é dada somente por A e b, a solução $\mathbf{x}^*$ continua viável. Resta determinarmos se continua ótima ou não.

A condição de otimalidade para uma solução é

$$c_j - z_j \leq 0, \quad \forall j.$$

Analisaremos dois casos: o primeiro, em que x_k^* era básica na solução ótima, e o segundo caso, em que x_k não era básica.

Primeiro caso (x_k^* básica): o novo valor z_j' será igual ao anterior (porque somente os c_j para j fora da base compõe z_j), mas c_k' passa a ser $c_k + \Delta$.

Primeiro observamos que

$$\begin{aligned} c_j - z_j &= c_j - c_B^T A_N \\ &= c_j - \sum_{i \leq m} c_i a_{ij}. \end{aligned}$$

Assim, queremos

$$
\begin{aligned}
c_j - z'_j &\leq 0 \\
c_j - \sum_{i \leq m} c'_i a_{ij} &\leq 0 \\
c_j - c'_k a_{kj} - \left(\sum_{i \leq m, i \neq k} c'_i a_{ij} \right) &\leq 0 && \text{(tirando k do somatório)} \\
c_j - c'_k a_{kj} - \left(\sum_{i \leq m, i \neq k} c_i a_{ij} \right) &\leq 0 && \text{(para } i \neq k, c_i = c'_i\text{)} \\
c_j - (c_k + \Delta) a_{kj} - \left(\sum_{i \leq m, i \neq k} c_i a_{ij} \right) &\leq 0 \\
c_j - c_k a_{kj} - \Delta a_{kj} - \left(\sum_{i \leq m, i \neq k} c_i a_{ij} \right) &\leq 0 \\
c_j - \Delta a_{kj} - \left(\sum_{i \leq m} c_i a_{ij} \right) &\leq 0 && \text{(devolvendo } c_k a_k \text{ ao somatório)} \\
c_j - \Delta a_{kj} - z_j &\leq 0 \\
(c_j - z_j) - \Delta a_{kj} &\leq 0 \\
\Delta a_{kj} &\geq (c_j - z_j)
\end{aligned}
$$

Multiplicamos a inequação por a_{kj}, e o resultado depende do sinal de a_{kj}:

$$
\begin{aligned}
\Delta &\geq \frac{(c_j - z_j)}{a_{kj}} && \text{se } a_{kj} > 0 \\
\Delta &\leq \frac{(c_j - z_j)}{a_{kj}} && \text{se } a_{kj} < 0
\end{aligned}
$$

Assim, se Δ respeitar essa restrição, $\mathbf{x}^*$ continua sendo ótima.

Segundo caso (x^*_k fora da base): o coeficiente c'_k modificado não altera o vetor $\mathbf{z}$, portanto somente o coeficiente reduzido de custo da própria variável x_k é modificado. Assim, queremos

$$
\begin{aligned}
c'_k - z_k &\leq 0 \\
(c_k + \Delta) - z_k &\leq 0 \\
\Delta &\leq -(c_k - z_k).
\end{aligned}
$$

■

Exemplo 6.2. Considere o problema a seguir:

$$\begin{aligned} \max\ & 3x + 4y \\ \text{s.a.:}\ & 2x + y <= 7 \\ & x + 2y <= 8 \\ & x - y <= 6 \\ & x, y \geq 0. \end{aligned}$$

O tableau final é

$$\begin{pmatrix} 1 & 0 & 2/3 & -1/3 & 0 & 2 \\ 0 & 1 & -1/3 & 2/3 & 0 & 3 \\ 0 & 0 & -1 & 1 & 1 & 7 \\ 0 & 0 & -2/3 & -5/3 & 0 & -18 \end{pmatrix},$$

com x, y e a variável s_3 na base. Suponha que queiramos mudar o coeficiente de x, de modo que a função objetivo passe a ser

$$z_0 = (3 + \Delta)x + 4y.$$

Aplicando o Teorema, obtemos os seguintes limites para Δ.

- Para $j = 3$: como $a_{13} = 2/3 > 0$, temos

$$\Delta \geq \frac{-2/3}{2/3} = -1.$$

- Para $j = 4$: como $a_{14} = -1/3 < 0$, temos

$$\Delta \leq \frac{-5/3}{-1/3} = +5.$$

Assim, com $\Delta \in [-1, +5]$ garantimos a otimalidade da solução que já tínhamos. De fato, para qualquer função objetivo entre $2x + 4y$ e $8x + 4y$, ou seja, da forma

$$[2, 8]x + 4y,$$

a solução $x = 2$ e $y = 3$ continua ótima, mas fora desse intervalo não. ◀

Exemplo 6.3. Agora trabalharemos no problema a seguir:

$$\begin{aligned} \max\ & -2x + 3y \\ \text{s.a.:}\ & 2x + y <= 4 \\ & x - y <= 5 \\ & -x + y >= 1 \\ & x, y \geq 0. \end{aligned}$$

O tableau final é

$$\begin{pmatrix} 3 & 0 & 1 & 0 & 1 & 3 \\ 3 & 0 & 1 & 1 & 0 & 9 \\ 2 & 1 & 1 & 0 & 0 & 4 \\ -8 & 0 & -3 & 0 & 0 & -12 \end{pmatrix},$$

representando a solução $x = 0$, $y = 4$. A base tem (s_3, s_2, y).

Queremos mudar o coeficiente de x:

$$\max(-2 + \Delta)x + 3y$$

Como x não está na base,

$$\Delta \leq -(-8)$$
$$\Delta \leq 8$$

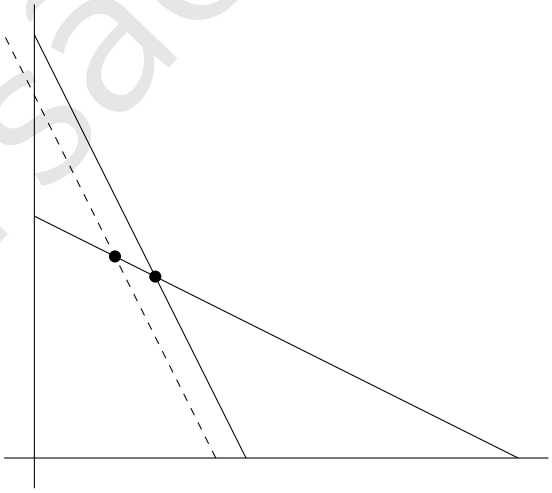
Para este problema, qualquer função objetivo da forma

$$(-\infty, 6]x + 3y$$

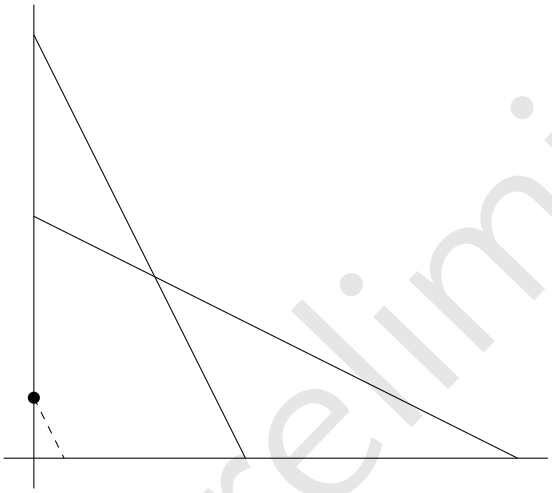
nos levará à mesma solução ótima, $x = 0$, $y = 4$. Quando o coeficiente de x é maior que 6, a solução muda. ◀

6.2 Mudanças no vetor b

Uma mudança no vetor **b** é um deslocamento paralelo de uma das restrições, como mostra a próxima figura.



A figura mostra que nas duas soluções a base é a mesma (tanto x como y estão presentes), mas o valor de x e de y muda. Se a distancia para a nova restrição for muito grande, pode ser que a base mude, como ilustra a figura a seguir.



A mudança em b não afeta o critério de otimalidade ($c_j - z_j \leq 0$) para nossa base – se ela continuar representando uma solução viável, a *base* continuará sendo ótima, mas os valores das variáveis podem mudar.

Teorema 6.4. *Suponha que tenhamos resolvido um problema e obtido uma solução ótima $\mathbf{x}^*$, com base A_B. Se um valor Δ é somado ao coeficiente b_k, resultando em $\mathbf{b}'$. A base A_B continuará sendo ótima, mas só será viável se*

$$\Delta \geq \max_q \left\{ -\frac{x_q^*}{\left(A_B^{-1}\right)_{ik}} \ : \quad \left(A_B^{-1}\right)_{ik} > 0 \right\}$$
$$\Delta \leq \min_q \left\{ -\frac{x_q^*}{\left(A_B^{-1}\right)_{ik}} \ : \quad \left(A_B^{-1}\right)_{ik} < 0 \right\}$$

Demonstração. Observamos que não temos somente uma solução $\mathbf{x}^*$. Temos uma *base* A_B, e calculamos nossa solução

$$\mathbf{x}^* = A_B^{-1}\mathbf{b}.$$

Seja

$$\begin{aligned}\mathbf{x}^{**} &= A_B^{-1}\mathbf{b}' \\ &= A_B^{-1}\mathbf{b} + A_B^{-1}\,(0,0,\ldots,\Delta,\ldots,0,\ldots)^T \\ &= \mathbf{x}^* + \underbrace{A_B^{-1}\,(0,0,\ldots,\Delta,0,\ldots)^T}_{\substack{\text{k-ésima coluna de } A_B,\\ \text{multiplicada por } \Delta}},\end{aligned}$$

ou seja, a k-ésima coluna de A_B^{-1} é multiplicada por Δ e somada a $\mathbf{x}^*$.

$$x_i^{**} = x_i^* + \left(A_B^{-1}\right)_{ik}\Delta.$$

Então queremos que, *para todo* i,

$$\begin{aligned} x_i^{**} &\geq 0 \\ x_i^* + \left(A_B^{-1}\right)_{ik}\Delta &\geq 0,\end{aligned}$$

o que nos leva diretamente ao enunciado do Teorema. ■

Exemplo 6.5. Agora trabalharemos no problema a seguir:

$$\begin{aligned} \max\ & -2x+3y \\ \text{s.a.:}\ & 2x+y \leq 4 \\ & x-y \leq 5 \\ & -x+y \leq 1 \\ & x,y \geq 0.\end{aligned}$$

O tableau final é

$$\left(\begin{array}{ccccc|c} 1 & 0 & 1/3 & 0 & -1/3 & 1 \\ 0 & 0 & 0 & 1 & 1 & 6 \\ 0 & 1 & 1/3 & 0 & 2/3 & 2 \\ \hline 0 & 0 & -1/3 & 0 & 8/3 & -4 \end{array}\right),$$

A solução é $x=1, y=2$. A submatriz com a inversa da base é

$$A_B^{-1} = \begin{pmatrix} 1/3 & 0 & -1/3 \\ 0 & 1 & 1 \\ 1/3 & 0 & 2/3 \end{pmatrix}.$$

Note que os valores de x e y estão nas posições 1 e 3 do vetor ao lado direito, e podemos obter a solução calculando

$$A_B^{-1}\mathbf{b} = \begin{pmatrix} 1/3 & 0 & -1/3 \\ 0 & 1 & 1 \\ 1/3 & 0 & 2/3 \end{pmatrix} \begin{pmatrix} 4 \\ 5 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 6 \\ 2 \end{pmatrix}$$

Consideramos mudar b_3 para $b_3 + \Delta$.

Queremos que para todo i,

$$x_i^* + (A_B^{-1})_{i2}\Delta \geq 0$$

ou seja,

$$\Delta \geq \frac{x_i^*}{(A_B^{-1})_{i3}}$$

$$\begin{aligned}
\Delta &\geq \max_q \left\{ -\frac{x_q^*}{(A_B^{-1})_{qk}} \ : \quad \left(A_B^{-1}\right)_{qk} > 0 \right\} \\
&= \max\left\{ -\frac{6}{1}, -\frac{2}{2/3} \right\} = -3. \\
\Delta &\leq \min_q \left\{ -\frac{x_q^*}{(A_B^{-1})_{qk}} \ : \quad \left(A_B^{-1}\right)_{qk} < 0 \right\} \\
&= \min\left\{ -\frac{1}{-1/3} \right\} = 3.
\end{aligned}$$

Assim, o valor de b_3 deve ficar entre $1-3$ e $1+3$, ou seja, $b_3 \in [-2, 4]$, para que a base atual continue ótima. No entanto, o *valor* da solução poderá mudar.

Por exemplo, se tomarmos $b_3 = 2$, a mesma base é ótima, mas os valores das variáveis mudam:

$$A_B^{-1}\mathbf{b} = \begin{pmatrix} 1/3 & 0 & -1/3 \\ 0 & 1 & 1 \\ 1/3 & 0 & 2/3 \end{pmatrix} \begin{pmatrix} 4 \\ 5 \\ 2 \end{pmatrix} = \begin{pmatrix} 2/3 \\ 7 \\ 8/3 \end{pmatrix}$$

O ótimo acontece com $x = 1/3$ e $y = 8/3$. ◀

6.3 Nova variável

Se uma nova variável x_{n+1} é adicionada ao problema, sem mudanças nos coeficientes já existentes em A, $\mathbf{b}$ e $\mathbf{c}$, teremos uma nova coluna $\mathbf{a_{n+1}}$ em A e um novo elemento c_{n+1} em $\mathbf{c}$.

Podemos tomar o tableau que usamos para obter $\mathbf{x}^*$ e adicionar a nova coluna com $\mathbf{a_{n+1}}$ e c_{n+1}. Teremos também que calcular $c_{n+1} - z_{n+1}$. Isso já nos dará a informação que queremos: a solução $\mathbf{x}^*$ continuará sendo ótima somente se $c_{n+1} - z_{n+1} \leq 0$. Caso não seja, podemos imediatamente incluir $\mathbf{a_{n+1}}$ na base e usar o algoritmo Simplex para obter uma nova solução ótima.

No entanto, há um problema: quando incluimos a nova variável, temos seus coeficientes *na descrição inicial do problema*, e *não no tableau final*. Temos que calcular a coluna $\mathbf{a_{n+1}}$ primeiro, e só depois determinar o coeficiente reduzido de custo da nova variável.

Exemplo 6.6. Exemplificamos com o seguinte problema:

$$\begin{aligned} \max\ & x_1 + 2x_2 \\ \text{s.a.:}\ & x_1 + x_2 \leq 10 \\ & -3x_1 + 5x_2 \leq 15 \\ & x_2 \leq 5 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Os tableaux inicial e final para este problema são mostrados a seguir.

$$\left(\begin{array}{rrrrr:r} 1 & 1 & 1 & 0 & 0 & 10 \\ -3 & 5 & 0 & 1 & 0 & 15 \\ 0 & 1 & 0 & 0 & 1 & 5 \\ \hdashline 1 & 2 & 0 & 0 & 0 & 0 \end{array}\right) \rightsquigarrow \left(\begin{array}{rrrrr:r} 1 & 0 & 1 & 0 & -1 & 5 \\ 0 & 0 & 3 & 1 & -8 & 5 \\ 0 & 1 & 0 & 0 & 1 & 5 \\ \hdashline 0 & 0 & -1 & 0 & -1 & -15 \end{array}\right) \tag{6.1}$$

A inversa da base está nas colunas $3, 4, 5$; no tableau final, temos portanto

$$A_B^{-1} = \begin{pmatrix} 1 & 0 & -1 \\ 3 & 1 & -8 \\ 0 & 0 & 1 \end{pmatrix}.$$

Agora incluímos uma nova variável, x_3, no problema:

$$\begin{aligned} \max\ & x_1 + 2x_2 + \mathbf{3x_3} \\ \text{s.a.:}\ & x_1 + x_2 + x_3 \leq 10 \\ & -3x_1 + 5x_2 + 2x_3 \leq 15 \\ & x_2 + x_3 \leq 5 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

A coluna de x_3 no tableau inicial seria

$$a_3 = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}$$

Queremos incluir a coluna de x_3 no tableau final (o da direita em 6.1, com a solução ótima para o problema original, que ainda não tinha x_3). Calculamos

$$a_3' = A_B^{-1} a_3 = \begin{pmatrix} 1 & 0 & -1 \\ 3 & 1 & -8 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ -3 \\ 1 \end{pmatrix}$$

Incluímos a nova coluna no tableau:

$$\left(\begin{array}{cccccc|c} 1 & 0 & 1 & 0 & -1 & \mathbf{0} & 5 \\ 0 & 0 & 3 & 1 & -8 & \mathbf{-3} & 5 \\ 0 & 1 & 0 & 0 & 1 & \mathbf{1} & 5 \\ \hline 0 & 0 & -1 & 0 & -1 & \mathbf{+1} & -15 \end{array}\right)$$

$$\begin{aligned} \mathbf{z} &= \mathbf{c}_B^T A_N = (1\ 0\ 2) \begin{pmatrix} 1 & -1 & 0 \\ 3 & -8 & -3 \\ 0 & 1 & 1 \end{pmatrix} \\ &= (1\ 1\ 2). \\ \mathbf{c}_N^T - \mathbf{z} &= (0\ 0\ 3) - (1\ 1\ 2) \\ &= (-1\ -1\ +1) \end{aligned}$$

Como o coeficiente reduzido de custo de x_3 é 1, devemos incluí-la na base.

$$\left(\begin{array}{cccccc|c} 1 & 0 & 1 & 0 & -1 & 0 & 5 \\ 0 & 3 & 3 & 1 & -5 & 0 & 20 \\ 0 & 1 & 0 & 0 & 1 & 1 & 5 \\ \hline 0 & -1 & -1 & 0 & -2 & 0 & -20 \end{array}\right)$$

$$\begin{aligned} \mathbf{z} &= \mathbf{c}_B^T A_N = (1\ 0\ 3) \begin{pmatrix} 0 & 1 & -1 \\ 3 & 3 & -5 \\ 1 & 0 & 1 \end{pmatrix} \\ &= (3\ 1\ 2) \\ \mathbf{c}_N^T - \mathbf{z} &= (2\ 0\ 0) - (3\ 1\ 2) \\ &= (-1\ -1\ -2). \end{aligned}$$

Temos agora uma solução ótima,

$$x_1 = 5,$$
$$x_2 = 0,$$
$$x_3 = 5,$$

com valor $x_1 + 2x_2 + 3x_3 = 1 \times 5 + 3 \times 5 = 20$. ◀

A inclusão de uma variável nova pode tornar o problema ilimitado ou inviável.

6.4 Nova restrição

Notas

Sinha [Sin06] e Vanderbei [Van07] discutem também *Programação Paramétrica*, que trata de problemas de programação linear onde as mudanças não são discretas como as de que tratamos aqui, mas contínuas: por exemplo, o vetor $\mathbf{c}$ pode variar continuamente com uma função

$$\mathbf{c} = \mathbf{c_0} + \tau\mathbf{d},$$

onde $\mathbf{c_0}$ e $\mathbf{d}$ são vetores.

Exercícios

Ex. 54 — Faça a análise de sensibilidade dos problemas apresentados no primeiro Capítulo: escolha elementos de $\mathbf{c}$, $\mathbf{b}$ e modifique, verificando para que intervalos as soluções continuam ótimas; inclua novas variáveis nos problemas, verificando se a solução ótima muda ou não.

Ex. 55 — Suponha que tenhamos resolvido um problema de maximização, e encontrado a solução ótima $\mathbf{x}^*$, com valor v. Em que situação é possível multiplicar uma linha i inteira da matriz A por -1 mantendo a otimalidade da solução?

Ex. 56 — Suponha que um programa linear tenha uma única solução ótima $\mathbf{x}^*$. Quanto podemos mudar um coeficiente da função objetivo de forma que $\mathbf{x}^*$ continue sendo a *única* solução ótima?

Ex. 57 — O que acontece com a solução ótima se removermos uma variável ou uma restrição de um problema de programação linear? Quando ela continua viável? E quando continua ótima?

Ex. 58 — Suponha que eu tenha resolvido um problema de programação linear e chegado a um tableau ótimo com uma solução em $\mathbb{R}^n$. Agora decidi remover uma das variáveis, simplesmente eliminando-a do problema, mantendo o resto das restrições – logo, agora estarei procurando por uma solução em $\mathbb{R}^{n-1}$. Como posso levar a solução anterior em $\mathbb{R}^{n-1}$, garantindo que será uma solução viável básica? Há como garantir que a nova solução será ótima? Explique geometricamente.

Capítulo 7

Outros Métodos

O algoritmo Simplex e suas variantes trabalham iterando soluções básicas (pontos extremos do poliedro definido pelas soluções viáveis do problema). É possível, embora muito raro na prática, que a quantidade de bases visitada pelo Simplex seja exponencial (lembramos que há $\binom{n}{m}$ delas).

7.1 O método do elipsoide

O russo[1] Khachiyan mostrou em 1979 como usar um algoritmo para resolver problemas de programação linear em tempo polinomial.

O algoritmo apresentado por Khachiyan na verdade resolve o problema das desigualdades estritas, que consiste em determinar, dados uma matriz A e um vetor $\mathbf{b}$, se existe $\mathbf{x}$ tal que $A\mathbf{x} < \mathbf{b}$. Nesta apresentação usaremos desigualdades não estritas, $A\mathbf{x} \leq \mathbf{b}$.

Começamos apresentando o algoritmo do elipsoide para sistemas de desigualdades estritas, e em seguida mostramos como resolver problemas de programação linear apenas encontrando soluções para sistemas de desigualdades.

7.1.1 O algoritmo

Um elipsoide é uma generalização de elipse para muitas dimensões. Em $\mathbb{R}^3$, um elipsóide é definido pela forma quadrática

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = k,$$

[1] Na época, também Soviético.

ou seja,

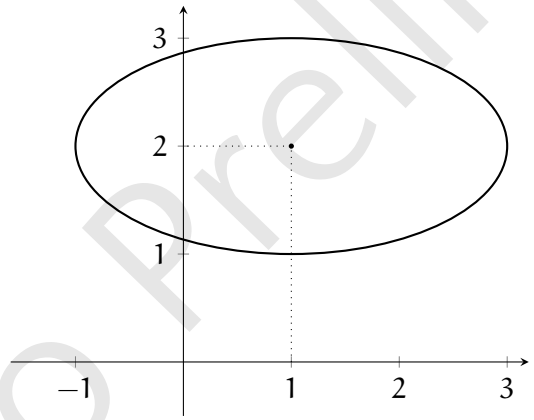
$$\begin{pmatrix} x & y & z \end{pmatrix} \begin{pmatrix} \frac{1}{a^2} & 0 & 0 \\ 0 & \frac{1}{b^2} & 0 \\ 0 & 0 & \frac{1}{c^2} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}.$$

Assim, podemos representar um elispóide por uma matriz quadrada simétrica, e a definição extende-se naturalmente para $\mathbb{R}^n$.

Definição 7.1 (Elipsoide)**.** Um *elipsoide* é o conjunto de pontos definido por

$$\left\{ \mathbf{x} : (\mathbf{x} - \mathbf{x}')M^{-1}(\mathbf{x} - \mathbf{x}') \leq 1 \right\}$$

onde x e x' são vetores com n elementos e M é uma matriz quadrada definida positiva e simétrica de ordem n. O vetor $\mathbf{x}'$ é o *centro* do elipsoide. ♦

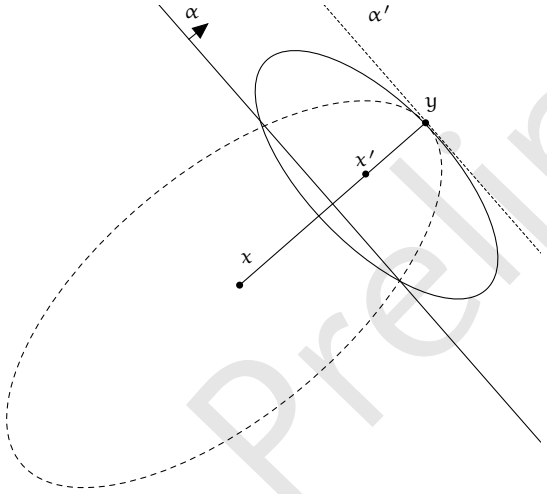


O algoritmo inicia com um elipsoide centrado na origem e tendo interseção com parte da região viável (se ela existir). Em cada iteração o algoritmo verifica se o centro do elipsoide é viável. Se for, obviamente a solução foi encontrada; senão, troca o elipsoide por outro, menor, que contém a interseção do anterior com a região viável. Pode-se demonstrar que se nenhuma solução for encontrada após um certo número de iterações, não há soluções viáveis para o sistema.

A cada iteração, uma desigualdade violada é usada para determinar o novo elipsoide (calcula-se novo centro x_{k+1} e nova matriz M_{k+1}), resultando em um novo elipsóide.

A cada iteração, se a solução atual (o centro do elipsóide) não é viável, há uma desigualdade violada $\mathbf{a}_i\mathbf{x} \leq b_i$. O algoritmo usa o hiperplano $\mathbf{a}_i\mathbf{x} = b_i$.

A próxima Figura ilustra o comportamento do algoritmo. A solução corrente era $\mathbf{x}$, e a restrição violada é representada por α ($\mathbf{x}$ está abaixo de α, mas a restrição determina que a região viável é acima dela). Se movermos α paralelamente a si mesma até tornar-se tangente no elipsoide, *na direção da viabilidade* (ou seja, para cima e à direita no exemplo dado), obtemos um ponto $\mathbf{y}$. A nova solução $\mathbf{x}'$ estará entre $\mathbf{x}$ e $\mathbf{y}$. O novo elipsoide, menor, será tangente a $\mathbf{y}$ e conterá toda a área viável que estava contida antes no primeiro elipsoide. Neste exemplo, $\mathbf{x}'$ é viável e o algoritmo pode parar.



O primeiro elipsoide deve ser grande o suficiente para conter pelo menos uma solução para o sistema, se alguma existir. Para determinar o tamanho deste elipsoide usamos o número de bits que o problema ocupa quando representado em um computador. Cada número pode ser representado por $\lfloor 1 + \log_2 |n| \rfloor$ bits mais um bit para o sinal. Para representar a matriz A, o vetor $\mathbf{b}$ e os números n e m usamos L bits, onde

$$\begin{aligned} L &= \left(\sum_{i \le m} \sum_{j \le n} 1 + \lfloor 1 + \log_2 |a_{ij}| \rfloor \right) + \left(\sum_{i \le m} 1 + \lfloor 1 + \log_2 |b_i| \rfloor \right) \\ &\quad + (\lfloor 1 + \log_2 n \rfloor) + (\lfloor 1 + \log_2 m \rfloor) \\ &\ge \sum_{i \le m} \sum_{j \le n} \lceil \log_2 |a_{ij}| \rceil + \sum_{i \le m} \lceil \log_2 |b_i| \rceil \\ &\quad + \lceil \log_2 n \rceil + \lceil \log_2 m \rceil + 2m(n+1) + 2 \end{aligned}$$

(Representamos n e m como inteiros sem sinal.)

Teorema 7.2. *Uma hiperesfera centrada na origem e com raio 2^L inclui pelo menos uma solução para $A\mathbf{x} \leq \mathbf{b}$, se alguma existir.*

Podemos portanto usar

$$M_0 = 2^L I$$

inicialmente.

Lema 7.3. *Se o volume do elipsoide na k-ésima iteração é V_k, então*

$$V_{k+1} < V_k e^{-1/(2n+2)} < 1.$$

Temos portanto $V_k \leq V_0 e^{-k/(2n+2)}$, e com isso podemos demostrar que o algoritmo roda em tempo polinomial.

Teorema 7.4. *O algoritmo do elipsoide tem complexidade de tempo polinomial.*

Demonstração. Como o volume da esfera inicial é $\pi C(n)(2^L)^n$, onde $C(n) \to 0$ quando $n \to \infty$, o algoritmo pode parar emm

$$\begin{aligned} k &= \left\lceil (2n+2) \log \left(\frac{\pi C(n) 2^{nL}}{\varepsilon} \right) \right\rceil \\ &\leq \left\lceil (2n+2) \log \left(\frac{\pi 2^{(n+1)L}}{\varepsilon} \right) \right\rceil \\ &= \left\lceil (2n+2) \left(\log \pi + \log(2^{n+1}) + \log \frac{2^L}{\varepsilon} \right) \right\rceil \end{aligned}$$

iterações. Cada iteração realiza $\mathcal{O}(n^2)$ operações, portanto a complexidade do algoritmo é $\mathcal{O}(n^4 + n^3 \log \frac{2^L}{\varepsilon})$.

É possível mostrar que o número de iterações que calculamos é menor que $6n(n+1)L$, e este é o critério de parada que usamos no algoritmo. ■

Se a inequação violada por $\mathbf{x}$ era a^i e o elipsoide era definido por M com centro em $\mathbf{x}$, então um novo elipsoide com centro em

$$\mathbf{x}' = x - \left(\frac{1}{n+1} \right) \frac{M\mathbf{a}^i}{\sqrt{\mathbf{a}_i^T M \mathbf{a}^i}}$$

e com

$$M' = \frac{n^2}{n^2-1} \left(M - \frac{2}{n+1} \frac{(M\mathbf{a}^i)(M\mathbf{a}^i)^T}{(\mathbf{a}^i)^T M \mathbf{a}^i} \right)$$

conterá a interseção do elipsoide anterior com o semiespaço definido pela inequação a^i. O pseudocódigo do algoritmo do elipsoide é dado a seguir.

$\mathbf{x} \leftarrow 0$
$M \leftarrow 2^L \mathcal{I}$
repita $6n(n+1)L$ `vezes:`
  se $A\mathbf{x} < \mathbf{b}$ `PARE --` **retorne** `x`
  senao
    `determine inequação violada por x (`$(\mathbf{a^i})^T\mathbf{x} > \mathbf{b}_i$`)`
    $\mathbf{x} \leftarrow \mathbf{x} - \left(\frac{1}{n+1}\right) \frac{M\mathbf{a}^i}{\sqrt{\mathbf{a}_i^T M \mathbf{a}^i}}$
    $M \leftarrow \frac{n^2}{n^2-1}\left(M - \frac{2}{n+1}\frac{(M\mathbf{a}^i)(M\mathbf{a}^i)^T}{(\mathbf{a}^i)^T M \mathbf{a}^i}\right)$
`PARE -- não há solução`

7.1.2 Resolvendo problemas de programação linear

Descrevemos o método do elipsoide como uma maneira de *encontrar um ponto que satisfaça um sistema de inequações* $A\mathbf{x} \leq \mathbf{b}$. Nesta seção mostramos que com isto podemos resolver quaisquer problemas de programação linear.

Considere o seguinte programa linear:

$$\begin{aligned} \max\ & \mathbf{c}^T\mathbf{x} \\ \text{s.a.:}\ & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Para resolvê-lo usando o método do elipsoide, usaremos seu dual:

$$\begin{aligned} \min\ & \mathbf{b}^T\mathbf{y} \\ \text{s.a.:}\ & A^T\mathbf{y} \geq \mathbf{c} \\ & \mathbf{y} \geq \mathbf{0} \end{aligned}$$

Sabemos que as soluções ótimas para estes problemas são tais que

$$\begin{aligned} A\mathbf{x} &\leq \mathbf{b} && \text{(restrições do primal)} \\ -A^T\mathbf{y} &\leq -\mathbf{c} && \text{(restrições do dual)} \\ -\mathbf{c}^T\mathbf{x} + \mathbf{b}^T\mathbf{y} &\leq 0 && \text{(dualidade: } \mathbf{c}^T\mathbf{x} = \mathbf{b}^T\mathbf{x}\text{)} \\ -\mathbf{x} &\leq 0 && \text{(não-negatividade)} \\ -\mathbf{y} &\leq 0 && \text{(não-negatividade)} \end{aligned}$$

Usamos o método do elipsoide para obter uma solução para este sistema, e teremos assim uma solução $\mathbf{x}$ ótima para o problema de programação linear.

Exemplo 7.5. Considere o problema a seguir:

$$\begin{aligned} \max\ & 3x_1 + 2x_2 \\ \text{s.a.:}\ & x_1 - 3x_2 \leq 4 \\ & 4x_1 + x_2 \leq 12 \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

O dual deste problema é

$$\begin{aligned} \min\ & 4y_1 + 12y_2 \\ \text{s.a.:}\ & y_1 + 4y_2 \geq 3 \\ & -3y_1 + y_2 \geq 2 \\ & \mathbf{y} \geq \mathbf{0}. \end{aligned}$$

O problema, para ser resolvido pelo método do elipsóide, é posto na seguinte forma:

$$\begin{aligned} & x_1 - 3x_2 \leq 4 \\ & 4x_1 + x_2 \leq 12 \\ \\ & -y_1 - 4y_2 \leq -3 \\ & 3y_1 - y_2 \leq -2 \\ \\ & -3x_1 - 2x_2 + 4y_1 + 12y_2 \leq 0 \\ \\ & -\mathbf{x} \leq 0 \\ & -\mathbf{y} \leq 0 \end{aligned}$$

◀

O algoritmo do elipsoide precisa de $6n(n+1)L$ iterações no *pior caso*. Infelizmente, o pior caso é o que quase sempre acontece na prática. O método Simplex, cujo pior caso roda em tempo exponencial, é quase sempre mais rápido que o elipsoide.

7.2 Pontos interiores

O método Simplex percorre diferentes pontos extremos do poliedro até encontrar a solução ótima para um programa linear.

Há métodos para resolução de problemas de programação linear que trabalham somente com *pontos interiores* do poliedro.

7.2.1 *Affine scaling*

Suponha que o problema esteja na forma

$$\max \mathbf{c}^T\mathbf{x} \text{ s.a.: } A\mathbf{x} = \mathbf{b}$$

O algoritmo começa com um ponto viável, muda a escala de forma que o ponto passe a ser um vetor unitário e o move na direção do gradiente do objetivo, garantindo que o valor da nova solução será melhor e que o novo ponto também será viável. Em cada iteração, temos um ponto viável $\mathbf{x}$. Mostraremos como obter o próximo ponto $\mathbf{x}'$. Primeiro mudamos a escala do problema para que o ponto passe a ser o vetor unitário: se $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, então definimos

$$D = \begin{pmatrix} x_1 & 0 & \ldots & 0 \\ 0 & x_2 & \ldots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \ldots & x_n \end{pmatrix}$$

O ponto modificado é $\mathbf{y} = D^{-1}\mathbf{x}$, de forma que

$$\mathbf{y} = D^{-1}\mathbf{x} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} = \mathbf{e}.$$

Dizemos que o algoritmo trabalhará tanto no *espaço-*$\mathbf{x}$ como no *espaço-*$\mathbf{y}$.

Temos portanto o problema $A(D\mathbf{y}) = \mathbf{b}$, ou $(AD)\mathbf{y} = \mathbf{b}$. Fazemos $S = AD$, e o novo programa linear é

$$\begin{aligned} \max\ & \mathbf{c}^T\mathbf{y} \\ \text{s.a: } & S\mathbf{y} = \mathbf{b} \\ & \mathbf{y} \geq 0 \end{aligned}$$

E neste novo problema temos $\mathbf{y} = \mathbf{e}$, que evidentemente é um ponto viável.

Moveremos $\mathbf{y}$ na direção do gradiente do objetivo. No entanto, não queremos sair da região viável. Uma maneira de permanecer na região viável é modificar a função objetivo, projetando-a no kernel de S.

Teorema 7.6. *Se $\mathbf{v}'$ é solução viável para um problema cujas restrições são $A\mathbf{v} = \mathbf{b}$, e $\mathbf{w}$ pertence ao espaço nulo de A, então $\mathbf{v}' + \mathbf{w}$ também é viável.*

Demonstração. Trivialmente, temos

$$\begin{aligned} A\mathbf{v}' &= \mathbf{b} \\ A\mathbf{w} &= \mathbf{0} \qquad\qquad (\mathbf{w} \text{ está no espaço nulo de } A) \end{aligned}$$

Ao somarmos $\mathbf{w}$ à solução viável $\mathbf{v}'$, obtemos outra solução viável:

$$A(\mathbf{v}' + \mathbf{w}) = A\mathbf{v}' + A\mathbf{w} = \mathbf{b} + \mathbf{0} = \mathbf{b}. \qquad \blacksquare$$

O operador de projeção no espaço nulo de S é[2]

$$\begin{aligned} P &= \mathcal{I} - S^T \left(SS^T\right)^{-1} S \\ &= \mathcal{I} - (AD)^T \left[AD(AD)^T\right]^{-1} AD \\ &= \mathcal{I} - DA^T \left[AD^2A^T\right]^{-1} AD \qquad\qquad (D = D^T \text{ (é diagonal)}) \end{aligned}$$

A projeção do gradiente $\mathbf{c}^T$ no espaço nulo de AD é

$$\begin{aligned} \mathbf{c_p} &= P(\mathbf{c}^T D)^T \\ &= PD\mathbf{c} \\ &= D\mathbf{c} - DA^T \left[AD^2A^T\right]^{-1} AD\,(D\mathbf{c}) \\ &= D\mathbf{c} - DA^T \left[AD^2A^T\right]^{-1} AD^2\mathbf{c}. \end{aligned}$$

Se c_p (que é o gradiente do objetivo, projetado) for igual a zero, não há direção viável que melhore a solução, que portanto é ótima.

Se $c_p \neq \mathbf{0}$, moveremos $\mathbf{y}$ em sua direção. Nos falta então determinar *quanto movê-lo no espaço-*$\mathbf{x}$.

[2] Uma demonstração de que este de fato é o projetor no espaço nulo pode ser encontrada no livro de Harry Dym [Dym07].

Suponha que queiramos, no espaço-$\mathbf{y}$, somar $\delta\mathbf{c_p}$ ao ponto, com $\delta > 0$ (não adicionaremos o vetor $\mathbf{c_p}$ inteiro). A nova solução será

$$\begin{aligned} \mathbf{y}' &= \mathbf{y} + \delta\mathbf{c}_p \\ D^{-1}\mathbf{x}' &= D^{-1}\mathbf{x} + \delta\mathbf{c}_p \\ \mathbf{x}' &= \mathbf{x} + \delta D\mathbf{c}_p. \end{aligned}$$

Assim,

$$\mathbf{d} = D\mathbf{c}_p$$

é o gradiente projetado, mas no espaço-$\mathbf{x}$.

Já garantimos que a nova solução respeitará as restrições do problema, porque projetamos o gradiente no espaço nulo de S. No entanto, *ainda falta garantir que a solução seja não-negativa:* precisamos que $\mathbf{x}' \geq \mathbf{0}$ (ou seja, *todos* os componentes de $\mathbf{x}'$ devem ser positivos). Como $\mathbf{x}' = D\mathbf{y}'$,

$$\begin{aligned} \mathbf{x}' &\geq \mathbf{0} \\ \mathbf{x} + \delta\mathbf{d} &\geq \mathbf{0} \\ \delta\mathbf{d} &\geq -\mathbf{x}. \end{aligned}$$

Para os componentes positivos de d_i esta condição é trivialmente satisfeita. Quando $d_i < 0$, precisamos de

$$\delta \leq -\frac{x_i}{d_i}.$$

Podemos então tomar

$$\delta = \min_{j \leq n} \left\{ -\frac{x_j}{d_j} : d_j < 0 \right\}.$$

Note que estamos minimizando em um conjunto de números *positivos* ($-x_j/d_j$, com $d_j < 0$, $x_j > 0$).

Teorema 7.7. *Se todos os d_i forem positivos, o problema é ilimitado.*

Demonstração. Se *todos* os d_i forem maiores que zero, então *qualquer* δ será viável. Poderíamos inclusive escolher δ maior que um. Como podemos andar à vontade na direção do gradiente sem que o ponto fique inviável, isto ($\mathbf{d} \geq \mathbf{0}$) significa que o problema é ilimitado. ■

Para termos $\mathbf{x}'$ ponto *interior* (e não na fronteira do poliedro), podemos escolher um valor um pouco aquém do que o δ calculado acima nos daria (por exemplo, $\alpha\delta$, com $\alpha = 0.95$).

Concluímos então com

$$\mathbf{x}' = \mathbf{x} + \alpha\delta\mathbf{d}, \quad \alpha \in (0,1).$$

Observe que, estritamente falando, o algoritmo nunca chega à solução ótima, mas a uma aproximação dela. Note no entanto que o gradiente do objetivo será cada vez menor, e o critério de parada pode ser portanto

$$\|\mathbf{c}_p\| < \varepsilon$$

para algum ε suficientemente pequeno.

Exemplo 7.8. Considere o problema a seguir

$$\begin{aligned} \max\ & x_1 + x_2 + x_3 \\ \text{s.a.:}\ & x_1 + 2x_2 + x_3 = 2 \\ & 2x_1 + 4x_2 - x_3 = 1 \\ & \mathbf{x} \geq 0. \end{aligned}$$

Começamos com a solução viável (mas não básica)

$$\mathbf{x} = (1/2, 1/4, 1)^T.$$

O valor do objetivo para esta solução é

$$\mathbf{c}^T\mathbf{x} = \frac{1}{2} + \frac{1}{4} + 1 = \frac{7}{4} = 1.75. \tag{7.1}$$

Assim, temos

$$A = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & -1 \end{pmatrix} b = \begin{pmatrix} 2 \\ 1 \end{pmatrix} c = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

Calculamos

$$D = \begin{pmatrix} 1/2 & 0 & 0 \\ 0 & 1/4 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad D^2 = \begin{pmatrix} 1/4 & 0 & 0 \\ 0 & 1/16 & 0 \\ 0 & & 1 \end{pmatrix}.$$

$$
\begin{aligned}
\mathbf{c}_p &= \left[\mathcal{I} - DA^T[AD^2A^T]^{-1}AD\right] D\mathbf{c} \\
&= \left[\mathcal{I} - \begin{pmatrix} 1/2 & 1 \\ 1/2 & 1 \\ 1 & -1 \end{pmatrix} \left[\begin{pmatrix} 3/2 & 0 \\ 0 & 3 \end{pmatrix}\right]^{-1} \begin{pmatrix} 1/2 & 1/2 & 1 \\ 1 & 1 & -1 \end{pmatrix}\right] \begin{pmatrix} 1/2 & 0 & 0 \\ 0 & 1/4 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \\
&= \left[\mathcal{I} - \begin{pmatrix} 1/2 & 1 \\ 1/2 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 2/3 & 0 \\ 0 & 1/3 \end{pmatrix} \begin{pmatrix} 1/2 & 1/2 & 1 \\ 1 & 1 & -1 \end{pmatrix}\right] \begin{pmatrix} 1/2 & 0 & 0 \\ 0 & 1/4 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} 1/4 & -1/8 & 0 \\ -1/4 & 1/8 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} 1/8 \\ -1/8 \\ 0 \end{pmatrix}
\end{aligned}
$$

Calculamos também

$$\mathbf{d} = D\mathbf{c}_p = \begin{pmatrix} 1/2 & 0 & 0 \\ 0 & 1/4 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1/8 \\ -1/8 \\ 0 \end{pmatrix} = \begin{pmatrix} 1/16 \\ -1/32 \\ 0 \end{pmatrix}$$

Determinamos

$$\delta = \min_{j \le n} \left\{ -\frac{x_j}{d_j} : d_j < 0 \right\} = -\frac{1/4}{-1/32} = 8.$$

O novo ponto será

$$
\begin{aligned}
\mathbf{x}' &= \mathbf{x} + \alpha\delta\mathbf{d} \\
&= \begin{pmatrix} 1/2 \\ 1/4 \\ 1 \end{pmatrix} + \alpha(8) \begin{pmatrix} 1/16 \\ -1/32 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 1/2 \\ 1/4 \\ 1 \end{pmatrix} + \alpha \begin{pmatrix} 1/2 \\ -1/4 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 1/2 + \alpha/2 \\ 1/4 - \alpha/4 \\ 1 \end{pmatrix}
\end{aligned}
$$

Se fizermos $\alpha = 0.95$, teremos

$$\mathbf{x}' = \begin{pmatrix} 0.5 + 0.95 \\ 0.25 - 0.2375 \\ 1 \end{pmatrix} = \begin{pmatrix} 0.975 \\ 0.0125 \\ 1 \end{pmatrix}$$

A nova solução é viável: além de ser positiva,

E seu valor é

$$\mathbf{c}^T\mathbf{x}' = 0.975 + 0.0125 + 1 = 1.9875,$$

maior que o que tínhamos antes (1.75, conforme a equação 7.1).

Continuamos até que a norma de $\mathbf{c}_p$ seja suficientemente pequena (por exemplo, menor que 0.001). ◀

Descrevemos a seguir o algoritmo em pseudocódigo. Esta é a versão para *maximização*.

repita :
    $D \leftarrow \text{diag}(\mathbf{x})$
    $\mathbf{c}_P \leftarrow \left(\mathcal{I} - DA^T[AD^2A^T]^{-1}AD\right)Dc$
    se $\|\mathbf{c}_p\|_\infty < \varepsilon$
        `PARE -- solução suficientemente boa`
    $\mathbf{d} \leftarrow D.\mathbf{c}_p$
    se $d_i > 0$ **para** `todo i`
        `PARE -- problema ilimitado`
    $\delta \leftarrow \varepsilon_\delta \min_j \left\{ -\frac{x_j}{d_j} : d_j < 0 \right\}$
    $\mathbf{x} \leftarrow \mathbf{x} + \delta\mathbf{d}$

7.2.2 Métodos de barreira

Notas

Descrições do algoritmo do elipsoide podem ser encontradas nos livros de Griva, Nash e Sofer [GNS09] e de Papadimitriou e Steiglitz [PS98], que discutem inclusive a representação computacional (o algoritmo calcula raízes quadradas, sendo necessário representar irracionais).

O algoritmo usando *affine scaling* apresentado neste texto é uma simplificação do algoritmo de Karmarkar, elaborado por Narendra Karmarkar [Kar84].

Exercícios

Ex. 59 — Mostre como obter uma solução viável inicial para o algoritmo *affine scaling*.

Ex. 60 — O algoritmo *affine scaling* poderia ser usado em um problema de otimização com função objetivo não-linear, já que só precisa do vetor *gradiente* do obejtivo. Determine que funções não-lineares poderiam ser otimizadas por este algoritmo.

Ex. 61 — Implemente o algoritmo *affine scaling*.

Capítulo 8

Programação Inteira

Como mencionamos na seção 2.3, é comum precisarmos resolver um problema de otimização linear onde só nos interessam as soluções inteiras.

Um problema de programação inteira é como um problema de programação linear, exceto que restringimos parte das variáveis a valores inteiros:

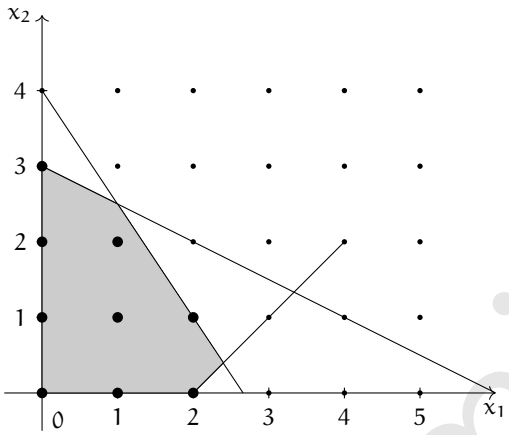
$$\begin{aligned} &\max \mathbf{c}^T\mathbf{x} \\ &\text{s.a.} : A\mathbf{x} = \mathbf{b} \\ &\quad \mathbf{x} \geq \mathbf{0} \\ &\quad x_j \in \mathbb{Z}, \text{ se } j \in K \end{aligned}$$

onde o conjunto K contém os índices das variáveis inteiras.

Exemplo 8.1. Considere o problema a seguir.

$$\begin{aligned} &\max x_1 + x_2 \\ &\text{s.a.:} 3x_1 + 2x_2 \leq 8 \\ &\quad x_1 + 2x_2 \leq 6 \\ &\quad x_1 - x_2 \leq 2 \\ &\quad \mathbf{x} \geq \mathbf{0} \\ &\quad \mathbf{x} \in \mathbb{Z}^2 \end{aligned}$$

Note que tanto x_1 como x_2 devem ser inteiros. As restrições são mostradas na figura a seguir. Os pontos viáveis, no entanto, são somente os pontos *inteiros* dentro da região.



Os únicos pontos viáveis são

$$(0,0),(1,0),(2,0)$$
$$(0,1),(0,2),(0,3),$$
$$(1,1),(1,2),(2,1).$$

Veja que dentre os extremos do poliedro, somente alguns são inteiros:

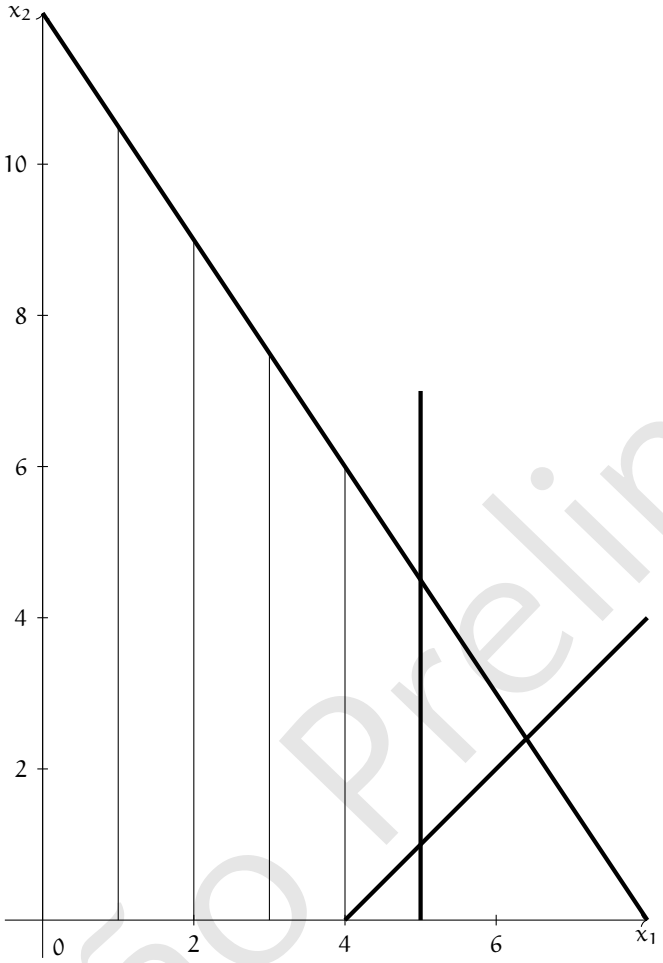
$$(0,0),(2,0),(0,3),(1,2.5),(2.4,0.4).$$

Em particular, se desconsiderarmos a exigencia de integralidade, o ótimo seria $(1,2.5)$, que não é inteiro. O melhor ponto inteiro de acordo com a função objetivo dada é $(1,2)$. ◀

Exemplo 8.2. Considere o problema a seguir.

$$\begin{aligned} \max\ & x_1 + x_2 \\ \text{s.a.:}\ & 3x_1 + 2x_2 \leq 24 \\ & x_1 \leq 5 \\ & x_1 - x_2 \leq 4 \\ & \mathbf{x} \geq \mathbf{0} \\ & x_1 \in \mathbb{Z} \\ & x_2 \in \mathbb{R} \end{aligned}$$

Neste problema x_1 assume valores inteiros,e x_2 é real. Isto significa que a região viável é a união das linhas verticais na figura a seguir.



Os pontos viáveis são aqueles nos segmentos de reta

$$(0,0) - (0,12)$$
$$(1,0) - (1,10.5)$$
$$(2,0) - (2,9.0)$$
$$(3,0) - (3,7.5)$$
$$(4,0) - (4,6.0)$$
$$(5,1) - (5,4.5)$$

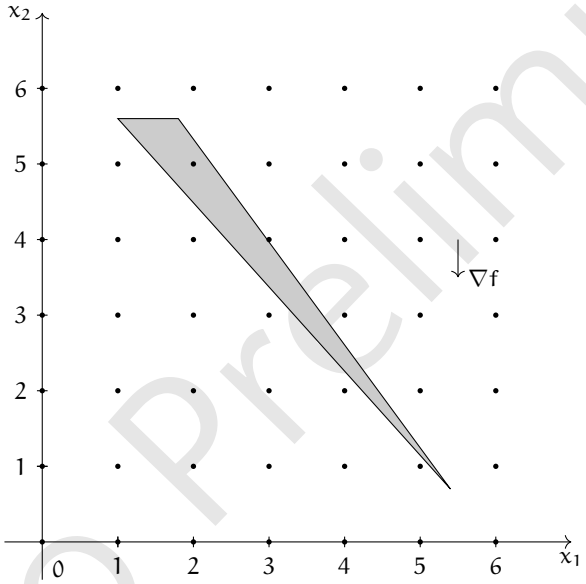
◀

Neste texto trataremos de dois métodos para resolver programas inteiros. O primeiro é o método dos planos de corte, que consiste em determinar novas desigualdades que reduzam o poliedro das soluções viá-

veis, facilitando a obtenção de uma solução inteira. O segundo é o método *branch-and-bound*, que enumera de forma inteligente as soluções.

8.1 Planos de corte

Suponha que queiramos resolver um programa linear inteiro de minimização e que a região viável seja como a que é mostrada na próxima figura. A seta mostra a direção para a qual o valor da função objetivo f decresce.



Suponha que queiramos tentar resolver o problema relaxando a restrição de integralidade e usando, por exemplo, o método Simplex. Obtemos uma solução não inteira. A partir dela, temos duas opções:

- Se tentarmos usar a solução inteira mais próxima, chegaremos ao ponto $(5, 1)$ fora do politopo (e portanto inviável).
- Se tentarmos percorrer o politopo procurando uma solução inteira viável, teremos que andar uma grande distância, já que a solução inteira viável ótima, $(2, 5)$, está longe de $(5, 1)$.

Podemos tentar adicionar restrições ao programa linear, de forma que a região viável fique menor, mas que nenhuma solução inteira seja removida. Essas restrições são chamadas de *planos de corte*, ou simplesmente *cortes*.

Suponha que tenhamos usado o método Simplex para resolver a versão relaxada do programa linear inteiro. O tableau terá a descrição das variáveis básicas em função das não básicas (que valem zero) e de $\mathbf{b}$.

$$x_i + \sum_{j>m} a_{ij} x_j = b_i \quad i \leq m \tag{8.1}$$

Como x_j não é negativo,

$$\sum_{j>m} \lfloor a_{ij} \rfloor x_j \leq \sum_{j>m} a_{ij} x_j,$$

e portanto

$$x_i + \sum_{j>m} \lfloor a_{ij} \rfloor x_j \leq b_i$$

Como procuramos um valor inteiro para x_i, e esta solução determina que $x_i = b_i$, podemos usar $\lfloor b_i \rfloor$ sem alterar a solução inteira.

$$x_i + \sum_{j>m} \lfloor a_{ij} \rfloor x_j \leq \lfloor b_i \rfloor \quad i \leq m \tag{8.2}$$

Subtraindo 8.2 de 8.1, obtemos

$$\sum_{j>m} (a_{ij} - \lfloor a_{ij} \rfloor) x_j \geq b_i - \lfloor b_i \rfloor$$

Observe que a variável x_i básica desaparece. Esta desigualdade é o *corte de Gomory* para a i-ésima linha, e a adicionamos diretamente ao tableau Simplex, após multiplicá-la por -1 e adicionar uma variável de folga s, *que passa a ser básica*, com valor $-\lfloor b_i \rfloor$:

$$-\sum_{j>m} (a_{ij} - \lfloor a_{ij} \rfloor) x_j + s = -b_i + \lfloor b_i \rfloor$$

Desta discussão fica claro o Teorema a seguir.

Teorema 8.3. *O corte de Gomory não exclui solução inteira.*

O próximo Teorema também é parte da essência do método dos planos de corte.

Teorema 8.4. *Após a adição de um corte de Gomory, o tableau torna-se inviável para o primal e viável para o dual.*

Demonstração. Se b_i não era inteiro, então $\lfloor b_i \rfloor > 0$, e o tableau agora tem a linha $s = -\lfloor b_i \rfloor$, e portanto é inviável para o primal.

A viabilidade para o dual segue trivialmente do fato de não termos modificado a última linha do tableau, onde estão os coeficientes relativos de custo $c_j - z_j$ (que são a solução para o dual). ■

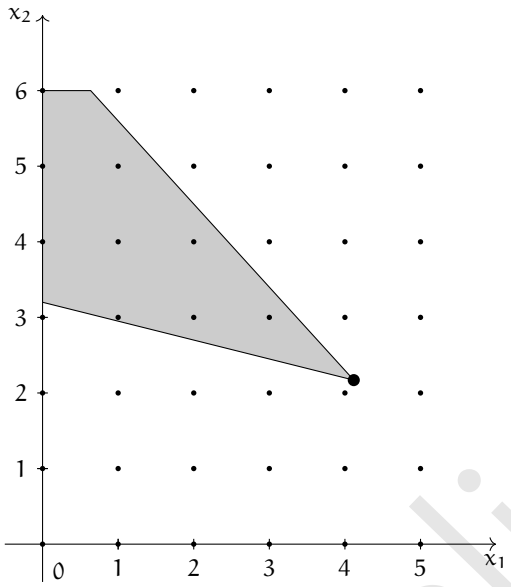
Podemos então prosseguir com o método Simplex dual, obtendo uma nova solução. Se a nova solução for inteira, encontramos o ótimo. Se for fracionária, recomeçamos e adicionamos um novo corte. Se não houver solução viável, é porque não há solução inteira para o problema original.

É possível garantir que o método dos planos de corte sempre termine em tempo finito, usando determinada disciplina nas escolhas feitas durante a execução do método Simplex. Não faremos a demonstração neste texto.

Exemplo 8.5. Mostraremos neste exemplo a adição de um plano de corte ao programa linear a seguir, que exige solução inteira.

$$\begin{aligned} \min \quad & -x_1 + 5x_2 \\ \text{s.a:} \quad & x_2 \leq 6 \\ & 0.5x_1 + 2x_2 \geq 6.4 \\ & 1.1x_1 + x_2 \leq 6.7 \\ & \mathbf{x} \geq \mathbf{0} \\ & \mathbf{x} \in \mathbb{Z}^2 \end{aligned}$$

Quando resolvemos o problema sem as restrições de integralidade, obtemos a solução $(4.117, 2.110)$. No entanto, como podemos ver na figura a seguir, esta solução está longe da única solução inteira para o problema.



Escolhemos uma variável para adicionar o corte de Gomory. O tableau Simplex é

$$\left(\begin{array}{cccccc|c} x_1 & x_2 & w_1 & & w_2 & w_3 & b_i \\ \hline 0 & 0 & 1 & & 0.323529 & 0.294118 & 3.82941 \\ 0 & 1 & 0 & & -0.323529 & -0.294118 & 2.17059 \\ 1 & 0 & 0 & & 0.294118 & 1.17647 & 4.11765 \\ \hline 0 & 0 & 0 & & 1.91176 & 2.64706 & 6.73529 \end{array}\right)$$

A linha de x_1 no tableau é

$$x_1 - 0.294118w_2 - 1.176472w_3 = 4.11765$$

Construimos o corte de Gomory:

$$\begin{aligned} (0.294118 - \lfloor 0.294118 \rfloor)w_2 + (1.17647 - \lfloor 1.17647 \rfloor)w_3 &\geq 4.11765 - \lfloor 4.11765 \rfloor \\ 0.294118w_2 + 0.17647w_3 &\geq 0.11765 \qquad (*) \\ 0.294118w_2 + 0.17647w_3 - s &= 0.11765 \\ -0.294118w_2 - 0.17647w_3 + s &= -0.11765 \end{aligned}$$

A nova variável de folga é

$$s = -0.11765 + 0.294118w_2 + 0.17647w_3$$

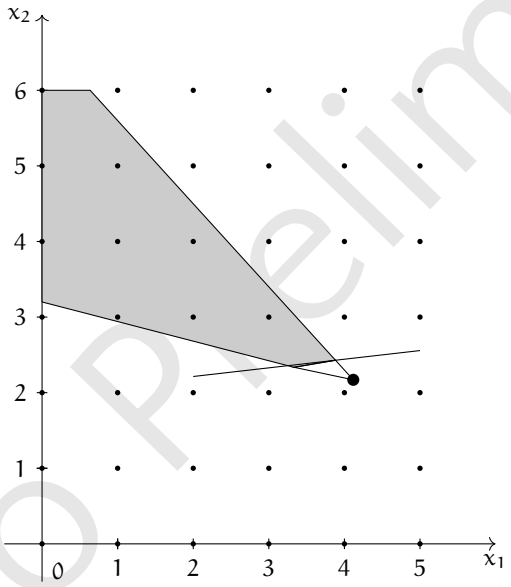
Podemos visualizar a nova restrição (marcada com (*)) se pudermos escrevê-la usando apenas x_1 e x_2. Usando o problema original, substituímos

$$w_2 = -6.4 + 0.5x_1 + 2x_2$$
$$w_3 = +6.7 - 1.1x_1 - x_2$$

Assim obtemos a restrição

$$-0.411766x_2 + 0.047058x_1 + 0.7000062 \geq -0.11765.$$

é mostrada no gráfico da região viável a seguir.



Observe que, como já demonstramos, o corte de Gomory não exclui solução inteira da região viável. Por outro lado, ele exclui a solução ótima não inteira.

Agora adicionamos a restrição ao tableau. Com isso, adicionamos a variável s, básica, com valor -0.11765, tornando a solução inviável para o primal:

$$\left(\begin{array}{cccccc:c} x_1 & x_2 & w_1 & w_2 & w_3 & s & b_i \\ \hline 0 & 0 & 1 & 0.323529 & 0.294118 & 0 & 3.82941 \\ 0 & 1 & 0 & -0.323529 & -0.294118 & 0 & 2.17059 \\ 1 & 0 & 0 & 0.294118 & 1.17647 & 0 & 4.11765 \\ 0 & 0 & 0 & -0.294118 & -0.17647 & 1 & -0.11765 \\ \hdashline 0 & 0 & 0 & 1.91176 & 2.64706 & 0 & 6.73529 \end{array}\right)$$

Para continuar, observamos que o tableau ainda é viável para o dual, mas que não é ótimo para ele, porque se fosse seria ótimo para o primal, e portanto viável aos dois.

Após obter nova solução (usando o dual), verificamos se é inteira. Se não for, obtemos um novo corte de Gomory e repetimos o processo. ◀

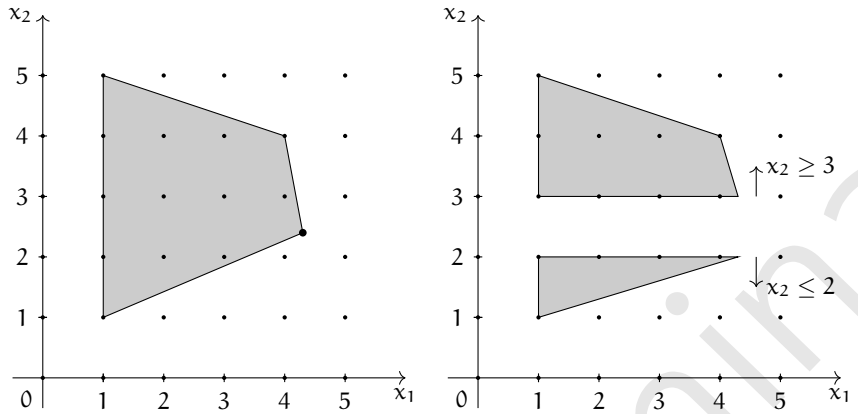
8.2 *Branch-and-bound*

Suponha que tenhamos resolvido a versão relaxada de um programa inteiro, e que a solução obtida, $\hat{\mathbf{x}}$, tem um elemento não inteiro $\hat{x}_i$. Resolvemos então dois problemas, um presumindo que $x_i \geq \lfloor \hat{x}_i \rfloor + 1$ e outro presumindo que $x_i \leq \lfloor \hat{x}_i \rfloor$.

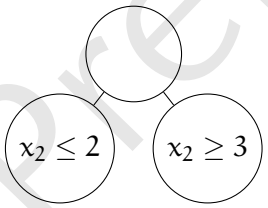
$$\begin{aligned} \max \mathbf{c}^T\mathbf{x} \text{ s.a : } & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \\ & x_j \in \mathbb{Z}, \text{ se } j \in K \\ & x_i \leq \lfloor \hat{x}_i \rfloor \end{aligned}$$

$$\begin{aligned} \max \mathbf{c}^T\mathbf{x} \text{ s.a : } & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \\ & x_j \in \mathbb{Z}, \text{ se } j \in K \\ & x_i \geq \lfloor \hat{x}_i \rfloor + 1 \end{aligned}$$

Exemplo 8.6. As duas figuras a seguir ilustram como a região viável é dividida em duas. Na primeira figura, a solução fracionária $(4.3, 2.4)$ é obtida. A variável x_2 é escolhida para criar uma ramificação, e na segunda figura temos as regiões viáveis para os dois programas lineares – um com $x_3 \leq 2$ e outra com $x_2 \geq 3$.



Cada vez que dividimos o problema, criamos dois outros, derivados do primeiro. Isto é naturalmente representado como uma árvore binária, onde cada nó é um problema de programação linear. A ramificação do exemplo que demos é representada como árvore a seguir.



◀

Um *limite superior* para o valor do objetivo do problema original é obtido facilmente da solução de qualquer um dos problemas relaxados.

Um *limite inferior* para o valor do objetivo do problema original é obtido quando um dos problemas relaxados tem solução inteira (pode-se ainda usar diversas técnicas para obter uma solução inteira viável a partir de uma solução fracionária).

Suponha que para um nó a da árvore tenhamos determinado que a melhor solução possível é α, e que para outro nó b tenhamos verificado que a pior solução terá valor no mínimo $\beta > \alpha$. Podemos evidentemente abandonar o nó α, já que qualquer solução que pudesse ser obtida ali será pior (ou no máximo tão boa quanto) as soluções obtidas a partir de β.

Há escolhas que podem interferir na eficiência do algoritmo:

- Tendo uma árvore com vários nós, a partir de qual deles faremos a próxima ramificação;

- Uma vez que tenhamos escolhido um nó da árvore, deveremos escolher uma das variáveis com valor não inteiro para criar a próxima ramificação.

8.3 Variantes: *branch-and-cut*, *branch-and-price*

O nome *branch-and-cut* é dado à combinação das técnicas descritas anteriormente (planos de corte e *branch-and-bound*. Após resolver uma versão relaxada do problema, pode-se adicionar um corte ou uma ramificação.

Um corte gera uma nova linha no tableau Simplex. Pode-se também gerar novas colunas – este é o método chamado de *pricing*, que quando combinado com *branch-and-bound* é chamado de *branch-and-price*.

8.4 Unimodularidade e poliedros integrais

Há também uma classe de problemas para os quais a região viável sempre é um poliedro cujos pontos extremos tem coordenadas inteiras. Um exemplo são os problemas de transporte, abordados no Capítulo 10. Nesta Seção tratamos brevemente desses problemas.

Definição 8.7. Um poliedro em $\mathbb{R}^n$ é *integral* se seus pontos extremos pertencem a $\mathbb{Z}^n$. ⧫

Definição 8.8. Uma matriz é *totalmente unimodular* se todos os seus subdeterminantes são iguais a $+1$, -1 ou 0. ⧫

Claramente, todos os elementos de uma matriz totalmente unimodular devem ser $+1$, -1 ou 0, caso contrário a submatriz 1×1 com cada elemento não seria unimodular. No entanto, nem toda matriz com elementos $+1$, -1 e 0 é totalmente unimodular: por exemplo, a matriz a seguir tem determinante igual a 2.

$$\begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

Teorema 8.9. *Se A é totalmente unimodular, então também o são:*

- *A transposta de A.*
- *As matrizes obtidas removendo linhas ou colunas inteiras de A.*
- *As matrizes obtidas multiplicando linhas ou colunas inteiras por -1.*

- *As matrizes obtidas por reordenação de linhas ou colunas.*
- *As matrizes obtidas por operação de pivoteamento (passo do Simplex).*
- (A, I).

Lema 8.10. *Se* A *e* $\mathbf{b}$ *tem apenas valores inteiros então* $A\mathbf{x} = \mathbf{b}$ *tem solução integral quando* $\det(A) = \pm 1$.

Demonstração. Usamos a regra de Cramer: seja A^j a matriz A onde a j-ésima coluna foi trocada pelo vetor $\mathbf{b}$. então $x_j = \frac{\det(A^j)}{\det(A)}$.

Quando $\det(A)$ é ± 1, x_j tem o mesmo valor absoluto que $\det(A^j)$. Como os elementos de A e de $\mathbf{b}$ são inteiros, $\det(A^j)$ é inteiro, e também o serão todos os x_j. ■

Teorema 8.11. *Se* A *é totalmente unimodular e* $\mathbf{b} \in \mathbb{Z}^n$*, então os pontos extremos do poliedro* $\{\mathbf{x} : A\mathbf{x} \leq \mathbf{b}\}$ *são inteiros.*

Demonstração. Os vértices de $A\mathbf{x} \leq \mathbf{b}$ são definidos por A', uma submatriz quadrada de A. Como A é totalmente unimodular, A' também é. $A'\mathbf{x} = \mathbf{b}'$ tem solução inteira quando $\det(A') = \pm 1$, e portanto $\mathbf{x}$ é integral. ■

Teorema 8.12. *Uma matriz* A *é totalmente unimodular se:*

i) todos seus elementos são $0, +1$ *ou* -1*;*

ii) cada coluna tem no máximo dois elementos não-zero;

iii) as linhas podem ser particionadas em dois conjuntos, A *e* B*, de forma que para cada coluna com dois elementos não-zero, a soma dos elementos em* A *é igual à soma dos elementos em* B*.*

Exemplo 8.13. A matriz

$$A = \begin{pmatrix} 1 & 0 & -1 & 0 & 1 \\ -1 & 1 & 0 & 1 & 0 \\ 0 & 1 & -1 & 0 & 0 \end{pmatrix}$$

é totalmente unimodular, porque satisfaz as condições do Teorema 8.11 (particione as linhas em $(1, 2)$; (3)). Logo, a solução ótima para $\max A\mathbf{x} = \mathbf{b}$ deve também ser inteira se $\mathbf{b}$ for inteiro. ◄

Exemplo 8.14. A matriz

$$A = \begin{pmatrix} 1 & 0 & 1 & 0 \\ -1 & -1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

não é unimodular: o determinante da submatriz quadrada 3×3 formada pelas tres primeiras colunas é

$$\det \begin{pmatrix} 1 & 0 & 1 \\ -1 & -1 & 0 \\ 0 & 1 & 1 \end{pmatrix} = -2.$$

Assim, embora a matriz satisfaça os critérios (i) e (ii) do Teorema 8.12, não é unimodular, e não satisfaz o critério (iii): não há como particionar as linhas da forma como determina o Teorema. ◀

O Teorema 8.12 é condição suficiente, mas não necessária para unimodularidade.

Exemplo 8.15. A matriz

$$\begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

É unimodular, e não satisfaz os critérios (ii) e (iii) do Teorema. ◀

Notas

O livro de Papadimitriou e Steiglitz [PS98] dá uma boa introdução à Programação Linear Inteira, incluindo os métodos de planos de corte e *branch-and-bound*. Há também o livro de Laurence Wolsey [Wol98], o de Wolsey e Nemhauser [WN99] e o de Korte [KV12]. Para uma visão mais completa da área veja o livro de Schriver [Sch03]. O livro de Goldbarg e Luna [GL00] cita numerosas técnicas usadas no desenvolvimento dos detalhes do algoritmo de *branch-and-bound*.

O método dos planos de corte foi desenvolvido por R. Gomory [Gom58].

O método de *branch-and-bound*, no contexto de Otimização Combinatória, foi proposto por Land e Doig [LD60] é semelhante à técnica de busca usando cortes $\alpha - \beta$, usada em Inteligência Artificial [NR10].

O método de *branch-and-price* foi descrito por Barnhart, Johnson, Nemhauser, Savelsbergh e Vance [Bar+98].

O livro de Alexander Schriver [Sch03] elabora o tópico da unimodularidade de matrizes e sua conexão com programação inteira.

Politopos relacionados a diferentes classes de problemas são estudados no volume editado por David Avis, David Bremner e Antoine Deza [ABD09].

Exercícios

Ex. 62 — Calcule e plote os outros dois cortes de Gomory, que não foram feitos no exemplo 8.5.

Ex. 63 — Implemente o algoritmo dos planos de corte.

Ex. 64 — A recíproca do Teorema 8.11 vale?

Ex. 65 — Prove o Teorema 8.9.

Ex. 66 — Suponha que um poliedro P definido por um programa linear seja integral (ou seja, $\mathbf{b}$ é inteiro e A é totalmente unimodular). Prove que após a execução do método Simplex os coeficientes reduzidos de custo do programa linear são inteiros.

Ex. 67 — Prove o Teorema 8.12.

Ex. 68 — Suponha que A não é totalmente unimodular, mas tem coeficientes inteiros, e que $\mathbf{b}$ seja integral. Quando a solução para $A\mathbf{x} = \mathbf{b}$ é integral? Responda para matrizes quadradas de ordem 2 e 3.

Ex. 69 — Prove que a matriz de incidência de um grafo bipartido é totalmente unimodular.

Ex. 70 — Unimodularidade total é fechada para multiplicação de matrizes? Para potência de matriz?

Ex. 71 — Desenvolva um algoritmo para gerar matrizes aleatóreas totalmente unimodulares, respeitando o critério do teorema 8.12.

Ex. 72 — Desenvolva um algoritmo para gerar matrizes aleatóreas totalmente unimodulares, *não* respeitando o critério do teorema 8.12. Não é necessário garantir que seu algoritmo gere as matrizes com distribuição uniforme.

Capítulo 9

Decomposição de Dantzig-Wolfe

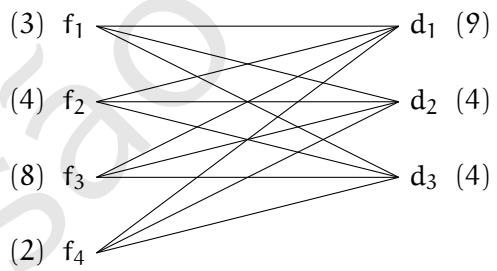
Parte II

Aplicações

Capítulo 10

Problemas de Transporte

Temos m fontes e n destinos; a produção da i-ésima fonte é f_i e a demanda do j-ésimo destino é d_j. O custo para levar cada unidade de f_i até d_j é t_{ij}. A oferta total (somando a oferta de cada fonte) é igual à demanda total. O problema consiste em conseguir enviar toda a produção, satisfazendo toda a demanda, mas minimizando o custo total. A figura a seguir mostra um problema de transporte. Há quatro fontes, com oferta igual a $3, 4, 8, 2$, e três destinos com demandas $9, 4, 4$.



Problemas de transporte podem ser modelados como programas lineares. Em um problema de transporte a oferta total é igual à demanda total, portanto temos

$$\sum_i f_i = \sum_j d_j$$

O problema então é

$$\begin{aligned} \min & \sum_i \sum_j t_{ij} x_{ij} \\ \text{s.a.}: & \sum_j x_{ij} = f_i \quad (\forall\, i \leq m) \\ & \sum_i x_{ij} = d_j \quad (\forall\, j \leq n) \\ & \mathbf{x} \geq \mathbf{0} \end{aligned} \tag{10.1}$$

Exemplo 10.1. Suponha que tenhamos as fontes e destinos a seguir, com as respectivas ofertas e demandas:

$$\begin{array}{ll} f_1 = 4 & d_1 = 12 \\ f_2 = 10 & d_2 = 8 \\ f_3 = 6 & \end{array}$$

O custos de transporte são

$$\begin{pmatrix} 4 & 7 \\ 9 & 5 \\ 5 & 4 \end{pmatrix}$$

Modelamos este problema como

$$\begin{aligned} \min\ & 4x_{11} + 7x_{12} + 9x_{21} + 5x_{22} + 5x_{31} + 4x_{32} \\ \text{s.a.}:\ & x_{11} + x_{12} = 4 \\ & x_{21} + x_{22} = 10 \\ & x_{31} + x_{32} = 6 \\ & x_{11} + x_{21} + x_{31} = 12 \\ & x_{12} + x_{22} + x_{32} = 8 \\ & \mathbf{x} \geq 0, \end{aligned}$$

ou como $\min \mathbf{c}^T\mathbf{x}$, $A\mathbf{x} = \mathbf{b}$, com

$$\mathbf{x} = \begin{pmatrix} x_{11} \\ x_{12} \\ x_{21} \\ x_{22} \\ x_{31} \\ x_{32} \end{pmatrix}, \quad \mathbf{c} = \begin{pmatrix} 4 \\ 7 \\ 9 \\ 5 \\ 5 \\ 4, \end{pmatrix}$$

e

$$A = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 4 \\ 10 \\ 6 \\ 12 \\ 8 \end{pmatrix}. \qquad \blacktriangleleft$$

Ao escrevermos o problema na forma matricial, observamos que a matriz das restrições tem a seguinte estrutura:

$$\begin{array}{cccccccccccccc|l}
x_{11} & + & x_{12} & + & \dots & + & x_{1n} & & & & & & & & = f_1 \\
 & & & & & & & x_{21} & + & x_{22} & \dots & + & x_{2n} & & = f_2 \\
 & & & & & & & & & & & & & \ddots & \vdots \\
 & & & & & & & & & & & & & & = f_m \\
\hline
x_{11} & & & & & & & + & x_{21} & & & & & \cdots & = d_1 \\
 & & x_{12} & & & & & & & + & x_{22} & & & & \vdots \\
 & & & \ddots & & & & & & & \ddots & & & & = d_n \\
 & & & & & & x_{1_n} & & & & & & + & x_{2n} \quad \cdots & = d_n
\end{array} \tag{10.2}$$

Ou seja,

$$A = \begin{pmatrix} \mathbf{1}^T & & & \\ & \mathbf{1}^T & & \\ & & \ddots & \\ & & & \mathbf{1}^T \\ \mathcal{I} & \mathcal{I} & \mathcal{I} & \mathcal{I} \end{pmatrix}$$

A matriz do exemplo 10.1, por exemplo, é

$$A = \left(\begin{array}{cc|cc|cc} 1 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 1 \\ \hline 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{array}\right)$$

Qualquer problema que possa ser descrito dessa forma é chamado de *problema de transporte*. Mostramos agora que todo problema de transporte tem solução ótima, porque todos tem solução viável e todos são limitados.

Teorema 10.2. *Todo problema de transporte tem uma solução viável.*

Demonstração. Seja X a demanda total (igual à oferta total). Então $x_{ij} = \frac{f_i d_j}{X}$ é viável, já que

$$\begin{aligned}\sum_j x_{ij} &= \sum_j \frac{f_i d_j}{X} \\ &= \frac{f_i}{X} \sum_j d_j \\ &= \frac{f_i}{X} X \\ &= f_i,\end{aligned}$$

e semelhantemente para $\sum_i x_{ij} = d_j$. ■

Exemplo 10.3. No exemplo 10.1, temos $X = 20$, e uma solução viável é

$$\begin{aligned}x_{11} &= a_1 d_1/20 = 12/5 \\ x_{12} &= a_1 d_2/20 = 8/5 \\ x_{21} &= a_2 d_1/20 = 6 \\ x_{22} &= a_2 d_2/20 = 4 \\ x_{31} &= a_3 d_1/20 = 18/5 \\ x_{32} &= a_3 d_2/20 = 12/5.\end{aligned}$$ ◄

Teorema 10.4. *Todo problema de transporte é limitado.*

Demonstração. Como cada x_{ij} é limitado por a_i e d_j, o problema é também limitado. ■

10.1 Solução básica inicial

A solução viável que determinamos anteriormente, $x_{ij} = \frac{f_i d_j}{X}$, não é básica. Nesta seção tratamos de como obter uma solução viável inicial que também seja básica.

Podemos representar o sistema como uma tabela com m linhas e n colunas, onde cada linha da tabela corresponde a uma linha $\sum_j x_{ij} = f_i$ e cada coluna da tabela corresponde a uma linha $\sum_i x_{ij} d_j$.

x_{11}	x_{12}	...	x_{1n}	f_1
x_{21}	x_{22}		x_{2n}	f_2
$\vdots$				$\vdots$
x_{m1}	x_{m2}		x_{mn}	f_m
d_1	d_2	...	d_n	

(10.3)

A *regra do canto noroeste* permite obter uma solução viável básica inicial.

O algoritmo começa com a célula no canto superior esquerdo. Alocamos tanto quanto puder (isso significa que estamos determinando quanto da fonte f_1 vai para o destino d_1), sem violar a viabilidade da solução. Se há oferta sobrando, passamos para uma célula para a direita (porque já saturamos a capacidade de d_1). Senão, vamos para a de baixo (porque a capacidade de f_1 foi completamente usada). Se não sobrou oferta ou demanda, vamos para uma célula para baixo *e* uma para a direita. Repetimos enquanto for possível mover-se para as células de baixo ou da esquerda.

Exemplo 10.5. Considere um problema de transporte com $\mathbf{f}$ e $\mathbf{d}$ mostrados a seguir.

$$\mathbf{f} = \begin{pmatrix} 22 & 4 & 17 \end{pmatrix}$$
$$\mathbf{d} = \begin{pmatrix} 9 & 27 & 7 \end{pmatrix}$$

Começamos montando uma tabela 3×3 representando as variáveis x_{ij}. Alocamos o máximo que podemos em x_{11}: como $f_1 = 22$ e $d_1 = 9$, alocamos 9 e movemos para a direita. Ainda faltam 13 unidades de f_1 para alocar. Como $d_2 = 27$, terminamos alocando todas as 13 em f_{12}.

				f_i
	9	13		**22**
				4
				17
d_j	**9**	**27**	**7**	

Continuamos agora descendo para a linha debaixo, mas ainda na coluna 2. $f_2 = 4$ e ainda sobram 14 da demanda de d_2, portanto alocamos todos os 4 e movemos novamente para a linha debaixo (ainda podemos alocar 10 em d_2. Das 17 unidades de f_3, 10 podem ser enviadas para d_2. As outras 7 vão para d_3. A tabela final é mostrada a seguir.

				f_i
	9	13		**22**
		4		**4**
		10	7	**17**
d_j	**9**	**27**	**7**	

Temos agora uma solução viável básica: as somas das linhas resultam nos f_i, e as somas das colunas resultam nos d_j. As variáveis básicas são $x_{11} = 9$, $x_{12} = 13$, $x_{21} = 4$, $x_{32} = 10$, e $x_{33} = 7$. Todos os outros x_{ij} ficam fora da base e valem zero. ◀

Há métodos diferentes para obter soluções viáveis básicas.

10.2 Solução inteira

O último Teorema desta seção é importante porque identifica os problemas de transporte como uma classe de problemas para as quais podemos obter soluções inteiras de forma eficiente (isto é relevante porque de maneira geral, obter soluções inteiras para problemas de otimização linear é difícil – veja o Capítulo 8).

Teorema 10.6. *Todo problema de transporte com ofertas e demandas inteiras tem solução inteira.*

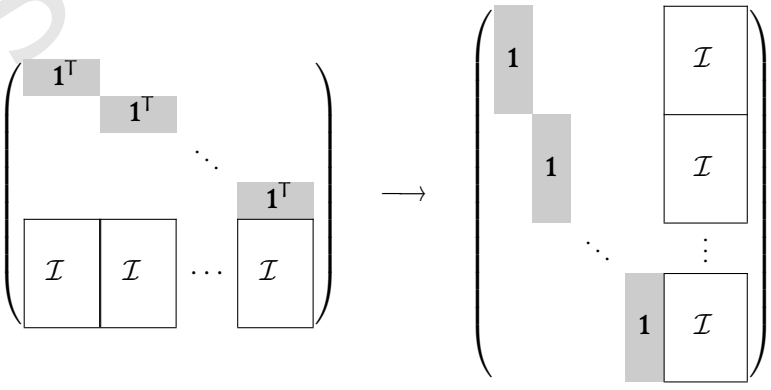
Demonstração. A matriz de coeficientes do programa linear é totalmente unimodular: aplica-se o teorema 8.12 (particione a matriz em linhas de restrição de demanda e linhas de restrição de oferta). ■

Pode-se também demonstrar este Teorema sem usar a unimodularidade da matriz; veja o Apêndice A.

10.3 Algoritmo para solução do problema de transporte

O método Simplex pode ser adaptado para explorar a estrutura dos problemas de transporte.

O diagrama a seguir representa de forma geral a matriz de coeficientes das restrições do primal de um problema de transporte e a de seu dual. Os retângulos em cinza (vetores com uns) estão alinhados com as submatrizes identidade.



O dual do problema de transporte descrito em (10.1) é, portanto,

$$\begin{aligned} \max & \sum_{i \le m} f_i u_i + \sum_{j \le n} d_j v_j \\ \text{s.a.} : & \; u_i + v_j \le t_{ij} \\ & \mathbf{u} \in \mathbb{R}^m \\ & \mathbf{v} \in \mathbb{R}^n \end{aligned}$$

Note que o vetor objetivo no dual foi particionado, para refletir mais claramente a estrutura do problema. As variáveis do dual são os u_i (associados às linhas superiores da matriz de restrições do primal) e os v_j (associados às linhas inferiores). As variáveis do dual são, como já vimos anteriormente, os coeficientes reduzidos de custo do primal.

Exemplo 10.7. No exemplo 10.1 apresentamos o problema de transporte com

$$\begin{array}{ll} f_1 = 4 & d_1 = 12 \\ f_2 = 10 & d_2 = 8 \\ f_3 = 6 & \end{array}$$

e custos

$$\begin{pmatrix} 4 & 7 \\ 9 & 5 \\ 5 & 4 \end{pmatrix}$$

Modelamos este problema como $\min \mathbf{c}^T\mathbf{x}$, $A\mathbf{x} = \mathbf{b}$, com

$$\mathbf{x} = \begin{pmatrix} x_{11} \\ x_{12} \\ x_{21} \\ x_{22} \\ x_{31} \\ x_{32} \end{pmatrix}, \quad \mathbf{c} = \begin{pmatrix} 4 \\ 7 \\ 9 \\ 5 \\ 5 \\ 4 \end{pmatrix},$$

e

$$A = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 4 \\ 10 \\ 6 \\ 12 \\ 8 \end{pmatrix}.$$

Seu dual é $\max \mathbf{b}^T\mathbf{w}, A^T\mathbf{w} \leq \mathbf{c}$. Note que

$$A^T = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Ou seja, o dual é

$$\begin{aligned} \max\, & 4u_1 + 10u_2 + 6u_3 + 12v_1 + 8v_2 \\ \text{s.a.:}\, & u_1 + v_1 \leq 4 \\ & u_1 + v_2 \leq 7 \\ & u_2 + v_1 \leq 9 \\ & u_2 + v_2 \leq 5 \\ & u_3 + v_1 \leq 5 \\ & u_3 + v_2 \leq 4 \\ & u_i, v_i \in \mathbb{R}. \end{aligned}$$

O Teorema das folgas complementares nos garante que uma solução $\mathbf{x}$ para o primal e a solução $(\mathbf{u}, \mathbf{v})$ correspondente para o dual são ótimas se e somente se

$$\begin{aligned} \mathbf{y}^T(A\mathbf{x} - \mathbf{b}) = \mathbf{0} \Rightarrow & \quad t_{ij} - u_i - v_j \geq 0 \\ \mathbf{x}^T(A^T\mathbf{y} - \mathbf{c}) = \mathbf{0} \Rightarrow & \quad x_{ij}(t_{ij} - u_i - v_j) = 0 \end{aligned}$$

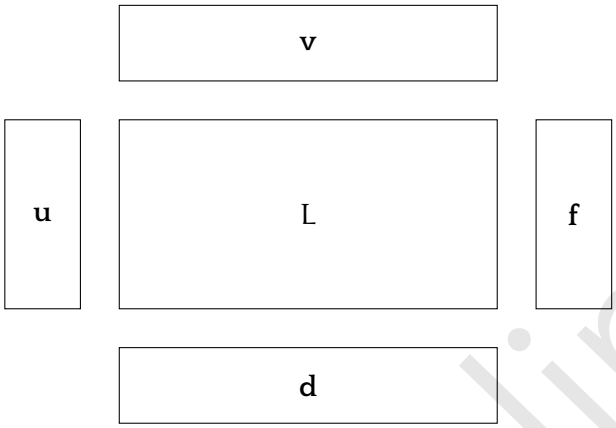
O coeficiente reduzido de custo da variável primal x_{ij} é $r_{ij} = t_{ij} - u_i - v_j$. As restrições do dual são as condições de *otimalidade*, e elas nos dizem exatamente que os coeficientes reduzidos de custo do primal devem ser positivos (o primal é um problema de minimização). Assim, para que a solução seja ótima, se x_{ij} está na base, com $x_{ij} > 0$, então seu coeficiente reduzido de custo é necessariamente zero ($t_{ij} - u_i - v_j = 0$). Se tivermos uma solução viável básica com $x_{ij} > 0$, ela será ótima se

$$t_{ij} - u_i - v_j = 0 \quad i,j \text{ na base.}$$

Como temos $m + n$ variáveis x_{ij}, esta última expressão representa um sistema com $m + n - 1$ equações e $m + n$ variáveis.

Será útil representar em um tableau tanto o problema primal como o dual. O diagrama abaixo mostra o tableau que usaremos: acima e à esquerda, manteremos os valores dos u_i e v_j, que modificaremos em cada

iteração do algoritmo; abaixo e à direita, manteremos os valores de f_i e d_j, para referência. No meio, teremos L, uma matriz onde a alocação dos x_{ij} será feita.



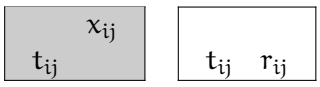
Em cada célula (i, j) da matriz representaremos:

- Se x_{ij} é *básica*: x_{ij}, t_{ij} (para estas $r_{ij} = 0$, portanto não os incluímos);
- Se x_{ij} é *não básica*: r_{ij}, t_{ij} (para estas $x_{ij} = 0$),

onde

$$r_{i,j} = t_{i,j} - u_u - v_j.$$

Representaremos as variáveis básicas com fundo cinza na matriz. A figura a seguir ilustra as duas células.



Exemplo 10.8. Considere o problema de transporte definido por

$$T = \begin{pmatrix} 5 & 4 \\ 3 & 5 \\ 4 & 3 \end{pmatrix}, \quad f = \begin{pmatrix} 6 \\ 8 \\ 5 \end{pmatrix}, \quad d = \begin{pmatrix} 4 \\ 15 \end{pmatrix}.$$

Montamos o tableau. À esquerda, temos apenas a representação esque-

maticame; à direita, temos já os t_{ij}, x_{ij}, f_i, d_j preenchidos:

	v_1	v_2	
u_1	x_{11} t_{11}	x_{12} t_{12}	f_1
u_2	t_{21} r_{21}	x_{22} t_{22}	f_2
u_3	t_{31} r_{31}	x_{32} t_{32}	f_3
	d_1	d_2	

	v_1	v_2	
u_1	4 5	2 4	6
u_2	3 r_{21}	8 5	8
u_3	4 r_{31}	5 3	5
	4	15	

Temos quatro variáveis básicas, e cinco não-básicas.

$$\begin{aligned} t_{11} - u_1 - v_1 &= 0 \\ t_{12} - u_1 - v_2 &= 0 \\ t_{22} - u_2 - v_2 &= 0 \\ t_{32} - u_3 - v_2 &= 0 \end{aligned}$$

Se determinarmos $u_1 = 0$, teremos uma solução para o sistema:

$$\begin{aligned} u_1 = 0 \;\; u_2 = -9 \;\; u_3 = -1 \\ v_1 = 5 \;\; v_2 = 4 \end{aligned}$$

Chegamos ao tableau parcial a seguir. Temos agora os valores de u_i e v_j.

	5	4	
0	5 4	2 4	6
-9	3 r_{21}	8 5	8
-1	4 r_{31}	5 3	5
	4	15	

Agora calculamos os coeficientes reduzidos de custo:

$$\begin{aligned} r_{21} &= t_{21} - u_2 - v_1 \\ r_{31} &= t_{31} - u_3 - v_1 \end{aligned}$$

Com isto temos o tableau completo:

	5	4	
0	5 4	2 4	6
−9	3 7	8 5	8
−1	4 0	5 3	5
	4	15	

Por acaso observamos que como todos os coeficientes reduzidos de custo são não-negativos,

$$r_{21} = 7$$
$$r_{31} = 0$$

o tableau é ótimo. ◀

Depois de obter uma solução viável básica para um problema de transporte, executamos o seguinte método:

1. Calculamos u_i, v_j, para cada variável x_{ij} básica.

2. Calculamos os coeficientes reduzidos de custo para as variáveis não básicas Sabemos que $r_{i,j} = t_{i,j} - u_u - v_j$, e já calculamos u_i, v_j para todos i, j no passo anterior.

3. Se houver algum $r_{ij} < 0$, a solução pode ser melhorada (lembramos que o problema é de minimização). Senão, a solução é ótima;

4. Após melhorar a solução, recomeçamos.

Suponha que o coeficiente reduzido de custo negativo seja o da variável x_{pq}. Aumentamos x_{pq} em uma quantidade Θ. Com isso a solução torna-se inviável, já que as somas de linhas e colunas devem ser iguais aos f_i e d_j. Alguma variável na mesma linha ou na mesma coluna que x_{pq} deve ser reduzida em Θ. Retiramos portanto Θ de uma variável na mesma linha que x_{pq}. Agora haverá uma outra coluna onde falta Θ. Modificamos esta coluna, e assim sucessivamente até que tenhamos chegado novamente à linha p, formando um ciclo.

```
encontre uma solução viável básica x
repita:
    determine u_i, v_j p/todos i,j na base
        (fixe u_i = 0, resolva t_ij − u_i − v_j = 0)
    calcule r_ij = t_ij − u_i − v_j para i,j fora da base
    se há i,j tais que r_ij < 0
        seja (p,q) a posição com menor r_ij
        aumente x_pq em Θ
        enquanto solução não é viável
            corrija removendo ou adicionando
            Θ a outra variável básica
    senao PARE -- solução ótima
```

Exemplo 10.9.

$$T = \begin{pmatrix} 15 & 18 & 10 \\ 10 & 12 & 16 \\ 13 & 15 & 17 \end{pmatrix}, \quad f = \begin{pmatrix} 5 \\ 15 \\ 20 \end{pmatrix}, \quad d = \begin{pmatrix} 10 \\ 12 \\ 18 \end{pmatrix}.$$

Começamos montando o tableau e aplicando a regra do canto noroeste, depois calculando u_i, u_j e r_{ij}.

	15	17	19	
0	5 15	18 1	10 −9	5
−5	5 10	10 12	16 0	15
−2	13 0	2 15	18 17	20
	10	12	18	

O coeficiente -9 é o único r_{ij} negativo, portanto é o que escolhemos. Somaremos Θ a x_{13}; com isso teremos que remover Θ de x_{11}; consequentemente, somaremos Θ a x_{21}; removemos de x_{22}; somamos a x_{32}; e final-

mente, removemos de x_{33}.

	15	17	19	
0	$5-\Theta$ 15	18 1	$+\Theta$ 10 -9	5
-5	$5+\Theta$ 10	$10-\Theta$ 12	16 0	15
-2	13 0	$2+\Theta$ 15	$18-\Theta$ 17	20
	10	12	18	

O maior valor que mantém a viabilidade é claramente $\Theta = 5$:

	10	12	14	
0	15 5	18 6	5 10	5
0	10 10	5 12	16 2	15
3	13 0	7 15	13 17	20
	10	12	18	

Como todos os coeficientes reduzidos de custo agora são positivos, o tableau é ótimo, e a solução é

$$\begin{array}{ll} x_{13} = 5 & x_{32} = 7 \\ x_{21} = 10 & x_{33} = 13 \\ x_{22} = 5 & \end{array}$$ ◀

10.4 Bases degeneradas

É possível, durante a execução do algoritmo para problemas de transporte, encontrar bases degeneradas. Se isto acontecer, podemos usar o método da perturbação: quando uma variável com valor zero entra na base, modificamos seu valor para algum $\varepsilon > 0$, suficientemente pequeno. Quando a variável sair da base, mudamos novamente seu valor para zero.

Exemplo 10.10. ◀

10.5 O problema de atribuição

Suponha que há n pessoas disponíveis para realizar n trabalhos. Cada pessoa se adequa de maneira diferente a trabalhos diferentes, portanto há um

custo t_{ij} para atribuir a j-ésima tarefa à i-ésima pessoa. Queremos designar cada pessoa para um trabalho, minimizando o custo total. Este é o *problema da atribuição* (ou da *designação* – ou ainda, problema de *emparelhamento de custo mínimo*) que pode ser modelado como programa linear da seguinte maneira.

$$\begin{aligned} \min & \sum_{i,j} t_{ij} x_{ij} \\ \text{s.a.:} & \sum_{j}^{n} x_{ij} = 1 \quad i \leq n \\ & \sum_{i}^{n} x_{ij} = 1 \quad j \leq n \\ & x_{ij} \geq 0 \end{aligned}$$

O problema da atribuição é claramente um caso especial de problema de transporte.

Teorema 10.11. *Seja* $\mathbf{x}$ *uma solução viável básica para um problema de atribuição. Então as variáveis básicas em* $\mathbf{x}$ *tem valor zero ou um.*

Demonstração. A solução é inteira por se tratar de um problema de transporte; e é composta de zeros e uns porque o lado direito das restrições é um e os x_{ij} não podem ser negativos. ■

O algoritmo para problemas de transporte descrito anteriormente neste Capítulo não é a melhor opção para problemas de atribuição, porque neste tipo de problema as soluções são naturalmente degeneradas (cada x_{ij} é igual a zero ou um). Há um algoritmo melhor, chamado de "método Húngaro".

Teorema 10.12. *Se uma constante for somada a uma das linhas ou colunas da matriz de custos de um problema de designação, a solução ótima permanece a mesma.*

Demonstração. Demonstramos o fato para linhas, sendo que para colunas o desenvolvimento é semelhante.

Seja T a matriz de custos do problema original com valor ótimo z. Somamos α a uma linha de T, obtendo a matriz T'. Ao calcularmos o valor

ótimo, a única mudança será na linha i, onde teremos

$$\begin{aligned}(t_{i1}+\alpha)x_{i1}+(t_{i2}+\alpha)x_{i2}+\ldots &= t_{i1}x_{i1}+\alpha x_{i1}+t_{i1}x_{i2}+\alpha x_{i2}+\ldots\\ &= t_{i1}x_{i1}+t_{i1}x_{i2}+\ldots+\alpha\sum_j x_{ij}\\ &= z+\alpha\sum_j x_{ij}\\ &= z+\alpha.\end{aligned}$$

O último passo segue de $\sum_j x_{ij}=1$.

Como o valor objetivo é apenas aumentado por uma constante para qualquer solução, a solução ótima permanece a mesma. ■

O Exercício 85 pede a demonstração da Proposição 10.13.

Proposição 10.13. *Suponha que em um problema de atribuição alguns custos* t_{ij} *são zero. Uma solução* $\mathbf{x}$ *tal que*

$$\begin{cases} x_{ij}=0 & \text{se } t_{ij}>0\\ x_{ij}\in\{0,1\} & \text{se } t_{ij}=0\end{cases}$$

é ótima.

O algoritmo Húngaro tenta modificar a matriz de custos, subtraindo valores de linhas e colunas para tentar obter nela zeros, sem modificar a solução ótima.

Durante a execução do algoritmo marcaremos linhas e colunas com zero, "riscando-as" da matriz. Dizemos que tais linhas ou colunas foram "cobertas" por "traços". Por exemplo, na matriz a seguir cobrimos três zeros com dois traços (um traço cobre a linha dois, e outro traço cobre a coluna tres).

$$\left(\begin{array}{ccc}4 & 8 & 0\\ \hline 0 & 5 & 9\\ \hline 3 & 10 & 0\end{array}\right)$$

Uma das operações usadas no algoritmo usadas precisa de uma explicação. Note que a seguinte operação:

- subtrair Q de toda a matriz;
- adicionar Q às linhas cobertas;
- adicionar Q às colunas cobertas;

é equivalente a esta outra:

- subtrair Q das linhas não cobertas;
- somar Q às interseções de traços.

Esta operação será usada nesta segunda forma pelo algoritmo.

O método Húngaro é listado a seguir em pseudocódigo.

para `toda linha i:`
    $\lambda_i \leftarrow$ `mínimo da linha i`
    $t_{ij} \leftarrow t_{ij} - \lambda_i$
para `toda coluna j:`
    $\kappa_j \leftarrow$ `mínimo da coluna j`
    $t_{ij} \leftarrow t_{ij} - \kappa_j$
repita:
    `cubra os zeros com o menor número R de traços`
    se $R = n$ `PARE`
    `seja Q a menor quantidade não coberta pelos traços`
    `subtraia Q das linhas não cobertas`
    `some Q ás interseções de traços`

Exemplo 10.14. Suponha que queiramos resolver o problema de atribuição com a seguinte matriz de custos:

$$\begin{pmatrix} 21 & 26 & 22 & 28 \\ 14 & 18 & 20 & 17 \\ 18 & 17 & 21 & 25 \\ 21 & 20 & 23 & 24 \end{pmatrix}$$

Seguimos os passos do algoritmo Húngaro.

$$\begin{pmatrix} 21 & 26 & 22 & 28 \\ 14 & 18 & 20 & 17 \\ 18 & 17 & 21 & 25 \\ 21 & 20 & 23 & 24 \end{pmatrix} \begin{matrix} -21 \\ -14 \\ -17 \\ -20 \end{matrix} \rightarrow \begin{pmatrix} 0 & 5 & 1 & 7 \\ 0 & 4 & 6 & 3 \\ 1 & 0 & 4 & 8 \\ 1 & 0 & 3 & 4 \end{pmatrix} \\ -0 \quad -0 \quad -1 \quad -3$$

$$\rightarrow \begin{pmatrix} 0 & 5 & 0 & 4 \\ 0 & 4 & 5 & 0 \\ 1 & 0 & 3 & 5 \\ 1 & 0 & 2 & 1 \end{pmatrix} \qquad \rightarrow \begin{pmatrix} \cancel{0} & \cancel{5} & \cancel{0} & \cancel{4} \\ \cancel{0} & \cancel{4} & \cancel{5} & \cancel{0} \\ 1 & \cancel{0} & 3 & 5 \\ 1 & \cancel{0} & 2 & 1 \end{pmatrix}$$

Conseguimos cobrir os zeros com menos de quatro traços. O menor elemento não coberto é um; subtraímos este valor dos elementos não coobertos, e o adicionamos aos elementos de interseção traços:

$$\begin{pmatrix} 0 & 6 & 0 & 4 \\ 0 & 5 & 5 & 0 \\ 0 & 0 & 2 & 4 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Tentamos novamente cobrir a matriz com traços.

$$\begin{pmatrix} \cancel{0} & \cancel{6} & \cancel{0} & \cancel{4} \\ \cancel{0} & \cancel{5} & \cancel{5} & \cancel{0} \\ \cancel{0} & \cancel{0} & \cancel{2} & \cancel{4} \\ \cancel{0} & \cancel{0} & \cancel{1} & \cancel{0} \end{pmatrix}$$

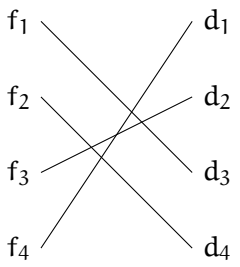
Agora o número mínimo de traços para cobrir a matriz é $n = 4$, e chegamos à solução ótima. Só precisamos determinar quatro zeros, sem que dois estejam em mesma linha ou mesma coluna:

$$\begin{pmatrix} 0 & 6 & \boxed{0} & 4 \\ 0 & 5 & 5 & \boxed{0} \\ 0 & \boxed{0} & 2 & 4 \\ \boxed{0} & 0 & 1 & 0 \end{pmatrix}$$

Os valores da atribuição ótima são um onde temos quatro t_{ij} independentes iguais a zero, portanto,

$$\begin{aligned} t_{41} = 0 &\Rightarrow x_{41} = 1, \\ t_{32} = 0 &\Rightarrow x_{32} = 1, \\ t_{24} = 0 &\Rightarrow x_{24} = 1, \\ t_{13} = 0 &\Rightarrow x_{13} = 1. \end{aligned}$$

A solução é ilustrada a seguir.



O custo desta solução é $21 + 17 + 17 + 22 = 77$. ◀

Notas

O método Húngaro foi desenvolvido por Kuhn [Kuh55] a partir do trabalho dos matemáticos Húngaros Egerváry e König.

O livro de Mokhtar Bazaraa e John Jarvis [BJ77] trata de programação linear relacionada a problemas em redes.

Exercícios

Ex. 73 — Use a regra do canto noroeste para obter soluções viáveis básicas iniciais para os problemas de transporte a seguir.

a) $\mathbf{f} = \begin{pmatrix}100 & 100 & 200\end{pmatrix}$, $\mathbf{d} = \begin{pmatrix}50 & 300 & 50\end{pmatrix}$

b) $\mathbf{f} = \begin{pmatrix}3 & 4 & 5 & 6 & 1\end{pmatrix}$, $\mathbf{d} = \begin{pmatrix}6 & 2 & 2 & 7 & 2\end{pmatrix}$

c) $\mathbf{f} = \begin{pmatrix}10 & 10 & 20 & 20 & 30\end{pmatrix}$, $\mathbf{d} = \begin{pmatrix}15 & 15 & 40 & 4 & 16\end{pmatrix}$

Ex. 74 — Resolva o problema de transporte a seguir usando o algoritmo elaborado neste Capítulo.

$$\mathbf{f} = \begin{pmatrix}10 & 21 & 7 & 27\end{pmatrix} \quad T = \begin{pmatrix}3 & 2 & 4 & 5\\ 4 & 3 & 3 & 2\\ 2 & 5 & 5 & 1\end{pmatrix}$$
$$\mathbf{d} = \begin{pmatrix}10 & 50 & 5\end{pmatrix}$$

Ex. 75 — Aplique o algoritmo de transporte no seguinte problema. Há três origens, com oferta igual a 20, 50 e 10; quatro destinos, com demandas 30, 5, 15 e 35. Os custos são dados abaixo.

	f_1	f_2	f_3
d_1	1	3	2
d_2	3	2	1
d_3	2	1	3
d_4	1	3	2

Ex. 76 — Implemente o algoritmo descrito neste Capítulo para problemas de transporte.

Ex. 77 — Prove que a regra do canto noroeste sempre resulta em uma solução viável básica.

Ex. 78 — Dissemos que é possível contornar o problema de bases degeneradas usando o método da perturbação (transformando zeros em $\varepsilon > 0$ em variáveis básicas). Se, ao invés disso, nada fizermos de especial em relação a soluções degeneradas, haveria a possibilidade do algoritmo entrar em um *loop* infinito ou ele poderia apenas ficar mais lento?

Ex. 79 — Mostre como transformar um problema de transporte com alguns custos negativos em outro com todos os custos positivos.

Ex. 80 — Se todos os valores que definem um problema de transporte forem inteiros, a solução *do dual* também será, em todas as etapas do algoritmo?

Ex. 81 — Mostre como transformar um problema de transporte não balanceado (com $\sum f_i \neq \sum d_j$) em um problema de transporte balanceado equivalente.

Ex. 82 — Considere um problema de transporte com uma restrição adicional: cada fonte não pode mandar toda sua produção para um único destino (a produção de cada fonte deve ser dividida em pelo menos dois destinos). Formule este problema. Tente mostrar como usar o algoritmo elaborado neste Capítulo para problemas de transporte pode ser usado para esta variante do problema, adaptando o algoritmo ou formulando o problema como um problema de transporte comum.

Ex. 83 — Explique brevemente o significado do dual de um problema de transporte: o que são as variáveis e que significam as restrições? E os coeficientes de cada variável no objetivo?

Ex. 84 — Resolva os problemas de atribuição com as matrizes de custos:

$$\begin{pmatrix} 10 & 20 & 30 & 40 \\ 25 & 15 & 25 & 20 \\ 30 & 20 & 15 & 30 \\ 20 & 25 & 30 & 10 \end{pmatrix}, \quad \begin{pmatrix} 40 & 75 & 55 & 85 \\ 15 & 85 & 65 & 60 \\ 105 & 95 & 30 & 100 \\ 45 & 100 & 90 & 90 \end{pmatrix}, \quad \begin{pmatrix} 35 & 39 & 33 & 40 \\ 26 & 16 & 16 & 30 \\ 20 & 10 & 5 & 20 \\ 5 & 15 & 20 & 5 \end{pmatrix}.$$

Ex. 85 — Demonstre a Proposição 10.13.

Capítulo 11

Teoria dos Jogos

O objeto de estudo da Teoria dos Jogos são situações envolvendo conflito de interesse entre diversas partes. Modelamos estas situações como "jogos", onde cada parte é um jogador que tem algumas *estratégias* a escolher. Usamos a suposição de que podemos quantificar o que uma estratégia traz de desejável ou indesejável a cada jogador.

11.1 Jogos de soma zero

Uma categoria importante de jogo é a dos *jogos de soma zero*.

Um jogo é dito *de soma zero* se o pagamento de um jogador sempre implica em perda da mesma quantidade pelos outros jogadores, de tal maneira que ao somarmos o saldo de todos os jogadores ao final do jogo, obteremos zero (presumindo, claro, que o saldo inicial de cada um também era zero). Como exemplos de jogos de soma zero temos diversos jogos recreativos entre duas pessoas, como xadrez, damas, pôquer e outros jogos semelhantes.

Os jogos de soma zero não tem, no entanto, sua relevância limitada a jogos recreativos – diversas situações diferentes podem ser modeladas como jogos de soma zero.

Descrevemos um jogo de soma zero com dois participantes por uma *matriz de pagamentos*: as linhas representam as possíveis estratégias do jogador A e as colunas representam as possíveis estratégias do jogador B. A posição i, j da matriz descreve o quanto o jogador A recebe (e consequentemente quanto B paga) quando A escolhe a estratégia i e B escolhe a estratégia j.

Em uma versão simples de jogo com soma zero, os dois jogadores não

se comunicam, conhecem a matriz de pagamentos, e cada um escolhe sua estratégia sem conhecimento da estratégia do oponente. Após a escolha das estratégias, os jogadores recebem ou pagam, de acordo com o resultado.

Exemplo 11.1. O jogo a seguir nos dá um primeiro exemplo. O jogador A e o jogador B devem escolher entre duas cores – preto e branco – em cada rodada. Quando as cores coincidem, o jogador A ganha. Quando não coincidem, o jogador B ganha. A matriz de pagamentos é mostrada a seguir. Chamamos o jogador A de "jogador linha", porque suas opções estão à esquerda, uma por linha. O jogador B é o "jogador coluna", porque suas escolhas estão acima, uma por coluna.

	preto	branco
preto	$+1$	-1
branco	-1	$+1$

◀

Exemplo 11.2. Damos agora um exemplo mais elaborado. O jogador A tem quatro opções de estratégia, e o jogador b tem três.

	b_1	b_2	b_3	min	
a_1	-2	4	8	-2	
a_2	1	3	4	1	$\leftarrow$
a_3	-3	-6	5	-6	
a_4	-1	3	2	-1	

O menor valor em cada linha é o menor valor que resulta daquela estratégia – e portanto $f(i) = \min_j c_{ij}$ representa o valor mínimo assegurado pela estratégia i. A estratégia a_1 garante retorno de no mínimo -2; a_2 garante retorno mínimo de 1; e assim por diante. Jogando de forma pessimista, o jogador A claramente quer *o maior dos retornos mínimos* – ou seja,

$$\arg\max_i \min_j a_{ij}.$$

A estratégia a_2 é a que garante o maior retorno mínimo para A. O menor valor que o jogador-linha pode garantir para si, então, é $\max_i \min_j a_{ij}$, chamado de *valor maxinmin*. Presumindo que A sempre usará a mesma estratégia, sua melhor opção é escolher sempre a estratégia a_2.

O jogador B realiza a mesma análise, e escolhe sua estratégia, $\min_i \max_j a_{ij}$

– garantindo um retorno mínimo que chamaremos de *valor minimax*:

	b_1	b_2	b_3	min	
a_1	-2	4	8	-2	
a_2	1	3	4	1	$\leftarrow$
a_3	-3	-6	5	-6	
a_4	-1	3	2	-1	
max	1	4	8		
	$\uparrow$				

Presumimos também que B escolherá uma única estratégia, que neste exemplo deve ser b_1, que minimiza sua perda. ◀

11.2 Estratégia mista

Quando o valor maximin é diferente do valor minimax, os dois jogadores devem ter como aproximar seus níveis de segurança – como há um espaço entre os níveis de segurança de cada jogador, podemos imaginar que deveria ser possível a ambos garantir retorno melhor. Isso não é possível, no entanto, usando estratégia fixa.

Se o jogador A usar estratégias diferentes e maneira previsível, o jogador B poderá usar esta informação e obter vantagem sobre A. Idealmente, então, A deve escolher sua estratégia aleatoriamente, mas com uma distribuição de probabilidades determinada – e a distribuição ótima é chamada de *estratégia mista ótima*.

11.2.1 Formulação como programa linear

Seja x_i a probabilidade de uso da estratégia i por A e y_j a probabilidade de uso da estratégia j por B. A esperança de retorno de A é

$$\sum_i \sum_j x_i c_{ij} y_j = \mathbf{y}^T C \mathbf{x}$$

Presumimos que B queira minimizar este valor, escolhendo a estratégia $\mathbf{y}$ tal que

$$\min_{\mathbf{y}} \mathbf{y}^T C \mathbf{x}$$

O jogador A, já supondo que esta será a estratégia de B, pretende escolher a estratégia $\mathbf{x}$ que maximize este valor:

$$\max_{\mathbf{x}} \min_{\mathbf{y}} \mathbf{y}^T C \mathbf{x}$$

Para formular este problema como um programa linear, observamos que a estratégia de A será tal que, *para qualquer estratégia escolhida por* B, A terá retorno de no mínimo v – e então temos, para cada coluna,

$$\sum_j c_{ij} x_i \geq v$$

E maximizamos $\mathbf{x}$. Além disso precisamos da restrição adicional $\sum x_i = 1$, uma vez que $\mathbf{x}$ representa uma distribuição de probabilidades.

Assim, se o jogo é descrito por uma matriz de pagamentos C, a estratégia mista ótima para o jogador linha é dada pela solução do programa linear a seguir.

$$\begin{aligned} &\max v \\ &\text{s.a.} : C^T \mathbf{x} \geq v \\ &\qquad \sum x_i = 1 \\ &\qquad \mathbf{x} \geq 0 \end{aligned}$$

O dual deste programa linear é o problema equivalente para o jogador B (veja o Exercício 89).

Como exemplo, considere o jogo definido pela matriz de pagamentos a seguir.

	1	**2**	**3**	**4**
1	−500	−600	500	0
2	600	500	−700	−400

As estratégia ótima para o jogador linha pode ser obtida a partir do programa linear a seguir.

$$\begin{aligned} &\max v \\ &\text{s.a.} : -500x_1 + 600x_2 \geq v \\ &\qquad -600x_1 + 500x_2 \geq v \\ &\qquad 500x_1 - 700x_2 \geq v \\ &\qquad -400x_2 \geq v \\ &\qquad x_1 + x_2 = 1 \\ &\qquad \mathbf{x} \geq 0 \\ &\qquad v \in \mathbb{R} \end{aligned}$$

Note que a variável v é livre (queremos permitir valores negativos para v). A solução ótima para este programa linear é

$$\mathbf{x} = \begin{pmatrix} 0.6 & 0.4 \end{pmatrix}^T$$

com valor ótimo $v = -160$. Isto significa que a estratégia ótima para o jogador linha é usar a estratégia 1 com probabilidade 0.6 e a estratégia 2 com probabilidade 0.4.

O dual do programa linear define a estratégia ótima para o adversário (jogador coluna), portanto podemos simplesmente usar os coeficientes reduzidos de custo para obter sua estratégia ótima. Neste exemplo, temos

$$\mathbf{y} = \begin{pmatrix} 0 & 0.2666 & 0 & 0.7333 \end{pmatrix}^T$$

e evidentemente, o valor ótimo para o jogador coluna é $+160$.

Teorema 11.3 (Minimax)**.** *Em um jogo de soma zero com dois participantes, sempre existem um valor v e estratégias mistas $\mathbf{x}$, $\mathbf{y}$ que garantem o retorno esperado de v para um dos jogadores e $-v$ para o outro.*

Demonstração. O programa linear que representa o problema de A é viável, porque $\mathbf{x} = (1, 0, \ldots, 0)$ é solução; além disso, é limitado (a restrição $\sum \mathbf{x} = 1$ nos garante isso). Assim, de acordo com o Corolário 5.17 tanto este programa linear como seu dual tem solução ótima.

O programa linear correspondente ao problema de otimização para o jogador A é o dual daquele correspondente ao problema do jogador B. O valor ótimo para os dois programas lineares é o mesmo, e com isso a prova está concluída. ■

Notas

John von Neumann e Oskar Morgenstern publicaram em 1944 o livro "Theory of Games and Economic Behavior" [NM07], onde desenvolvem sistematicamente a Teoria dos Jogos, inclusive o Teorema minimax e a Teoria da Utilidade.

Há diversos outras categorias de jogos além dos que descrevemos neste Capítulo, com duas pessoas e soma zero. Apenas como exemplo, jogos podem ser classificados de acordo com os seguintes critérios, dentre outros.

- o resultado das ações pode ser perfeitamente determinado; probabilístico; ou indeterminado (há um conjunto de possíveis resultados para cada estratégia, mas não se conhece probabilidades sobre eles).

- pode-se ter um só jogador (em um *jogo contra a natureza*) ou diversos jogadores
- há jogos com *informação imperfeita*, onde a informação a respeito dos movimentos anteriores não é completo
- quanto aos pagamentos, além de soma zero há jogos com soma constante e jogos com soma variável.

Uma exposição mais ampla da Teoria dos Jogos pode ser encontrada no livro de Martin Osborne e Ariel Rubinstein [OR94] ou, em uma segunda leitura, no livro de Drew Fudenberg e Jean Tirole [FT91].

Exercícios

Ex. 86 — seria possível caracterizar o jogo de par-ou-ímpar como jogo de soma zero? Se não for possível, o que deveria ser modificado?

Ex. 87 — Sabemos que no jogo-da-velha, após os dois jogadores escolherem suas primeiras jogadas (o primeiro X e o primeiro O), o resultado do jogo já estará determinado, se nenhum deles cometer nenhum erro dali para a frente. Mostre a matriz de pagamentos para este jogo, levando em consideração apenas a primeira escolha de cada um dos jogadores, e presumindo que esta jogada inicial é feita simultaneamente pelos dois jogadores (sem que um saiba a jogada do outro).

Ex. 88 — No jogo de palitinho (ou "porrinha[1]"), dois jogadores tem à sua disposição três palitos para cada um. Cada jogador põe as mãos atrás do corpo e esconde zero, um, dois ou três palitos em uma das mãos, e depois que cada jogador escondeu seus palitos, ambos dão seus palpites a respeito da quantidade total de palitos que existe em suas mãos (que será um número entre zero e seis). Ambos abrem as mãos e o jogador que tiver acertado o palpite ganha. Normalmente um jogador dá o primeiro palpite e o outro dá seu palpite em seguida. Supondo que ambos dessem seus palpites *sem conhecer* o palpite do outro, como seria a matriz de pagamentos do jogo?

Ex. 89 — Verifique que o dual do programa linear que define a estratégia de um jogador em um jogo de soma zero é o programa linear que define a estratégia de seu oponente.

[1]Muito provavelmente derivado do jogo romano "morra", onde os jogadores usam os dedos das mãos ao invés de palitos.

Capítulo 12

Controle Discreto

Este Capítulo trata de problemas de controle em tempo discreto – um problema que pode ser naturalmente modelado como programa linear. Uma vez que o problema também pode ser resolvido usando a técnica de programação dinâmica, esta também é explorada. Em algumas áreas da Matemática Aplicada, controle discreto e programação dinâmica são tratados como um só tópico.

12.1 Programação Dinâmica

Programação Dinâmica é o nome de uma técnica geral de resolução de problemas. Iniciamos com um subproblema pequeno, achamos a solução ótima para esta subproblema e depois a usamos para construir soluções para subproblemas maiores.

Considere a seguinte situação: temos diversos itens, $x_1, x_2, \ldots, x_n$ – cada um com um peso $w_1, \ldots, w_n$ e um valor $v_1, \ldots, v_n$. Temos uma mochila com capacidade C e queremos incluir na mochila itens que maximizem o valor, sem exceder a capacidade da mochila.

O problema da mochila pode ser formulado como um programa linear:

$$\begin{aligned} &\max \mathbf{v}^T\mathbf{x} \\ &\text{s.a} : \mathbf{w}^T\mathbf{x} \leq C \\ &\mathbf{x} \geq \mathbf{0} \end{aligned}$$

Esta formulação só faz sentido, no entanto, se os diversos itens puderem ser fracionados. Se exigirmos que as quantidades sejam inteiras teremos um problema de programação inteira.

Denotamos por $m_{i,k}$ o valor que podemos incluir em uma mochila de capacidade k usando somente os i primeiros itens (e claramente, $m_{i,0} = 0$ para qualquer i).

Se sabemos o valor máximo que conseguimos acomodar na mochila usando apenas os $i-1$ primeiros itens, podemos calcular o valor máximo usando também o i-ésimo item. Se o novo item "cabe" em alguma das soluções "ótimas para capacidade k" anteriores, temos uma nova solução. A seguinte equação recursiva expressa este método.

$$m_{i,k} = \max\{m_{i-1,k}, m_{i-1,k-w_i} + v_i\}$$

Começamos com $m_{i,0} = m_{0,k} = 0$, para todo i e k. Os valores para os $m_{i,j}$ podem ser obtidos por programação dinâmica, e o valor ótimo é $m_{n,C}$. O algoritmo para determinar $m_{n,C}$ é mostrado a seguir.

```
mochila(v, w, C):
    m_{i,0} ← 0
    m_{0,k} ← 0
    para i de 1 a n:
        para k de 1 a C:
            se w_i ≤ k: // item i cabe
                m_{i,k} ← max{m_{i-1,k}, m_{i-1,k-w_i} + v_i}
            senão:
                m_{i,k} ← m_{i-1,k}
    retorne m_{n,C}
```

O algoritmo tem complexidade de tempo $\mathcal{O}(nC)$. O *tamanho* de C é codificado em bits, portanto a complexidade é $\mathcal{O}(2^k)$, onde k é o número de bits usado para representar C.

Exemplo 12.1. Suponha que os pesos, valores e a capacidade sejam $\mathbf{w}$, $\mathbf{v}$ e C dados a seguir.

$$\begin{aligned} \mathbf{w} &= \begin{pmatrix} 3 & 1 & 4 & 2 \end{pmatrix} \\ \mathbf{v} &= \begin{pmatrix} 4 & 3 & 8 & 2 \end{pmatrix} \\ C &= 8 \end{aligned}$$

A tabela com os $m_{i,k}$ mostra o valor ótimo de 15.

	0	1	2	3	4	5	6	7	8
0	0	0	0	0	0	0	0	0	0
1	0	0	0	4	4	4	4	4	4
2	0	3	3	4	7	7	7	7	7
3	0	3	3	4	8	11	11	12	15
4	0	3	3	4	8	11	11	12	15

◀

12.1.1 Aplicabilidade do Algoritmo

A técnica de programação dinâmica é interessante quando o problema a ser resolvido apresenta as duas características a seguir.

- *Subestrutura ótima:* uma solução ótima para o problema pode ser construída a partir de soluções ótimas para subproblemas. No exemplo do problema da mochila, os subproblemas consistiam em maximizar o valor dos itens que poderíamos acomodar em mochilas menores.

- *Subproblemas não completamente isolados:* uma solução para um subproblema pode ser usada mais de uma vez, porque há diferentes subproblemas maiores que a usam. No exemplo da mochila, após calcularmos o valor máximo para uma mochila de tamanho 3, usamos este dado na construção das soluções para mochilas de tamanho 4, 5, etc.

12.2 Processo Markoviano de Decisão

Há problemas de otimização que podem ser formulados como programas lineares, mas que também podem ser resolvidos por programação dinâmica. No restante deste Capítulo analisamos os processos Markovianos de decisão.

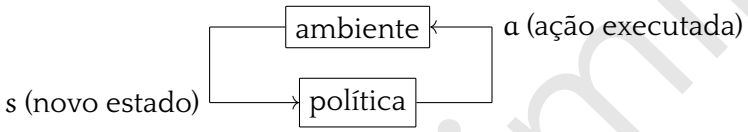
O seguinte problema é frequentemente usado para ilustrar o conceito de processo Markoviano de decisão: um empresário produz um único tipo de produto. O empresário deve, todo mês, verificar seu estoque e decidir quantas unidades adicionais deve encomendar para a fábrica, sabendo, além da quantidade que tem em estoque, a probabilidade de que cada unidade seja vendida, e o custo de armazenamento por unidade. O empresário quer maximizar o lucro nos próximos meses.

O estoque no começo do mês t é s_t, que chamamos de *estado* do sistema. A quantidade de unidades encomendada é a_t. A quantidade de

pedidos é aleatória (com distribuição conhecida), e a denotamos por d_t. Então

$$s_{t+1} = \max\{0, s_t + a_t - d_t\}$$

O empresário poderá pedir mais ou menos unidades, dependendo do estoque. O problema descrito envolve um sistema com estado interno. O estado do sistema muda de forma probabilística a cada intervalo de tempo, dependendo de ações escolhidas – e o que se quer é a redução do custo ao longo do tempo (a minimização de uma função). Esta é uma instância de um problema mais geral, chamado de *processo Markoviano de decisão*.



Definição 12.2 (Processo Markoviano de Decisão)**.** Um processo Markoviano de decisão consiste de um sistema com estado interno, um agente controlador, uma função de transição para o próximo estado e uma função que dá a recompensa para cada estado. Define-se o processo Markoviano de decisão, então, por (S, A, T, R), onde S é o conjunto de possíveis estados do sistema; A é o conjunto de possíveis ações (ou "controles"); $T : S \times A \times S \to [0, 1]$ é a função de transição: $T(s, a, s')$ dá a probabilidade do sistema passar ao estado s' dado que estava no estado s e a ação a foi executada; $R : S \to \mathbb{R}$ é a função recompensa: $R(s)$ é a recompensa por estar no estado s.
Também se deve associar ao problema um *horizonte* h, que consiste no número de decisões a serem tomadas e um *critério de otimalidade*, que determina o que se quer otimizar ao longo das épocas de decisão. ♦

Este processo de decisões em sequência é chamado de *Markoviano* porque a decisão tomada em um dado momento depende apenas do estado do sistema naquele momento, e não das decisões ou estados anteriores – esta é a *propriedade de Markov*.

Ao definirmos um critério de otimalidade, definimos um *valor* para cada estado s em cada época de decisão k, que denotamos $v_k(s)$, de forma que em cada decisão a melhor ação é aquela que maximiza o critério de otimalidade. Por exemplo, podemos decidir maximizar a recompensa média nas épocas de decisão ($1/z \sum_t V_t(s)$) – ou a recompensa total, $\sum_t V_t(s)$, e neste último caso os valores de cada estado serão ótimos quando

$$V_t(s) = \max_{a \in A} \left\{ R(s) + \sum_{s' \in S} T(s, a, s') V_{t+1}(s') \right\}$$

Note que esta equação pode ser usada diretamente na elaboração de um algoritmo recursivo para a solução do problema: parte-se de $v_z(s)$ e calcula-se os valores da época anterior, até chegar à primeira época.

12.3 Horizonte infinito e convergência

Podemos estender estas noções para horizonte infinito. Claramente, não é possível usar o critério de otimalidade da recompensa total, porque o somatório não necessariamente convergiria. Podemos, no entanto, multiplicar a decisão na época k por γ^k, com $0 < \gamma < 1$, garantindo assim que o problema continuará tendo solução. A interpretação deste fator é a seguinte: *valorizamos mais as recompensas obtidas num horizonte mais próximo e menos aquelas obtidas em um horizonte distante*. Este critério de otimalidade é o da *recompensa total esperada*, e nos leva à seguinte versão da equação de Bellman.

$$v(s) = \max_{a \in A} \left\{ R(s) + \gamma \sum_{s' \in S} T(s, a, s') v(s') \right\}.$$

Sua formulação como programa linear é

$$\begin{aligned} &\min \sum_s V(s) \\ &\text{s.a.} : V(s) \geq R(s, a) + T(s, a, s') V(s') \end{aligned}$$

O programa linear tem $|S|$ variáveis e $|S|^2|A|$ restrições.

Da mesma forma que com horizonte finito, podemos extrair da equação de otimalidade um algoritmo recursivo. Partimos de um ponto qualquer e calculamos os valores da época anterior até que os valores entre duas iterações estejam suficientemente próximos.

12.3.1 Convergência

Para demonstrar que o algoritmo de programação dinâmica converge para MDPs de horizonte infinito com $0 < \gamma < 1$, precisamos mostrar que cada passo do algoritmo aplica uma contração na função valor.

O que o algoritmo de programação dinâmica para MDPs faz é essencialmente aproximar uma função, e para garantir que o algoritmo converge mostramos que cada passo dele aplica uma *contração* sobre V.

Definição 12.3 (Sequência de Cauchy)**.** Uma sequência de elementos em um espaço métrico é dita *de Cauchy* quando a distância entre dois elementos a_n, a_{n+1} consecutivos da sequência tende a zero quando $n \to \infty$. ♦

Definição 12.4 (Espaço de Banach)**.** Um espaço vetorial V com uma norma é um espaço de Banach se toda sequência de Cauchy de elementos de V tem limite em V. ♦

Teorema 12.5. *Seja $\|.\|$ a norma obtida tomando o maior valor absoluto dos elementos de um vetor. $\mathbb{R}^n$ com esta norma é um espaço de Banach.*

Definição 12.6 (Contração)**.** Seja V um espaço de Banach. Um operador $F : V \to V$ é uma contração se para todos $u, v \in V$, $\|Fv - Fu\| \leq \alpha\|v - u\|$, com $0 \leq \alpha \leq 1$. ♦

Teorema 12.7. *Seja V um espaço de Banach e $F : V \to V$ uma contração em V. Então F tem um único ponto fixo x^* tal que $Fx^* = x^*$. Além disso, se x_0 é um elemento qualquer de V, e $x_0, x_1, \ldots, x_k$ uma sequência tal que $x_i = Fx_{i-1}$, então esta sequência converge para x^*.*

Observamos agora que podemos reescrever a equação de otimalidade de forma a ficar claro que se trata de um operador linear.

$$\begin{aligned} v_i(s) &= Hv_{i-1}(s) \\ Hv(s) &= \max_{a \in A} \left\{ R(s,a) + \gamma \sum_{s' \in S} T(s,a,s')v(s') \right\}. \end{aligned}$$

Teorema 12.8. *O operador H é uma contração em R_n com a norma $\|v\| = max_i|v_i|$.*

Demonstração. Sejam $u, v \in \mathbb{R}^n$ duas funções valor, com $Hu(s) \geq Hv(s)$. Seja a_s^* uma ação ótima para a função valor v – ou seja,

$$a_s^* \in \arg\max_a \left\{ R(s,a) + \gamma \sum T(s,a,s')v(s') \right\}.$$

Então

$$\begin{aligned} Lv(s) - Lu(s) &\leq R(s, a_s^*) + \sum \gamma T(s, a, s')v(s') \\ &\quad - R(s, a_s^*) + \sum \gamma T(s, a, s')u(s') \\ &= \gamma \sum_{s'} T(s, a, s')(v(s) - u(s)) \\ &\leq \gamma \sum_{s'} T(s, a, s')\|v - u\| \\ &= \gamma\|v - u\| \leq \|v - u\|. \end{aligned}$$ ■

Pode-se definir diferentes critérios de otimalidade, como minimizar a variância da recompensa, sujeitando a solução a uma recompensa média mínima, ou minimizar a recompensa média, limitando a variância da recompensa.

12.4 Formulação como Programa Linear

Embora a equação do princípio da otimalidade já seja ela mesma a expressão de um algoritmo recursivo para a resolução do problema, ele pode também ser modelado como programa linear. Para cada estado, temos

$$V(s) = \max_a \left\{ R(s, a) + \gamma T(s, a, s')V(s') \right\}$$

A equação de Bellman determina o valor ótimo que queremos.

Dado um estado s, qualquer valor acima do que a equação de Bellman determina é uma cota superior para os possíveis valores de $V(s)$. De todos os valores maiores ou iguais a este, tomamos então o menor deles que respeite as restrições $V(s) \geq R(s, a) + T(s, a)V(s')$. Definimos então o problema de minimização

$$\begin{aligned} &\min \sum_s V(s) \\ &\text{s.a.} : V(s) \geq R(s, a) + \gamma \sum_{s'} T(s, a, s')V(s') \quad \forall\, t, s, a \end{aligned}$$

As variáveis são $V(s)$, para todo s. O programa linear tem portanto $|S| \times |A|$ restrições, e $|S|$ variáveis. O dual tem $|S|$ restrições.

Exemplo 12.9. Neste exemplo, para maior concisão, usamos a notação v_s ao invés de $v(s)$.

Suponha que um processo possa estar em dois estados, s_0, s_1, e que possamos controlá-lo usando três ações, a_0, a_1, a_2. Suponha também que a probabilidade de transição entre estados é como na tabela a seguir.

	a_0	a_1	a_2
s_0	$s_0 : 0.2$ $s_1 : 0.8$	$s_0 : 0.5$ $s_1 : 0.5$	$s_0 : 0.4$ $s_1 : 0.6$
s_1	$s_0 : 0.6$ $s_1 : 0.4$	$s_0 : 0.9$ $s_1 : 0.1$	$s_0 : 0.7$ $s_1 : 0.3$

As recompensas para ações realizaas em cada estado são dadas a seguir.

	a_0	a_1	a_2
s_0	20	50	10
s_1	10	10	80

O fator de desconto que queremos usar é $\gamma = 0.9$.

Sabemos que *para cada par estado/ação*

$$v_s \geq R(s,a) + \gamma \sum_{s'} T(s,a,s')v_{s'},$$

portanto as usamos como restrições. A primeira, por exemplo, é

$$\begin{aligned} v_0 &\geq 20 + \gamma(0.2v_0 + 0.8v_1) \\ v_0 &\geq 20 + (0.9)0.2v_0 + (0.9)0.8v_1. \end{aligned}$$

Preferimos manter a multiplicação por γ explicitamente, para manter a clareza de exposição. Na prática, usaríamos "$0.18v_0 + 0.72v_1$" na inequação.

O objetivo será $v_0 + v_1$, e o programa linear é

$$\begin{aligned} \min\ & v_0 + v_1 \\ \text{s.a.:}\ & v_0 \geq 20 + (0.9)0.2v_0 + (0.9)0.8v_1 \\ & v_0 \geq 50 + (0.9)0.5v_0 + (0.9)0.5v_1 \\ & v_0 \geq 10 + (0.9)0.4v_0 + (0.9)0.6v_1 \\ & v_1 \geq 10 + (0.9)0.6v_0 + (0.9)0.4v_1 \\ & v_1 \geq 10 + (0.9)0.9v_0 + (0.9)0.1v_1 \\ & v_1 \geq 80 + (0.9)0.1v_0 + (0.9)0.3v_1 \\ & \mathbf{v} \geq \mathbf{0} \end{aligned}$$

◀

12.5 Variantes de MDPs

Esta seção discute algumas variantes de MDPs, sem no entanto detalhar os métodos para sua resolução. A Seção de Notas ao final do Capítulo traz indicações de leitura adicional para o leitor interessado.

12.5.1 Tempo contínuo

Pode-se definir MPDs em tempo contínuo, mas com estados e ações discretos.

Os conjuntos de estados e ações (S e A) e a função de transição T em um SMDP são semelhantes às de um MDP.

- $R : S \times A \to \mathbb{R}$ é composta de duas outras funções;
- $F : \mathbb{R}_+ \times S \times A \to [0, 1]$ dá a probabilidade da próxima época de decisão acontecer antes do tempo t, sendo que o processo está no estado s e a ação a será executada.

A função de recompensa é composta por duas outras,

- $r : S \times A \to \mathbb{R}$ dá a recompensa imediata após uma decisão a em um estado s.
- $\mathcal{R} : S \times S \times A \to \mathbb{R}$ dá a recompensa por permanecer em um estado s' dado que o estado anterior era s e a ação executada foi a.

Como essa descrição abre a possibilidade de termos infinitas épocas de decisão em tempo finito, presumimos que para todos $s \in S$, $a \in A$, existem $\varepsilon, \delta > 0$ tais que

$$F(s, a, \delta) \leq 1 - \varepsilon$$

O fator de desconto usado para horizonte infinito é $\alpha > 0$, e a recompensa no tempo t é multiplicada por $e^{-\alpha t}$ (o que é equivalente a usar γ^t, com $0 < \gamma < 1$).

A recompensa entre duas épocas de decisão acontecendo nos tempos u_k e u_{k+1} é

$$\int_{u_k}^{u_{k+1}} e^{-\alpha(t-u_k)} \mathcal{R}(s, s_t, a) dt.$$

s_t é o estado no tempo t.

$$V(s) = E\left\{ \sum_{k=0}^{\infty} e^{-\alpha u_k} \left[r(s, a) + \int_{u_k}^{u_{k+1}} e^{-\alpha(t-u_k)} \mathcal{R}(s, s_t, a) dt \right] \right\}$$

A recompensa $R(s,a)$ pode ser calculada a partir de r e F.

$$R(s,a) = r(s,a) + \int_0^\infty \sum_{s'} \left[\int_0^u e^{-\alpha t} \mathcal{R}(s,s',a) T(s,a,s') dt \right] F(du,s,a)$$

$Q(s',t,s,a)$ é a probabilidade da próxima época de decisão acontecer antes do tempo t *e* do próximo estado ser s', dado que o estado atual era s e a ação executada foi a.

$$Q(s',t,s,a) = T(s',s,a)F(t,s,a).$$

Denotamos por $m(s',s,a)$ a probabilidade do próximo estado ser s' dado que o estado atual é s e a ação executada é a.

$$m(s',s,a) = \int_0^\infty e^{-\alpha t} Q(dt,s',s,a)$$

A equação de otimalidade para SMDPs é mostrada a seguir. Com ela podemos usar programação dinâmica na solução destes problemas.

$$V_k(s) = \max_a \left[R(s,a) + \sum_{s'} m(s',s,a) V_{k+1}(s') \right]$$

Diversos outros métodos para solução de MDPs também funcionam para SMDPs, inclusive a formulação como programa linear.

12.5.2 Parâmetros imprecisos

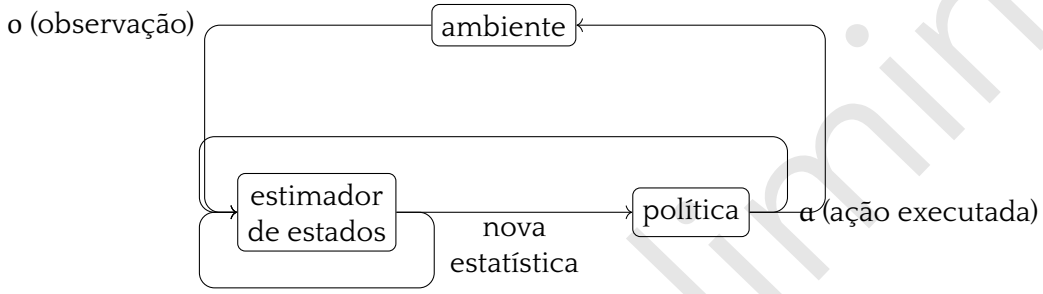
Suponha que alguns dos parâmetros de um MDP foram especificados com imprecisão: sabemos apenas que algumas probabilidades de transição e valores de recompensa estão situados em determinados intervalos. Um MDP especificado desta forma é chamado de "MDP com parâmetros imprecisos" (MDPIP)

12.5.3 Observabilidade parcial

Em um Processo de Decisão de Markov Parcialmente Observável (POMDP[1]), não presumimos que o tomador de decisões conhece o estado do sistema. Ele conhece apenas uma distribuição de probabilidades sobre os estados na primeira época de decisão. Depois disso, a cada época de decisão ele percebe, além de sua recompensa, uma *observação* vinda do ambiente (o

[1] Partially Observable Markov Decision Process

conjunto de observações é finito). Após obter a recompensa e verificar a observação, o tomador de decisões atualiza internamente uma *estatística suficiente* a respeito do sistema. Esta estatística representa, de certa forma, sua crença, e é chamada de *estado de informação*, porque não representa o estado do sistema, e sim o estado da informação que se tem a respeito do sistema.



Um POMDP pode ser descrito da mesma forma que um MDP, pelos conjuntos S, A e funções T, R, além de:

- Ω : o conjunto de possíveis observações.
- $O : S \times A \times \Omega \to [0, 1]$: dá a probabilidade da observação o ser recebida após a ação a ser executada em um estado s.

Há dois exemplos relevantes de estatística suficiente.

- *Histórico completo:* a sequência completa de ações, observações e recompensas desde o primeiro momento é uma estatística suficiente.
- *Distribuição de probabilidades sobre estados:* o tomador de decisões pode, também, manter uma distribuição de probabilidade sobre os estados, e atualizá-la após cada estágio.

A representação de políticas para POMDPs não pode ser feita diretamente, porque seria necessário mapear uma quantidade infinita e não enumerável de distribuições em valores. Pode-se definir, no entanto, um MDP sobre um conjunto infinito de estados de crença, e a função valor torna-se convexa e linear por partes. O número de hiperplanos usado em cada época de decisão) é exponencial em $|\Omega|$, e portanto a quantidade de vetores para z épocas de decisão é duplamente exponencial em z. O algoritmo de programação dinâmica, quando aplicado diretamente a POMDPs,

tem complexidade de tempo $\mathcal{O}(|S||A|^{|\Omega|^z})$. O problema é P-ESPAÇO completo.

Há outras formas de representar políticas para POMDPs, como por exemplo controladores (autômatos de Buchi) estocásticos.

Notas

O problema de Programação Dinâmica foi popularizado por Richard Bellman [Bel03] e Ronald Howard [How60].

O livro de Martin Puterman trata detalhadamente do problema de decisões em sequência [Put05], inclusive a formulação como programa linear. A abordagem moderna para programação dinâmica envolve métodos diferentes da programação linear – Bertsekas aborda o mesmo tema, de forma diferente [Ber07] [BT96]; o livro de Sutton e Barto apresenta métodos biologicamente inspirados [SB98]; o livro de Warren Powell trata de métodos para a resolução de grandes problemas de programação dinâmica [Pow11]. O livro de Robert Stengel é uma boa introdução à Teoria do Controle Ótimo [Ste94], também indicado ao leitor interessado no assunto.

O algoritmo de programação dinâmica, elaborado para resolver um problemas de Controle Ótimo, mostrou-se útil também em outros contextos, tornando-se tópico obrigatório de estudo em Ciência da Computação [Cor+09; DPV06].

Os SMDPs foram estudados inicialmente por W. S. Jewell [Jew63], Howard [How71] e deCani [Can64], e são discutidos nos livros de Martin Puterman [Put05] e de Dimitri Bertsekas [Ber07].

A demonstração de que o problema de resolver POMDPs é P-ESPAÇO-difícil foi dada por Christos Papadimitriou e John Tsitsiklis [PT87].

MDPIPs são discutidos em diversos artigos [SL70; IE94; Fil+07], e há também POMDPs com parâmetros imprecisos, chamados de POMDPIPs [IN07].

Exercícios

Ex. 90 — Implemente o algoritmo de programação dinâmica para o problema da mochila.

Ex. 91 — Implemente o algoritmo de programação dinâmica para MDPs.

Ex. 92 — Se os pesos w_i e a capacidade C em uma instância do problema da mochila são todos muito grandes, é possível reescrever o problema de

forma que exija tempo de execução menor do algoritmo de programação dinâmica?

Ex. 93 — Mostre como resolver as seguintes do problema da mochila usando programação dinâmica ($k \in \mathbb{N}$ é parâmetro de entrada):

i) que funcione permitindo repetição de itens;

ii) que permita até k itens do mesmo tipo;

iii) que permita que até $1/k$ do peso total seja de itens do mesmo tipo;

iv) que permita que até $1/k$ do valor total seja de itens do mesmo tipo.

v) que maximize a quantidade de itens, e não seus valores.

Diga como fica a complexidade de cada um dos seus algoritmos.

Ex. 94 — Tente formular os algoritmos do Exercício 93 como programas lineares.

Ex. 95 — O algoritmo dado neste Capítulo usando programação dinâmica para o problema de mochila apenas determina o *valor* máximo que pode ser acomodado na mochila, e não os itens. Mostre como modificá-lo para que inclua também os itens. Mostre como fica a complexidade do seu algoritmo modificado.

Ex. 96 — Uma fábrica usa uma máquina que se comporta da seguinte maneira: quando a máquina está em perfeito estado, ela produz \$200 por semana. A máquina pode apresentar um problema típico que a fará produzir apenas \$100 por semana. A manutenção preventiva da máquina custa \$10 por semana. A cada semana sem manutenção preventiva, a probabilidade da máquina apresentar esse defeito é 0.2. A cada semana com manutenção preventiva, a probabilidade da máquina apresentar o defeito é 0.1. Se a máquina estiver com esse defeito, pode-se pagar \$300 a um técnico para consertá-la. Se a máquina estiver com defeito, ela também pode quebrar completamente na semana seguinte com probabilidade 0.08. Nesse caso uma nova máquina precisa ser comprada por \$10000. Um gerente da fábrica precisa decidir quando realizar manutenção preventiva e quando consertar máquinas defeituosas.

Ex. 97 — Usando a formalização de SMDPs dada neste Capítulo, descreva MDPs, possivelmente modificando pequena parte daquele formalismo.

Ex. 98 — Descreva claramente a relação entre o algoritmo para o problema da mochila e o princípio da indução matemática (a versão forte).

Ex. 99 — Imagine um processo de decisão de Markov com horizonte *finito* onde a recompensa é definida probabilisticamente em função do tempo: temos $R(s, a, t, h)$ – onde t é o tempo e h é o horizonte (fixo) – ao invés de $R(s, a)$. Você ainda pode determinar a equação de otimalidade? Ela poderia ser usada para modelar este problema como programa linear?

Ex. 100 — Mostre como modelar MDPs de horizonte *finito* como programas lineares.

Parte III

Tópicos Relacionados

Capítulo 13

O Algoritmo Primal-Dual e suas Aplicações

Neste Capítulo descrevemos o algoritmo Primal-Dual para resolução de programas lineares. A importância deste método está não em sua aplicabilidade em programas lineares comuns, mas sim no seu uso para o desenvolvimento de algoritmos de otimização combinatória.

13.1 O algoritmo primal-dual

Definição 13.1 (primal restrito)**.** ♦

13.2

Exercícios

Capítulo 14

Programação Quadrática

Um problema de programação quadrática é semelhante a um problema de programação linear, exceto que a função objetivo é quadrática e não linear (mas as restrições continuam sendo lineares). O objetivo deste Capítulo é meramente o de ilustrar conceitos relacionados a Programação Quadrática e apresentar um algoritmo para solução deste tipo de problema, sem a pretensão de compor um estudo detalhado do assunto.

14.1 Problemas modelados como programas quadráticos

Esta seção ilustra brevemente a utilidade da programação quadrática atrvés de alguns exemplos.

Exemplo 14.1 (regressão por mínimos quadrados)**.** Um primeiro exemplo de problema de programação quadrática é o de regressão por mínimos quadrados. Suponha que tenhamos um conjunto de dados $(\mathbf{x_i}, y_i)$, onde $\mathbf{x_i}$ é um vetor com k elementos. Queremos obter coeficientes a_i para a função

$$f(x_i) = a_0 + a_1 x_{i,1} + a_2 x_{i,2} + \ldots + a_k x_{i,k}$$

de forma a minimizar a soma dos quadrados dos erros

$$\sum_i (y_i - f(\mathbf{x_i}))^2 .$$

Queremos também impor algumas restrições lineares aos coeficientes, e as descrevemos como $R\mathbf{a} = \mathbf{b}$. Este é um problema de programação qua-

drática:

$$\begin{aligned} \min & \sum_i (y_i - a_0 - a_1 x_{i,1} - a_2 x_{i,2} - \ldots - a_k x_{i,k})^2 \\ \text{s.a.} : & R\mathbf{a} = \mathbf{b} \\ & \mathbf{a} \geq \mathbf{0}. \end{aligned}$$

◀

Exemplo 14.2 (maximização de lucro)**.** Em um segundo exemplo, um empresário pretende maximizar o lucro que provém da fabricação de diferentes produtos. O preço p_i de cada produto decai linearmente com a quantidade x_i produzida:

$$p_i = k_i - \gamma_i x_i$$

onde $k_i > 0$ e $\gamma_i > 0$.

Para maximizar o lucro, o empresário resolve o programa quadrático

$$\begin{aligned} \min & \sum_i (k_i - \gamma_i x_i)\, x_i \\ \text{s.a.} : & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

onde A e $\mathbf{b}$ determinam restrições advindas da quantidade de recursos para a fabricação de cada tipo de produto. ◀

Exemplo 14.3 (otimização de portfólio)**.** O terceiro exemplo é o de um investidor procurando construir uma carteira de ativos. O investidor conhece a variância σ_{ii} e as covariâncias σ_{ij} dos ativos, além da esperança de retorno (μ_i) de cada um. Ele quer fixar um retorno mínimo esperado M e minimizar seu risco. Este é um problema de programação quadrática:

$$\begin{aligned} \min & \sum_i \sum_j \sigma_{ij} x_i x_j \\ \text{s.a.} : & \sum_i x_i = 1 \\ & \sum_i \mu_i x_i \geq M \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

◀

14.2 Representação

Na forma mais geral, um programa quadrático é usualmente definido por

$$\begin{aligned} \min\ & z(\lambda, \mathbf{x}) = \lambda \mathbf{c}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T Q \mathbf{x}. \\ \text{s.a.}:\ & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

onde λ é um parâmetro escalar e $\mathbf{x}$ é um vetor com n elementos.

- $\lambda \mathbf{c}^T \mathbf{x}$ descreve a parte linear da função quadrática;
- Q é uma matriz quadrada de ordem n descrevendo uma função quadrática;
- A e $\mathbf{b}$ descrevem as restrições. A é uma matriz $m \times n$, e $\mathbf{b}$ é um vetor com m elementos.

É comum exigir que Q seja semidefinida positiva, para garantir que a função sendo otimizada é convexa.

Exemplo 14.4. A função

$$f(vex) = 2x_1^2 + x_3^2 + x_1x_2 - 4x_2x_3 - x_1 + 3x_2$$

é descrita por $\lambda = 1$ e

$$\mathbf{c} = \begin{pmatrix} -1 \\ 3 \\ 0 \end{pmatrix}, \quad Q = \begin{pmatrix} 4 & 1 & 0 \\ 1 & 0 & -4 \\ 0 & -4 & 2 \end{pmatrix},$$

jé que

$$\mathbf{c}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T Q \mathbf{x} = 2x_1^2 + x_3^2 + x_1x_2 - 4x_2x_3 - x_1 + 3x_2. \qquad \blacktriangleleft$$

Pode-se também adotar uma descrição mais compacta:

$$\begin{aligned} \min\ & z(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q \mathbf{x}. \\ \text{s.a.}:\ & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

Proposição 14.5. *Qualquer função quadrática com $n-1$ variáveis pode ser descrita por uma única matriz simétrica Q de ordem n.*

Demonstração. Basta que a variável x_n seja igual a um.

Mais detalhadamente: sejam

$$Q = \begin{pmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{23} & q_{33} \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix}$$

Então

$$\begin{aligned}
\mathbf{x}^T Q \mathbf{x} &= \begin{pmatrix} 1 & x_1 & x_2 \end{pmatrix} \begin{pmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{23} & q_{33} \end{pmatrix} \begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix} \\
&= \begin{bmatrix} \begin{pmatrix} q_{11} & q_{12} & q_{13} \end{pmatrix} \\ +x_1 \begin{pmatrix} q_{21} & q_{22} & q_{23} \end{pmatrix} \\ +x_2 \begin{pmatrix} q_{31} & q_{32} & q_{33} \end{pmatrix} \end{bmatrix} \begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix} \\
&= \begin{pmatrix} q_{11} + x_1 q_{21} + x_2 q_{31} \\ q_{12} + x_1 q_{21} + x_2 q_{32} \\ q_{13} + x_1 q_{23} + x_2 q_{33} \end{pmatrix}^T \begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix} \\
&= \begin{array}{r} q_{11} + x_1 q_{21} + x_2 q_{31} \\ +x_1(q_{12} + x_1 q_{22} + x_2 q_{32}) \\ +x_2(q_{13} + x_1 q_{23} + x_2 q_{33}) \end{array} \\
&= q_{11} + (q_{21} + q_{12})x_1 + (q_{31} + q_{13})x_2 + (q_{32} + q_{23})x_1 x_2 + q_{22}x_1^2 + q_{33}x_2^2
\end{aligned}$$

Note a presença de termos constantes e de grau um. Descrevemos a forma geral de uma função de grau dois em duas variáveis com uma matriz de ordem tres. ■

Exemplo 14.6. A função dada no exemplo 14.4,

$$f(vex) = 2x_1^2 + x_3^2 + x_1 x_2 - 4x_2 x_3 - x_1 + 3x_2$$

é descrita por

$$Q = \begin{pmatrix} 0 & 0 & 3/2 & 0 \\ -1/2 & 2 & 1/2 & 0 \\ 3/2 & 1/2 & 0 & -2 \\ 0 & 0 & -2 & 2 \end{pmatrix},$$

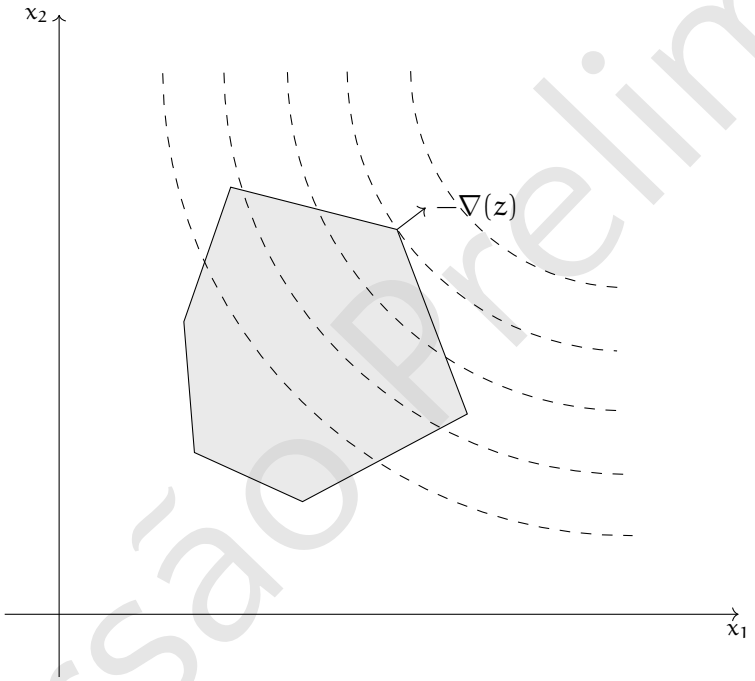
jé que

$$\mathbf{x}^T Q \mathbf{x} = 2x_1^2 + x_3^2 + x_1 x_2 - 4x_2 x_3 - x_1 + 3x_2. \quad ◀$$

14.3 Soluções ótimas para programas quadráticos

Observamos que, como somente a função objetivo é quadrática, a região viável continua sendo um poliedro – da mesma forma que em programação linear. E similarmente, podemos usar o gradiente da função objetivo para determinar a solução ótima.

A próxima figura mostra um exemplo de programa quadrático: o poliedro da região viável tem arestas sólidas, e as curvas de nível da função objetivo $z(\mathbf{x})$ são pontilhadas.



Se a função objetivo de um problema de minimização é convexa e o vértice do parabolóide está dentro da região viável, ele é a solução ótima.

Se a função objetivo de um problema de minimização é côncava (ou seja, $-z(\mathbf{x})$ é convexa), a solução ótima estará na borda do poliedro, mas não necessariamente em um ponto extremo.

Observamos também que a função pode não ser convexa e nem côncava (e neste caso terá pontos de sela).

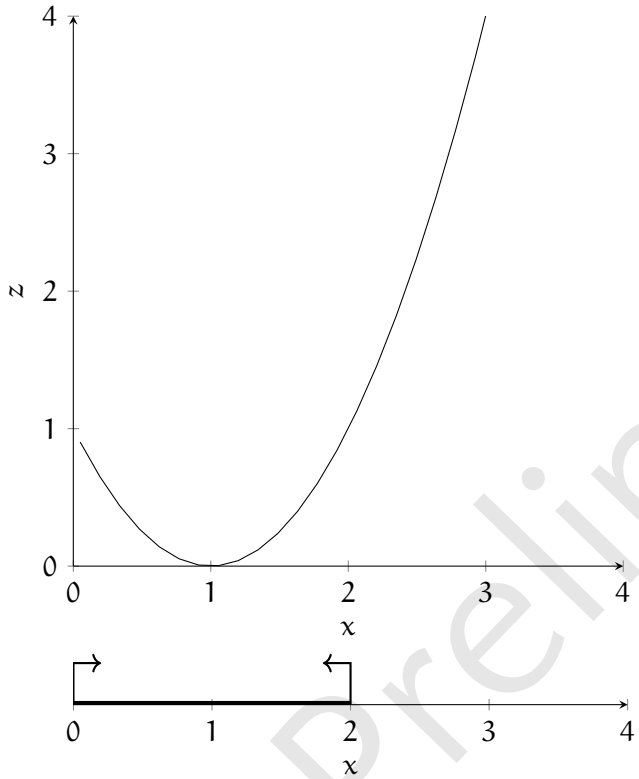
14.4 Método de Beale

As restrições impostas ao problema são lineares (e portanto descritas como $A\mathbf{x} = \mathbf{b}$, como em um proglema de programação linear). Só temos uma função quadrática no objetivo cujas derivadas direcionais podemos calcular, de forma semelhante ao que fizemos no método Simplex. No entanto, *para cada função quadrática de uma variável, o gradiente aponta para direções opostas em lados diferentes do véretice da parábola* (no caso linear, o gradiente do obejtivo é constante – porque é uma função linear – e a direção de maior aumento é fixa. Já no caso quadrático a direção muda, porque o gradiente de uma função quadrática é linear.). Isto significa que seguir cegamente o gradiente pode não levar ao ótimo.

Exemplo 14.7. Para um primeiro exemplo minimalista, temos o seguinte problema.

$$\begin{aligned} \min\ & (x-1)^2 \\ \text{s.a.:}\ & x \leq 2 \\ & x \geq 0 \end{aligned}$$

A figura a seguir mostra a região viável e a função objetivo.



Observe que neste exemplo *o poliedro tem dimensão um*, como mostra parte de baixo da figura: ele é o segmento de reta entre $x = 0$ e $x = 2$. Na parte superior da figura usamos o plano para representar somente a função objetivo.

Os pontos 0 e 2 são os únicos pontos extremos do poliedro, e claramente o ponto ótimo, $x = 1$, não é um deles. ◀

O que ocorre no exemplo 14.7 pode acontecer cada vez que uma variável não básica (e que portanto valia zero) é selecionada para entrar na base.

O método de Beale é uma adaptação simples do algoritmo Simplex que leva em consideração esta situação. Começamos em um vértice da região viável (que é um poliedro, exatamente como em problemas de programação linear). Depois, seguimos por diferentes soluções viáveis básicas. Quando observarmos que uma derivada muda de sinal entre dois pontos extremos, modificamos o problema de forma a garantir que a solução encontrada não estará em nenhum dos dois pontos extremos, mas sim onde aquela derivada é zero.

Há duas diferenças importantes, portanto, entre o método Simplex e o de Beale:

- As derivadas direcionais das variáveis não básicas não são mais dadas por $\mathbf{c} - \mathbf{z}$, porque a função objetivo não é linear. Para a função objetivo $\mathbf{x}^T Q \mathbf{x}$, calculamos a derivada de x_i:

$$\begin{aligned}
&\frac{\partial z}{\partial x_i} \\
&= \frac{\partial}{\partial x_i}\Bigg(q_{22}x_2^2 + q_{33}x_3^2 + \cdots + \underbrace{q_{ii}x_i^2}_{2q_{ii}} + \cdots \\
&\qquad + q_{23}x_2x_3 + q_{24}x_2x_4 + \cdots + q_{i2}x_ix_2 + q_{i3}x_ix_3 + \cdots \\
&\qquad + q_{21}x_2 + \cdots + \underbrace{q_{i1}x_i}_{q_{i1}} + \cdots + \underbrace{q_{1i}x_i}_{q_{1i}} + \cdots \\
&\qquad + q_{11}\Bigg) \\
&= 2q_{ii}x_i + q_{i1} + q_{1i} \\
&= 2q_{ii}x_i + 2q_{i1}.
\end{aligned}$$

  Fica portanto claro agora porque é conveniente usarmos metade desta derivada:

$$\frac{1}{2}\frac{\partial z}{\partial x_i} = q_{ii}x_i + q_{i1}.$$

- Podemos ter que parar entre um ponto extremo e outro, se uma derivada parcial mudar de sinal naquele intervalo. Isso é feito incluindo-se no tableau uma variável e uma restrição adicionais:

$$u_t = \frac{1}{2}\frac{\partial z}{\partial x_i} = q_{ii}x_i + q_{i1}.$$

  nesta situação, para determinar o valor que deveremos atribuir a x_i, igualamos a derivada u_t a zero:

$$\begin{aligned}
u_t = \frac{1}{2}\frac{\partial z}{\partial x_i} &= 0 \\
q_{ii}x_i + q_{i1} &= 0 \\
x_i &= -\frac{q_{i1}}{q_{ii}}.
\end{aligned}$$

Ao escolher a variável a sair da base em otimização linear, tomamos o menor dos valores

$$\frac{b_i}{a_{ij}}$$

Em um problema com função objetivo quadrática, devemos além disso garantir que a derivada da variável x_j não mude de sinal.

Teorema 14.8. *Seja* $\min \mathbf{x}^T C\mathbf{x}$ *s.a.* $A\mathbf{x} \geq \mathbf{b}$, $\mathbf{x} \geq 0$ *um problema de programação quadrática e suponha que uma solução viável básica para o problema seja tal que a variável* x_j *deve entrar na base em um próximo passo do Simplex.*

Para que a nova solução seja viável e que sua derivada não mude de sinal, seu valor deve ser o menor dentre

$$\min_i \left\{ \frac{b_i}{a_{ij}} \right\} \ e \ \frac{q_{i1}}{q_{ii}}.$$

Demonstração. Seja $g(x_j)$ a função quadrática definida quando fixamos todas as variáveis do objetivo exceto x_j. Temos

$$g(x_j) = q_{jj}x_j^2 + q_{j1}x_j + k.$$

O vértice da parábola[1] descrita por g está em

$$x_j = \frac{-q_{j1}}{2q_{jj}},$$

já que os coeficientes de x_j^2 e x_j são q_{jj} e q_{j1}.

Como x_j era não básica, tinha valor zero; como já sabemos que vale a pena incluir x_j na base, a derivada direcional do objetivo em relação a x_j é negativa (aponta para o vértice da parábola). Ao mudarmos continuamente o valor de x_j, uma de duas coisas acontecerá primeiro:

- O valor de x_j chegará a $\min_i \left\{ \frac{b_i}{a_{ij}} \right\}$
- A derivada $\frac{\partial g}{\partial x_j}$ mudará de sinal ao atingirmos o vértice da parábola.

■

1) Comece com uma solução viável básica, da mesma forma que no método Simplex;

[1] O vértice da parábola descrita por $ax^2 + bx + c$ é $(-b/2a, -\Delta/4a)$, onde $\Delta = b^2 - 4ac$ é o discriminante.

2) Escreva as variáveis básicas em função das não básicas (no tableau, isto é o mesmo que usar a matriz identidade como base).

3) Expresse a função objetivo em função das variáveis não básicas.

4) Verifique as derivadas parciais de cada variável não básica x_k.

5) Se nenhuma derivada parcial indica que poderia haver melhora no objetivo, a solução é ótima.

6) Se há alguma derivada parcial que indica que há vantagem em introduzir uma variável não-básica na base, aumentamos a variável não básica, até que uma das duas situações a seguir se concertize:

   a) uma das básicas chega em zero;
   b) a derivada parcial $\partial f/\partial z_k$ muda de sinal.

   Se (a) acontecer, escolha uma coluna para deixar a base, como no método Simplex. Se (b) ocorrer, insira na base a variável

$$u_i = \frac{1}{2}\frac{\partial z_k}{\partial f}$$

7) Repita o processo até que as derivadas parciais das variáveis originais do problema indiquem otimalidade, *e* que as variáveis artificiais introduzidas sejam todas zero.

Exemplo 14.9. Considere o problema

$$\begin{aligned} \min \ & -x_1 - 2x_2 + 2x_2^2 \\ \text{s.a.:} \ & x_1 + x_2 \leq 2 \\ & 2x_1 + x_2 \leq 4 \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Pomos o problema na forma padrão,

$$\begin{aligned} \min \ & -x_1 - 2x_2 + 2x_2^2 \\ \text{s.a.:} \ & x_1 + x_2 + x_3 = 2 \\ & 2x_1 + x_2 + x_4 = 4 \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

. Temos

$$Q = \begin{pmatrix} 0 & -1/2 & -1 \\ -1/2 & 0 & 0 \\ -1 & 0 & 2 \end{pmatrix}$$

Neste problema, podemos começar com a base $x_3 = 2, x_4 = 4$. Temos portanto

$$\begin{aligned} x_3 &= 2 - x_1 - x_2 \\ x_4 &= 4 - 2x_1 - x_2 \\ z &= x_2^2 - 2x_2 - x_1 \end{aligned}$$

As derivadas parciais nas direções das não-básicas são

$$\begin{aligned} \frac{1}{2}\frac{\partial z}{\partial x_1} &= -\frac{1}{2} \\ \frac{1}{2}\frac{\partial z}{\partial x_2} &= -1 + \underbrace{-4x_2}_{\text{zero}} \\ &= -1. \end{aligned}$$

O tableau correspondente é

$$\left(\begin{array}{cccc:c} 1 & 1 & 1 & 1 & 2 \\ 2 & 1 & 0 & 0 & 4 \\ \hline -\frac{1}{2} & -1 & & & \end{array}\right)$$

É interessante portanto incluir x_2 na base. Verificamos agora que

$$\begin{aligned} 2/1 &= 2 \\ 4/2 &= 2 \\ -(-1)/2 &= 1/2 \qquad \text{(vértice da parábola } 2x_2^2 - 2x_2\text{)} \end{aligned}$$

Criamos a nova variável

$$u_1 = \frac{1}{2}\frac{\partial z}{\partial x_2} = -1 - 2x_2$$

e reescrevemos as básicas em função das não básicas:

$$\begin{aligned} x_2 &= \frac{1}{2}u_1 + \frac{1}{2} \\ s_1 &= \frac{3}{2} - x1 - \frac{1}{2}u_1 \\ s_2 &= \frac{7}{2} - 2x1 - \frac{1}{2}u_1 \\ z &= -x_1 - u_1 - 1 + \left(\frac{1}{2}u_1 + \frac{1}{2}\right)^2 \\ &= -\frac{3}{4} - x_1 - \frac{1}{2}u_1 + \frac{1}{4}u_1^2 \end{aligned}$$

O tableau correspondente, já com as derivadas parciais $\frac{\partial z}{\partial x_1}$ e $\frac{\partial z}{\partial u_1}$ é

$$\left(\begin{array}{ccccc|c} 1 & 0 & 1 & 0 & 1/2 & 3/2 \\ 2 & 0 & 0 & 1 & 1/2 & 7/2 \\ 0 & 1 & 0 & 0 & -1/2 & 1/2 \\ \hline -1 & & & & -1/2 & \end{array}\right)$$

Incluimos portanto x_1 na base.

$$\begin{aligned} (3/2)/1 &= 1.5 \\ (7/2)/2 &= 1.75 \end{aligned}$$

A variável s_1 sairá da base. Como a função objetivo é linear em x_1, não há parábola cujo vértice exija nossa atenção. Simplesmente procedemos como no método Simplex, obtendo o tableau a seguir.

$$\left(\begin{array}{ccccc|c} 1 & 0 & 1 & 0 & 1/2 & 3/2 \\ 0 & 0 & -2 & 1 & -1/2 & 7/2 \\ 0 & 1 & 0 & 0 & -1/2 & 1/2 \\ \hline & & +1 & & 0 & \end{array}\right)$$

Como a derivada de s_1 é positiva e a de u_1 é zero, encontramos a solução ótima

$$\begin{aligned} x_1 &= 3/2 \\ x_2 &= 1/2. \end{aligned}$$

A base também inclui a variável de folga s_2, com valor $1/2$. ◀

14.5 Pontos interiores

Métodos de pontos interiores podem ser usados na resolução de problemas de programação quadrática. O método de *affine scaling*, descrito na seção 7.2.1, adapta-se à programação quadrática: o gradiente do objetivo, projetado no espaço nulo de A, muda a cada iteração porque depende do ponto. Além disso, não basta que projetemos no espaço nulo de A, porque se o ótimo for ponto interior, o algoritmo andará de lado a lado da região viável, sem chegar ao ótimo.

Notas

Os primeiros métodos para resolução de programas quadráticos foram desenvolvidos por Wolfe [Wol59], Dantzig [Dan61], e Beale [Bea55]. Métodos modernos para otimização de funções convexas em geral são descritos por Luenberger [Lue10].

O método do elipsoide pode ser usado para programas quadráticos quando a função objetivo é convexa. Quando a função objetivo não é convexa, o problema é NP-difícil [PS91].

Exercícios

Ex. 101 — Converta os problemas do início do Capítulo para a forma $\min \mathbf{x}^T Q\mathbf{x}$, s.a. : $A\mathbf{x} = \mathbf{b}$.

Ex. 102 — Desenvolva um programa que leia uma expressão quadrática e mostre Q tal que $\mathbf{x}^T Q\mathbf{x}$ seja equivalente à expressão dada (este poderia ser um módulo de entrara para um sistema de otimização).

Ex. 103 — Use o método de Beale para obter uma solução ótima para os

seguintes problemas.

$$
\begin{aligned}
&(\mathrm{i})\\
\min\ & -x_1 - 2x_2 + x_2^2\\
\text{s.a.:}\ & x_1 + 2x_2 \leq 4\\
& 3x_1 + 2x_2 \leq 6\\
& \mathbf{x} \geq \mathbf{0}.
\end{aligned}
$$

$$
\begin{aligned}
&(\mathrm{ii})\\
\max\ & 3x_1 + 6x_2 - 4x_1^2 + 4x_1x_2 - x_2^2\\
\text{s.a.:}\ & x_1 + 4x_2 \geq 9\\
& x_1 + x_2 \leq 3\\
& \mathbf{x} \geq \mathbf{0}.
\end{aligned}
$$

Ex. 104 — Implemente o método de Beale

Ex. 105 — O método de Beale pode ser usado com qualquer função objetivo convexa. Mostre como usá-lo com funções diferenciáveis convexas não quadráticas.

Ex. 106 — Implemente o método de *affine scaling* para programação quadrática.

Capítulo 15

Otimização Não Linear

Neste Capítulo abordamos otimização não linear. Além de ser um tópico importante por sua aplicabilidade, também permite compreender melhor os casos mais restritos de programação linear e quadrática.

Um problema de programação não linear consiste em minimizar uma função qualquer $f(x)$, sujeito a restrições quaisquer $g_i(x)$:

$$\begin{aligned} &\min f(\mathbf{x}) \\ &\text{s.a.} : g_i(\mathbf{x}) \geq 0 \quad i = 1, \ldots, k \\ &\qquad h_j(\mathbf{x}) = 0, \quad j = 1, \ldots, l \end{aligned}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$, $g_i : \mathbb{R}^n \to \mathbb{R}$ e $h_j : \mathbb{R}^n \to \mathbb{R}$ são funções *quaisquer*.

Podemos também descrever um problema de otimização não linear apenas com as restrições de desigualdade, já que cada igualdade é equivalente a duas desigualdades.

Definimos a seguir os conceitos de ótimo local e global.

Definição 15.1 (Ótimo local)**.** Seja $f : \mathbb{R}^n \to \mathbb{R}$. O ponto $\mathbf{x}$ é um *mínimo local* de f se $f(\mathbf{x}) \leq f(\mathbf{x}')$ para todo $\mathbf{x}'$ em alguma vizinhança de $\mathbf{x}$. Um *máximo local* é definido de forma análoga. ⧫

Definição 15.2 (Ótimo global)**.** Seja $f : \mathbb{R}^n \to \mathbb{R}$. O ponto $\mathbf{x}$ é o *mínimo global* de f se $f(\mathbf{x}) \leq f(\mathbf{x}')$ para todo $\mathbf{x}' \in \text{Dom}(f)$. O *máximo global* é definido de forma análoga. ⧫

Tratamos neste Capítulo de dois casos diferentes: otimizações sem restrições (quando as restrições não existem) e com restrições. Em ambos os casos, tratamos primeiro da caracterização dos pontos ótimos (as "condições de otimalidade"), e depois dos métodos.

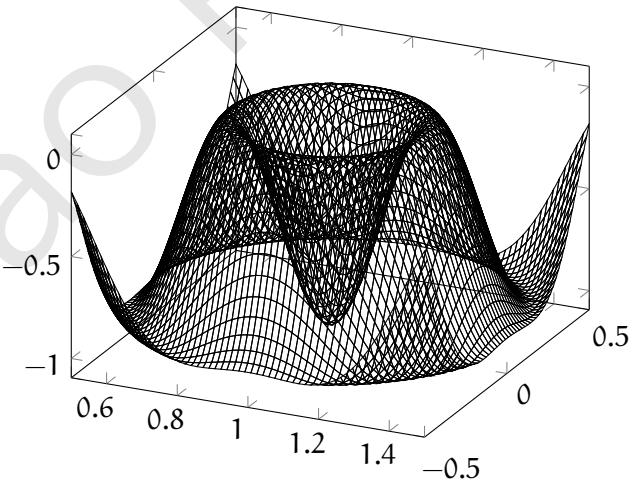
15.1 Otimização sem restrições

Quando as restrições g_i e h_j não existem, temos um problema de otimização sem restrições.

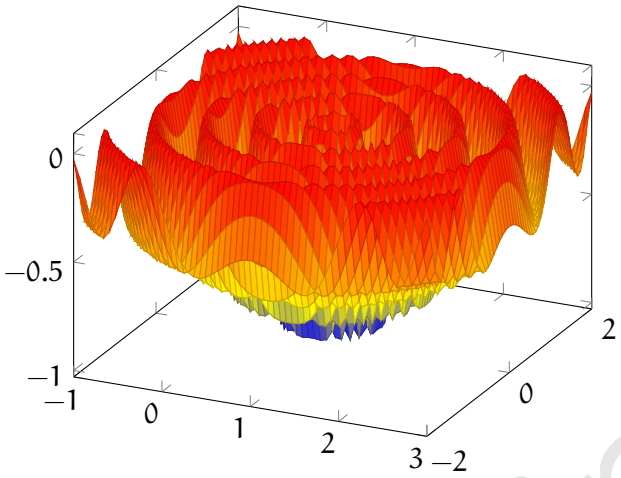
Exemplo 15.3. A função a seguir,

$$f(x,y) = -\frac{1+\cos\left(12\sqrt{(x-1)^2+y^2}\right)}{\frac{1}{2}\left((x-1)^2+y^2\right)+2},$$

é uma variante da função *"drop wave"*[1], usada como teste prático para algoritmos de otimização. Seu mínimo global é $f(1,0) = -1$.



[1]Esta função pode ser transformada na *drop wave* original trocando "$x+1$" por "x". O mínimo global passará a ser no ponto $(0,0)$.



O gráfico da função mostra que pode haver uma grande quantidade de ótimos locais em uma função não convexa – o que é uma grande dificuldade para os métodos de otimização. ◀

15.1.1 Condições de otimalidade

Nesta seção abordamos as condições necessárias e suficientes para que um ponto seja ótimo local. Tratamos separadamente de condições de primeira ordem (que envolve o gradiente do objetivo) e de segunda ordem (onde a matriz Hessiana é usada).

Para demonstrar as condições necessárias de otimalidade usamos o Teorema de Taylor.

Teorema 15.4 (de Taylor)**.** *Seja* $f : \mathbb{R}^n \to \mathbb{R}$ *contínua e diferenciável, e seja* $\mathbf{y} \in \mathbb{R}^n$*. Então existe* $t \in (0, 1)$ *tal que*

$$f(\mathbf{x} + \mathbf{y}) = f(\mathbf{x}) + \nabla f(\mathbf{x} + t\mathbf{y})^T \mathbf{y}.$$

Além disso, se f *é duas vezes diferenciável,*

$$f(\mathbf{x} + \mathbf{y}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T \mathbf{y} + \frac{1}{2} \mathbf{y}^T \nabla^2 f(\mathbf{x} + t\mathbf{y}) \mathbf{y}.$$

Teorema 15.5 (Condição necessária de otimalidade de primeira ordem)**.** *Seja* $f : \mathbb{R}^n \to \mathbb{R}$*. Se* f *é diferenciável em* $\mathbf{x}$ *e* $\mathbf{x}$ *é ótimo (máximo ou mínimo) local de* f*, então* $\nabla f(\mathbf{x}) = \mathbf{0}$*.*

Intuitivamente, o gradiente em $\mathbf{x}^*$ deve ser zero, de outra forma ele indicaria uma direção para onde f teria valores ainda melhores. Este raciocínio é formalizado a seguir.

Demonstração. Suponha que $\mathbf{x}^*$ seja um mínimo local em um problema de minimização com função objetivo $f : \mathbb{R}^n \to \mathbb{R}$. Suponha também, por contradição, que $\nabla f(\mathbf{x}^*) \neq \mathbf{0}$. Então seja

$$\mathbf{d} = -\nabla(\mathbf{x}*).$$

Temos portanto $\mathbf{d}^T \nabla f(\mathbf{x}^*)$ igual a $-\|\nabla f(\mathbf{x})\|^2$, que é menor que zero. Como o gradiente de f é contínuo perto de $\mathbf{x}^*$, deve existir algum $k \in \mathbb{R}$ tal que para todo $t \in (0, k]$,

$$\mathbf{d}^T \nabla f(\mathbf{x}^* + t\mathbf{d}) < 0.$$

No entanto, pelo teorema de Taylor, para qualquer $s \in (0, k]$ existe algum r tal que

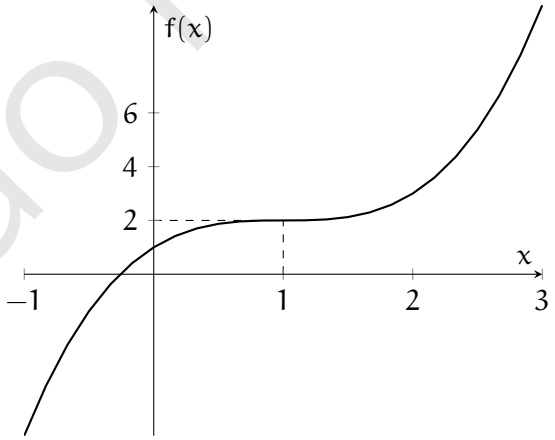
$$f(\mathbf{x}^* + s\mathbf{d}) = f(\mathbf{x}^*) + s\mathbf{d}^T \nabla f(\mathbf{x}^* + r\mathbf{d}),$$

e $f(\mathbf{x}^* + s\mathbf{d}) < f(\mathbf{x}^*)$ para todo $s \in (0, k)$, e portanto há uma direção a partir de $\mathbf{x}^*$ na qual f decresce – mas como presumimos inicialmente que $\mathbf{x}^*$ é mínimo local, concluímos que nossa suposição (de que $\nabla f(\mathbf{x}^*) \neq 0$) é falsa. ■

Para problemas de maximização o raciocínio é semelhante.

A recíproca do Teorema 15.5 não vale: se estivermos minimizando, poderemos encontrar $\nabla f(\mathbf{x}) = \mathbf{0}$ em um máximo local ou ponto de sela.

Exemplo 15.6. A figura a seguir mostra a função $f(x) = x^3 - 3x^2 + 3x + 1$.



No ponto $x = 1$ temos o gradiente igual a zero:

$$\begin{aligned} \nabla f(x) = (f'(x)) &= (3x^2 - 6x + 3) \\ &= (3 - 6 + 3) \qquad \text{(substituindo } x = 1\text{)} \\ &= \mathbf{0}. \end{aligned}$$

No entanto, não se trata de um mínimo local. De fato, como esta função é estritamente crescente[2], ela não tem mínimos (locais ou globais)! ◀

Se a função é convexa, todo mínimo local é mínimo global, e o teste do gradiente, enunciado nas condições de primeira ordem é também suficiente. Em outros casos, precisamos de testes adicionais.

Teorema 15.7 (Condição necessária de otimalidade de segunda ordem)**.** *Seja* $f : \mathbb{R}^n \to \mathbb{R}$*. Então:*

- *Se* $\mathbf{x}^*$ *é mínimo local então* $\nabla^2 f(\mathbf{x}^*)$ *é semidefinida positiva.*
- *Se* $\mathbf{x}^*$ *é máximo local então* $\nabla^2 f(\mathbf{x}^*)$ *é semidefinida negativa.*

Demonstração. Suponha que $\nabla^2 f(\mathbf{x}^*)$ não é semidefinida positiva. Então deve haver algum $\mathbf{d} \in \mathbb{R}^n$ tal que

$$\mathbf{d}^T \nabla^2 f(\mathbf{x}^*)\mathbf{d} < 0.$$

Como $\nabla^2 f$ é contínua perto de $\mathbf{x}^*$, deve haver algum $k \in \mathbb{R}$ tal que para todo $r \in [0, k]$,

$$\mathbf{d}^T \nabla^2 f(\mathbf{x}^* + r\mathbf{d})\mathbf{d} < 0.$$

Fazendo uma expansão de Taylor, para todo $s \in (0, k]$, existe algum r tal que

$$\begin{aligned} f(\mathbf{x}^* + s\mathbf{d}) &= f(\mathbf{x}^*) + s\mathbf{d}^T \nabla f(\mathbf{x}^*) + \frac{1}{2} s^2 \mathbf{d}^T \nabla^2 f(\mathbf{x}^* + r\mathbf{d})\mathbf{d} \\ &= f(\mathbf{x}^*) + \frac{1}{2} s^2 \mathbf{d}^T \nabla^2 f(\mathbf{x}^* + r\mathbf{d})\mathbf{d} \qquad (\nabla f(\mathbf{x}^* = 0), \text{Teorema 15.5}) \\ &< f(\mathbf{x}^*), \end{aligned}$$

e encontramos uma direção a partir de $\mathbf{x}^*$ na qual f decresce. ■

Se f é convexa, a matriz hessiana de f será positiva definida para *todo* o domínio de f.

O Teorema 15.8 dá condições suficientes para que um ponto seja ótimo local. A demonstração é pedida no Exercício 108.

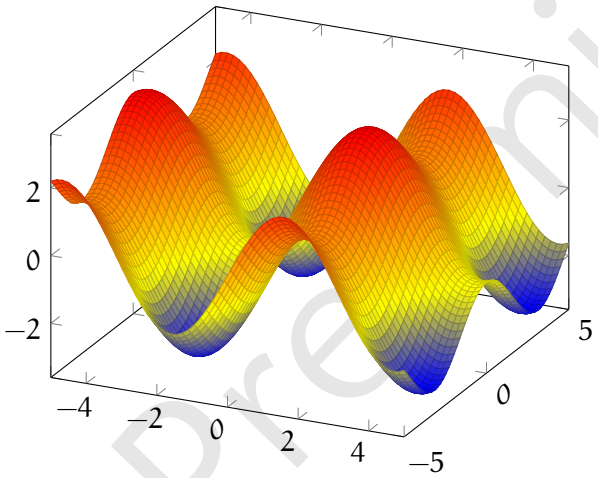
Teorema 15.8 (Condições suficientes de otimalidade de segunda ordem)**.** *Seja* $f : \mathbb{R}^n \to \mathbb{R}$ *uma função contínua e duas vezes diferenciável; suponha que* $\nabla^2 f$ *é contínua em uma vizinhança aberta de* $\mathbf{x}^*$*, e que* $\nabla f(\mathbf{x}^*) = 0$*. Então,*

[2] Verifica-se facilmente: trata-se de $f(x) = (x-1)^3 + 2$, que é x^3 transladada para a direita em 1 e para cima em 2.

- *Se* $\nabla^2 f(\mathbf{x}^*)$ *é definida positiva então* $\mathbf{x}^*$ *é mínimo local.*
- *Se* $\nabla^2 f(\mathbf{x}^*)$ *é definida negativa então* $\mathbf{x}^*$ *é máximo local.*

Note que exigimos que a Hessiana seja *definida* positiva (e não apenas semidefinida).

Exemplo 15.9. Seja $f(x, y) = 2\,\mathrm{sen}(x) + \cos(y)$, cujo gráfico mostramos a seguir.



Temos

$$\nabla f(x, y) = (2\cos x, -\sin(x))^T, \quad \nabla^2 f(x, y) = \begin{pmatrix} -2\,\mathrm{sen}(x) & 0 \\ 0 & -\cos(x) \end{pmatrix}.$$

Queremos determinar se um dos pontos a seguir é mínimo local:

$$\begin{pmatrix} -\frac{\pi}{2} \\ 0 \end{pmatrix}, \quad \begin{pmatrix} -\frac{\pi}{2} \\ \pi \end{pmatrix}.$$

Tomando o primeiro ponto, temos

$$\begin{aligned} \nabla f(-\frac{\pi}{2}, 0) &= (2\cos -\frac{\pi}{2}, -\sin(0))^T \\ &= (2(0) - 0)^T = \mathbf{0}. \end{aligned}$$

No entanto,

$$\nabla^2 f(x, y) = \begin{pmatrix} -2\,\mathrm{sen}(x) & 0 \\ 0 & -\cos(x) \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 0 & -1 \end{pmatrix},$$

portanto não podemos concluir que se trata de um ponto ótimo (de fato, é um ponto de sela!)

Agora, para o segundo ponto,

$$\begin{aligned}\nabla f(-\frac{\pi}{2}, 0) &= (2\cos -\frac{\pi}{2}, -\sin(\pi))^T \\ &= (2(0) - 0)^T = \mathbf{0}.\end{aligned}$$

E a Hessiana é

$$\nabla^2 f(x, y) = \begin{pmatrix} -2\,\mathrm{sen}(x) & 0 \\ 0 & -\cos(x) \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix},$$

e como a Hessiana é *definida* positiva (seus autovalores são estritamente maiores que zero), temos um mínimo local. ◀

15.1.2 Métodos

Há uma grande quantidade de métodos para otimização não linear sem restrições. Apresentamos brevemente aqui algumas categorias de métodos.

Busca linear

Ajuste de curvas

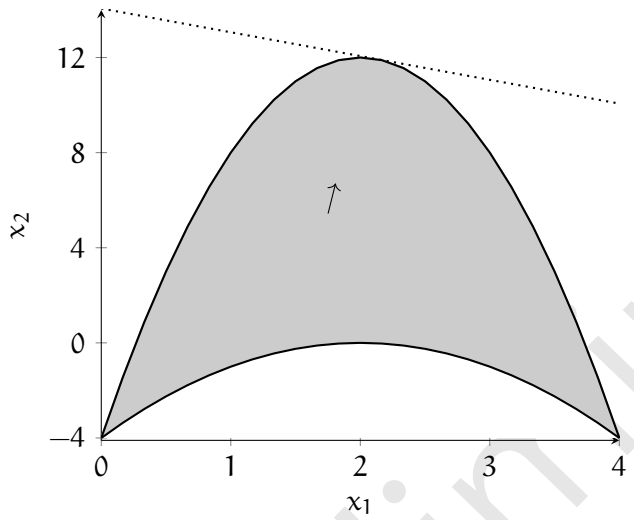
Busca pelo gradiente (descida mais íngreme)

15.2 Otimização com restrições

Exemplo 15.10. Considere o seguinte problema.

$$\begin{aligned}&\max \mathbf{1}^T\mathbf{x} \\ &\text{s.a. : } -x_1^2 + 4x_1 - 4 \leq x_2 \\ &\qquad -4x_1^2 + 16x_1 - 4 \geq x_2 \\ &\qquad \mathbf{x} \in \mathbb{R}^2\end{aligned}$$

A próxima figura mostra a região viável deste problema, o gradiente do objetivo e o hiperplano perpendicular a ele.



A solução ótima para este problema é $\mathbf{x}^* = (2.125, 11.9375)^T$, e o valor da função objetivo neste ponto é 14.0625. ◀

15.2.1 O Lagrangeano

Do Cálculo, relembramos o método dos multiplicadores de Lagrange para otimização.

Dada uma função diferenciável $F(x_1, x_2, \ldots, x_n)$ e restrições $g_i(x_1, x_2, \ldots, x_n) = 0$, se $\mathbf{x}^*$ é um ponto extremo (máximo ou mínimo local) de f, então existem $\lambda_1, \lambda_2, \ldots, \lambda_n$ não negativos tais que

$$\nabla f(\mathbf{x}^*) = \sum_i \lambda_i \nabla_i g_i(\mathbf{x}^*),$$

ou seja, o gradiente de f é combinação linear dos gradientes das g_i com coeficientes λ_i.

Primeiro observamos que podemos escrever

$$\nabla f(\mathbf{x}^*) - \sum_i \lambda_i \nabla_i g_i(\mathbf{x}^*) = \mathbf{0}.$$

Notamos também que se o ponto $\mathbf{x}^*$ satisfaz de maneira estrita uma desigualdade $g_i(\mathbf{x}) \geq 0$ – ou seja,

$$g_i(\mathbf{x}^*) > 0$$

– então podemos ignorá-la, usando $\lambda_i = 0$. Desta forma podemos presumir que, quando o multiplicador $\lambda_i > 0$, a restrição é satisfeita como igualdade, $g_i(\mathbf{x}^*) = 0$.

Disso obtemos a definição do Lagrangeano:

Definição 15.11 (Lagrangeano)**.** Seja $\boldsymbol{\lambda}$ o vetor $(\lambda_1, \ldots, \lambda_m)^T$. O *Lagrangeano* para o problema é a função

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) - \sum \lambda_i g_i(\mathbf{x}). \qquad \blacklozenge$$

15.2.2 Dualidade

Elaboramos agora o conceito de dualidade para problemas de otimização não linear.

Definição 15.12 (Dual de problema de otimização não linear)**.** Para o problema de otimização não linear

$$\begin{aligned} &\min f(\mathbf{x}) \\ &\text{s.a.} : g_i(\mathbf{x}) \geq 0 \quad i = 1, \ldots, k \end{aligned}$$

a *função dual* é

$$L(\lambda) = \inf_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \lambda)$$

e o *problema dual* é

$$\max L(\lambda),$$

sem restrições. $\blacklozenge$

15.2.3 Condições de otimalidade

Nesta seção tratamos das condições de otimalidade de Karush-Kuhn-Tucker. Estas são, basicamente, uma aplicação do método dos multiplicadores de Lagrange para a determinação de ótimos locais.

Enunciamos as condições necessárias de primeira ordem para otimalidade, sem demonstração.

Teorema 15.13 (Condições necessárias de primeira ordem (Karush-Kuhn--Tucker) para otimalidade)**.** *Se a função objetivo e todas as restrições são diferenciáveis em um ótimo local* $\mathbf{x}^*$*, então existe* $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)$ *tal que*

i) ponto estacionário: $\nabla_x \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}) = 0$

ii) viabilidade dual: $\lambda_i \geq 0$

iii) folgas complementares: $\lambda_i g_i(\mathbf{x}^*) = 0$

iv) viabilidade primal: $g_i(\mathbf{x}^*) \leq 0$.

Além das condições necessárias de Karush-Kuhn-Tucker, há também uma condição suficiente, de segunda ordem, para otimalidade.

Teorema 15.14 (Condições suficientes de segunda ordem para otimalidade)**.** *Se as condições de Karush-Kuhn-Tucker são satisfeitas e* $\nabla^2_{xx}\mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda})$ *é positiva definida, então* $\mathbf{x}^*$ *é ótimo local.*

Não conhecemos método para resolver para um problema de otimização não linear analiticamente usando apenas as condições de otimalidade. Estas condições são usadas por algoritmos para direcionar a busca pelo ótimo. No exemplo a seguir apenas verificamos que as condições são verdadeiras para a solução do exemplo 15.10.

Exemplo 15.15. Suponha que tenhamos o ponto $(2.125, 11.9375)$ como candidato a solução do problema descrito no exemplo 15.10. Verificaremos primeiro se este ponto *pode* ser uma solução ótima usando as condições necessárias de primeira ordem. Se forem, verificaremos a condição de segunda ordem.

O Lagrangeano é

$$\begin{aligned}\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) =& f(\mathbf{x}) - \sum \lambda_i g_i(\mathbf{x}) \\ =& (x_1 + x_2) - \left(\lambda_1(-x_1^2 + 4x_1 - x_2 - 4)\right) - \left(\lambda_2(4x_1^2 - 16x_1 + x_2 + 4)\right),\end{aligned}$$

Precisamos então calcular $\nabla_x \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})$:

$$\begin{aligned}\nabla_x \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) =& \nabla_x \left(x_1 + x_2 - \lambda_1(-x_1^2 + 4x_1 - x_2 - 4) - \lambda_2(4x_1^2 - 16x_1 + x_2 + 4)\right) \\ =& \Big(-\lambda_2(8x_1 - 16) - \lambda_1(4 - 2x_1) + 1, \quad \lambda_1 - \lambda_2 + 1 \Big)\end{aligned}$$

As condições KKT são, portanto:

i) $-\lambda_2(8x_1 - 16) - \lambda_1(4 - 2x_1) + 1 = 0$
$\lambda_1 - \lambda_2 + 1 = 0$

ii) $\lambda_1, \lambda_2 \geq 0$

iii) $\lambda_1(-x_1^2 + 4x_1 - x_2 - 4) = 0$
$\lambda_2(4x_1^2 - 16x_1 + x_2 + 4) = 0$

iv) $-x_1^2 + 4x_1 - x_2 - 4 \leq 0$
$4x_1^2 - 16x_1 + x_2 + 4 \leq 0$

Pode-se inferir destas condições que $\lambda_1 = 0$ e $\lambda_2 = 1$.

i) Temos

$$\begin{aligned} -\lambda_2(8x_1 - 16) - \lambda_1(4 - 2x_1) + 1 &= 0 \\ \lambda_1 - \lambda_2 + 1 &= 0 \end{aligned}$$

A segunda condição é trivialmente satisfeita. Verificamos a primeira:

$$-(8x_1 - 16) + 1 = -(17 - 16) + 1 = 0.$$

ii) Trivialmente, os dois λ são ≥ 0.

iii) Como $\lambda_1 = 0$ a primeira equação é satisfeita. Para a segunda, com $\lambda_2 = 1$, temos

$$4x_1^2 - 16x_1 + x_2 + 4 = 0$$

iv) Para a primeira inequação,

$$-x_1^2 + 4x_1 - x_2 - 4 = -11.953125 \leq 0.$$

A segunda já verificamos no item anterior.

Já verificamos as condições necessárias. Agora usamos a condição de segunda ordem, que pode nos garantir que de fato a solução é ótima. O problema é de maximização, por isso verificamos agora que a hessiana $\nabla^2_{xx}\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})$ é negativa semidefinida. A Hessiana é

$$\begin{aligned} \nabla_x \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) &= (-\lambda_2(8x_1 - 16) - \lambda_1(4 - 2x_1) + 1, \lambda_1 - \lambda_2 + 1) \\ \nabla^2_{xx}\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) &= \begin{pmatrix} 2\lambda_1 - 8\lambda_2 & 0 \\ 0 & 0 \end{pmatrix}. \end{aligned}$$

Para qualquer vetor $\mathbf{v} = (v_1, v_2)^T$,

$$\mathbf{v}\left(\nabla^2_{xx}\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})\right)\mathbf{v}^T = \mathbf{v}\begin{pmatrix} 2\lambda_1 - 8\lambda_2 & 0 \\ 0 & 0 \end{pmatrix}\mathbf{v}^T$$

Substituindo os valores de $\boldsymbol{\lambda}$, temos

$$\mathbf{v}\left(\nabla^2_{xx}\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})\right)\mathbf{v}^T = -8v_1^2 \leq 0,$$

o que nos garante que a solução é realmente um ótimo local.

Observe que a Hessiana *não* depende de x_1 ou x_2. ela será semidefinida negativa independente do ponto onde estivermos. ◀

15.2.4 Métodos

15.3 Otimização linear e dualidade

Claramente, um problema de programação linear é um caso particular de otimização não linear.

Notas

A demonstração do Teorema 15.4 (de Taylor) pode ser encontrada em livros de Cálculo de Várias Variáveis [CJ99; Apo69].

Os livros de Luenberger [Lue10], Boyd e Vandenberghe [BV04] e de Nocedal [NW06] são boas introduções ao assunto e contém demonstrações detalhadas das condições de Karush-Kuhn-Tucker. Os livros de Bazaraa [BSS06] e Griva [GNS09] são bons como segunda leitura.

Exercícios

Ex. 107 — Reescreva o primeiro exemplo como problema de minimização e mostre as condições de otimalidade para o problema modificado.

Ex. 108 — Demonstre o Teorema 15.8.

Ex. 109 — Mostre como foram obtidos λ_1 e λ_2 no exemplo 15.15.

Ex. 110 — Considere novamente o exemplo 15.10. Suponha que tenhamos mudado o problema, mantendo a mesma função objetivo e as mesmas restrições, mas agora queremos *minimizar*. Faça uma análise semelhante àquela feita no exemplo 15.15, mostrando a solução ótima, os multiplicadores de Lagrange e o valor da Hessiana.

Ex. 111 — Considere o seguinte problema:

$$\begin{aligned} \min\ & x^2 + y^2 \\ \text{s.a: } & x^2 + y^2 \geq (x + y + 1)\cos(x) + xy \end{aligned}$$

i) Mostre o dual do problema.

ii) Descreva o Lagrangeano e as condições de Karush-Kuhn-Tucker.

iii) Determine se $(0, 0)^T$ é solução ótima.

Ex. 112 — Mostre a forma geral do dual de um programa quadrático.

Parte IV

Apêndices

Apêndice A

Solução Inteira para Problemas de Transporte (demonstração sem unimodularidade

Teorema A.1. *A matriz A de um problema de transporte, da maneira como descrita anteriormente, tem posto* $m + n - 1$.

Demonstração. Como a soma das primeiras m linhas é igual à soma das outras n linhas, claramente o posto não pode ser $m+n$ (podemos escrever uma das linhas como combinação linear das outras).
Mostramos agora que retirando uma das linhas, o conjunto torna-se linearmente independente e que o posto de A deve ser $m + n - 1$. Denotamos as linhas por l_i quando representam uma das m restrições do tipo $\sum_j x_{ij} = a_i$ (que convencionamos posicionar na parte superior da matriz) e por L_j quando representam uma das outras n restrições do tipo $\sum_i x_{ij} = b_j$.

Suponha, sem perda de generalidade, que retiramos a linha L_n da matriz (a última). Agora suponha, por absurdo, que ainda há uma combinação linear das linhas, com nem todos os coeficientes nulos, igual a zero:

$$\alpha_1 l_1 + \alpha_2 l_2 + \ldots + \alpha_m l_m + \beta_1 L_1 + \beta_2 L_2 + \ldots + \beta_{n-1} L_{n-1} = 0$$

Voltamos a atenção para a parte superior da matriz – veja (10.2). Cada x_{in} só aparece na linha l_i (de fato, x_{1n} só aparece na linha l_1, uma vez que não estamos utilizando a linha L_n), e a única maneira de termos a combinação linear igual a zero é ter o coeficiente $\alpha_i = 0$ (de outra forma teríamos $\alpha_i x_{ik}$ no somatório). Como isso vale para todo i, então todos os α_i devem ser zero.

Como os coeficientes α_k são zero, nos interessam os coeficientes β_j. Mas na parte inferior da matriz, cada x_{ij} aparece uma única vez, e portanto precisamos também que todos os β_j sejam zero.

Assim, a única combinação linear que resulta em zero é com $\alpha_i = 0$ e $\beta_j = 0, \forall i, j$ – portanto as $m + n - 1$ linhas são linearmente independentes e o posto da matriz de restrições é $m + n - 1$. ■

Observe que isto significa que durante a execução do método Simplex teríamos também $m + n - 1$ variáveis básicas.

Definição A.2. Uma matriz A é *triangular* se puder ter suas linhas reordenadas de forma que $a_{ij} = 0$ sempre que $i > j$. ♦

Teorema A.3. *A base para um problema de transporte é triangular.*

Demonstração. Considere a tabela de transporte (10.3). Ela mostra claramente a solução viável básica que estivermos trabalhando, portanto podemos usá-la para representar uma base.

Suponha que tenhamos alocado valores em um conjunto de células, obtendo uma solução viável básica.

Mostraremos que cada linha ou coluna tem o valor de *exatamente* uma variável básica: primeiro mostramos que deve haver pelo menos uma variável básica em cada linha/coluna, e depois argumentamos que deve haver ao menos uma linha ou coluna onde não há mais que uma variável básica alocada – e que o resultado deve valer para o sistema que resulta de sua remoção.

Primeiro, notamos que não há linha ou coluna sem variável básica (ou seja, sem célula alocada), porque os a_i e b_j são positivos.

Suponha, por absurdo, que cada linha e cada coluna tenham pelo menos duas variáveis básicas alocadas. O número de variáveis básicas é então

$$\begin{aligned} k \geq & 2m \\ k \geq & 2n. \end{aligned}$$

E portanto $k \geq m + n$ – mas temos somente $m + n - 1$ variáveis básicas (porque este é o posto da matriz de restrições). Assim, existe ao menos uma linha ou coluna onde há somente uma variável básica alocada. Se removermos esta linha ou coluna, obteremos um sistema reduzido e o argumento pode ser repetido.

Como podemos obter os valores de cada variável básica, uma a uma, neste processo, por simples substituição, concluímos que a matriz é triangular. ■

Além do fato da base ser triangular, os coeficientes nela são unitários, e disso concluímos o Teorema a seguir.

Teorema A.4. *Quando todos os* a_i *e* b_j *são inteiros, os valores das variáveis básicas para qualquer base serão também inteiros.*

Exercícios

Ex. 113 — O Teorema A.1 pode ser mostrado extraindo de A uma matriz quadrada de ordem $m + n - 1$ e observando seu determinante. Elabore esta demonstração.

Apêndice B

Respostas e Dicas

Este Apêndice traz respostas e dicas para alguns dos exercícios propostos no texto.

Resp. (Ex. 1) — Primeiro problema, $\mathbf{x} = (0, 6)$, com valor do objetivo -12. Segundo problema, $\mathbf{x} = (5, 5)$, com valor do objetivo igual a 20. Terceiro problema, $\mathbf{x} = (15/2, 5/2)$, com objetivo igual a zero.

Resp. (Ex. 4) — (i) Basta mudar as restrições de conservação de fluxo,

$$\sum_j x_{ji} - \sum_j x_{ij} = 0 \qquad (\forall i)$$

para que levem em conta o fluxo F consumido no nó:

$$\sum_j x_{ji} - \sum_j x_{ij} = F \qquad (\forall i)$$

(Se o nó gera mais fluxo, usa-se valor negativo para F)

Resp. (Ex. 6) —

$$\begin{aligned} &\max 300a + 400p + 500s \\ &\text{s.a} : a + p + s = 100 \\ &\qquad 100p + 200s \leq 14000 \\ &\qquad 100000p + 200000s \leq 12750000 \end{aligned}$$

Resp. (Ex. 7) — (Dica) *Não* é um problema de fluxo em redes, embora a restrição de conservação de fluxo possa ser usada.
Tente representar o mapa como grafo. Se o veículo passará pelo caminho de v_i para v_j, denota $x_{ij} = 1$. Depois, represente os custos como c_{ij}, recompensas como r_{ij}, e formule o objetivo e as restrições. Uma das restrições deve determinar que, para todo i, $\sum_j x_{ji} \leq \sum_j x_{ij}$. Outra, $x_{ij} \leq 1$ para todos i e j.

Resp. (Ex. 12) — (Dica) Prove que cada uma das definições tem a outra como consequência. ($\Rightarrow$) Presuma que a definição a ser usada é a primeira (a que fala de combinações convexas). (i) uma reta é um conjunto convexo (ii) S é convexo (iii) interseção de convexos é convexa (iv) um conjunto de pontos colineares mas não no mesmo segmento não poderia ser convexo - então a interseção é um segmento.
($\Leftarrow$) Presuma que a definição é a segunda (a que fala de interseção ao invés de combinações convexas) (i) para quaisquer pontos a e b em S, considere a reta que passa por a e b. (ii) a interseção com S é um segmento contendo a e b. (iii) ssim, as combinações convexas de a e b (que ficam no caminho entre eles) pertencem a S.

Resp. (Ex. 21) — (i) Não:

$$\nabla^2 f(a, b) = \begin{pmatrix} e^a & \\ & \frac{1}{b^2} \end{pmatrix}$$

e

$$\mathbf{x}^T \nabla^2 f(a, b)\mathbf{x} = e^a x_1^2 - \frac{x_2^2}{b^2}.$$

Com $x_1 = 0$, temos $-\frac{x_2^2}{b^2}$ negativo para todos x_2 e b.
Ou ainda, geometricamente, se fixarmos a temos a função $\log(b)$, que não é convexa.
(ii) $g(a, b) = e^a - \log(b)$ é convexa, porque é soma de duas funções convexas. Ou ainda, porque

$$\mathbf{x}^T \nabla^2 g(a, b)\mathbf{x} = e^a x_1^2 + \frac{x_2^2}{b^2}$$

é sempre positivo.

(iii) Sim: a Hessiana é

$$\begin{pmatrix} e^{a+b} & e^{a+b} \\ e^{a+b} & e^{a+b} \end{pmatrix},$$

com autovalores 0 e $2e^{a+b}$.

Resp. (Ex. 23) — Em (iv), os autovalores são

$$\frac{z \pm \sqrt{z^2 + 4y^2}}{2y^2}.$$

O autovalor relevante é

$$\frac{z - \sqrt{z^2 + 4y^2}}{2y^2},$$

portanto a função seria convexa onde

$$\begin{aligned} z &\geq \sqrt{z^2 + 4y^2} \\ z^2 &\geq z^2 + 4y^2, \end{aligned}$$

ou seja, onde $y = 0$. No entanto, a função original toma $\log(y)$, e não existe, portanto, subdomínio onde a função é convexa.
Em (v), a Hessiana é

$$\begin{pmatrix} e^x + x^{-2} & 0 & 0 \\ 0 & y^{-2} & 0 \\ 0 & 0 & 2 \end{pmatrix},$$

com todos autovalores positivos e portanto convexa em todo domínio.

Resp. (Ex. 27) — Sim. Verifique se o gradiente do objetivo é ortogonal aos hiperplanos definidos pelas restrições.

Resp. (Ex. 29) — (iv) $x_1 = 1/3$, $x_2 = 31/6$; (v) ilimitado; (vi) dois pontos ótimos, $x_3 = 10, x_5 = 10$ e $x_5 = 22$; (vi) $x_1 = 2.5$, $x_2 = 1.5$; (vii) $x_1 = 5.5$, $x_2 = 4.5$.

Resp. (Ex. 30) — (i) $\mathbf{x} = (5, 0, 0)^T$, $z_0 = 10$. (ii) $\mathbf{x} = (2, 2)^T$. (iii) inviável. (iv) ilimitado.

Resp. (Ex. 35) — (Dica) Se x_j não está na base, a proposiçõ vale trivialmente. Se x_j está na base, x_j é componente do vetor $\mathbf{x} = A_B^{-1}\mathbf{b}$, logo é o produto de m elementos de A_B^{-1} por elementos de $\mathbf{b}$. Já B^{-1} é um determinante de ordem $m-1$ dividido por um determinante de ordem m...

Resp. (Ex. 36) — (i) não. (ii) É um hiperplano com dimensão $m-1$. Todo ponto viável é ótimo.

Resp. (Ex. 39) — O problema é degenerado.

Resp. (Ex. 42) — $x_4 = 30, x_6 = 20$. O valor do objetivo é 160.

Resp. (Ex. 45) — Escreva o dual, e observe que $M\mathbf{v} = \mathbf{w}$ significa que $\mathbf{w}$ é combinação linear das colunas de M, com coeficientes em $\mathbf{v}$.

Resp. (Ex. 49) — O Exemplo 5.2 mostra como obter o dual com uma variável a mais. Tente fazer uma troca de variáveis no dual obtido daquela forma.

Resp. (Ex. 50) — Não será única.

Resp. (Ex. 58) — Para garantir uma solução viável básica, faça projeção paralela a uma aresta que liga o ponto extremo atual para um em $\mathbb{R}^{n-1}$.

Resp. (Ex. 66) — O poliedro definido pelo dual também é integral.

Resp. (Ex. 88) — As opções de cada jogador são a quantidade de palitos ($0 \leq p \leq 3$) a esconder *e* seu palpite ($0 \leq a \leq 6$). Uma formulação ingênua usaria $4 \times 7 = 28$ linhas e colunas para ambos os jogadores, mas se notarmos que, sabendo quantos palitos tem em sua própria mão, sobram quatro possibilidades para o palpite ($n+0$, $n+1$, ..., $n+3$), chegamos à

seguinte matriz 16×16:

	0,0	0,1	0,2	0,3	1,1	1,2	1,3	1,4	2,2	2,3	2,4	2,5	3,3	3,4	3,5	3,6
0,0		+	+	+	−				−				−			
0,1	−					+	+	+	−				−			
0,2	−				−					+	+	+	−			
0,3	−				−				−					+	+	+
1,1	+		+	+		−				−				−		
1,2		−			+		+	+		−				−		
1,3		−				−			+		+	+		−		
1,4		−				−				−			+		+	+
2,2	+	+		+			−				−				−	
2,3			−		+	+		+			−				−	
2,4			−				−		+	+		+			−	
2,5			−				−				−		+	+		+
3,3	+	+	+					−				−				−
3,4				−	+	+	+					−				−
3,5				−				−	+	+	+					−
3,6				−				−				−	+	+	+	

Resp. (Ex. 92) — Sim: divida os w_i e C pelo seu mdc (o algoritmo tem complexidade $\mathcal{O}(nC)$, portanto se diminuirmos C diminuímos o tempo de execução).

Resp. (Ex. 97) — Use

$$F(s, a, t) = \begin{cases} 1 & \text{se } t < t' \\ 0 & \text{se } t \leq t' \end{cases}$$

Resp. (Ex. 103) — (i) $x_1 = 14/9, x_2 = 2/3$.

Resp. (Ex. 107) —

$$\begin{aligned} \min \ & -x_1 + x_2 \\ \text{s.a.} : & \ x_1^2 - 4x_1 + 4 \geq x_2 \\ & 4x_1^2 - 16x_1 + 4 \leq x_2 \\ & \mathbf{x} \in \mathbb{R}^2 \end{aligned}$$

Resp. (Ex. 110) — Pode-se inferir, do gráfico da região viável, que o mínimo ocorre no ponto onde as restrições se encontram, no lado esquerdo. Com isso obtém-se $\mathbf{x}^* = (0, -4)^T$, com valor -4. Com algumas contas simples

chega-se em $\boldsymbol{\lambda} = -\frac{1}{12}(5, 17)$. A Hessiana será tal que $\mathbf{v}^T H \mathbf{v}$ sempre igual a $\frac{v_1^2}{2} \geq 0$, portanto semidefinida positiva.

Resp. (Ex. 111) – Seja $\mathbf{v} = (x, y)^T$. Então $\mathcal{L}(\mathbf{v}, \lambda) = x^2 + y^2 - \lambda(x^2 + y^2 - (x + y + 1)\cos(x) - xy)$. (i) $\max \inf x^2 + y^2 - \lambda(x^2 + y^2 - (x + y + 1)\cos(x) - xy), \lambda \geq 0$
(ii) O gradiente do lagrangeano é

$$\nabla_v(\mathbf{v}, \lambda) = \Big(2x - \lambda\left[y + \cos(x) - \sin(x)(y + x + 1)\right], \quad 2y - \lambda\left[\cos(x) + x\right]\Big)$$

Assim, as condições KKT são:

$$\begin{aligned} 2x - \lambda\left[y + \cos(x) - \sin(x)(y + x + 1)\right]\right] &= 0 \\ 2y - \lambda\left[\cos(x) + x\right] &= 0 \\ \lambda(x^2 + y^2 - (x + y + 1)\cos(x) - xy) &= 0 \\ x^2 + y^2 - (x + y + 1)\cos(x) - xy &\geq 0 \\ \lambda &\geq 0 \end{aligned}$$

(iii) Não, porque

$$x^2 + y^2 - (x + y + 1)\cos(x) - xy = -1\cos(0) < 0.$$

Ficha Técnica

Este texto foi produzido inteiramente em LaTeX em um sistema GNU/Linux. Os diagramas foram criados sem editor gráfico, usando diretamente o pacote Ti*k*Z. O ambiente Emacs foi usado para edição do texto LaTeX.

Bibliografia

[ABD09] David Avis, David Bremner e Antoine Deza. *Polyhedral Computation*. AMS, 2009. ISBN: 978-0-8218-4633-9.

[Apo69] Tom M. Apostol. *Calculus, Vol. 2: Multi-Variable Calculus and Linear Algebra with Applications to Differential Equations and Probability*. Wiley, 1969. ISBN: 978-0471000075.

[Bar+98] Cynthia Barnhart et al. "Branch-and-Price: column generation for solving huge integer programs". Em: *Operations Research* 46.3 (1998), pp. 316–329.

[Bar02] Alexander Barvinok. *A Course in Convexity*. AMS, 2002. ISBN: 978-0821829684.

[Bea55] E. M. L Beale. "On Minimizing a Convex Function Subject to Linear Inequalities". Em: *Journal of the Royal Statistics Society*. B 17 (1955), pp. 173–184.

[Bel03] Richard Bellman. *Dynamic Programming*. Dover, 2003. ISBN: 978-0486428093.

[Ber07] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. 2007. ISBN: 978-1886529083.

[Bis14] Johannes Bisschop. *AIIMS: optimization modelling*. 2014. ISBN: 978-1-84753-912-0.

[BJ77] Mokhtar S. Bazaraa e John J. Jarvis. *Linear Programming and Network Flows*. Wiley, 1977. ISBN: 0-471-06015-1.

[BN03] Dimitri Bertsekas e Angelia Nedic. *Convex Analysis and Optimization*. Athena Scientific, 2003. ISBN: 978-1886529458.

[BSS06] Mokhtar S. Bazaraa, Hanif D. Sherali e C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley-Insterscience, 2006. ISBN: 978-0471486008.

[BT96] Dimitri P. Bertsekas e John Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996. ISBN: 978-1886529106.

[BV04] Stephen Boyd e Lieven Vandenberghe. *Convex Optimization*. Cambridge, 2004. ISBN: 978-0521833783.

[Can64] J. S. De Cani. "A dynamic programming algorithm for embedded Markov chains when the planning horizon is at infinity". Em: *Management Science* 10 (1964), p. 716.

[CJ99] Richard Courant e Fritz John. *Introduction to Calculus and Analysis, Vol. II/1*. Springer, 1999. ISBN: 978-3540665694.

[Cor+09] Thomas Cormen et al. *Introduction to Algorithms*. 3ª ed. MIT Press, 2009.

[Dan61] George B. Dantzig. *Quadratic Programming, A Variant of the Wolfe-Markovitz Algorithms*. Rel. téc. OCR 61-2. Operations Research Center, University of California, Berkeley, 1961.

[Dan90] George B. Dantzig. "The Diet Problem". Em: *Interfaces* 20.4 (1990), pp. 43–47.

[DPV06] Sanjoy Dasgupta, Christos Papadimitriou e Umesh Vazirani. *Algorithms*. McGraw-Hill, 2006. ISBN: 0073523402.

[Dym07] Harry Dym. *Linear Algebra in Action*. AMS, 2007. ISBN: 978-0-8218-3813-6.

[Eri03] Bajalinov Erik B. *Linear-Fractional Programming: theory, methods, applications and software*. Springer, 1003. ISBN: 978-1-4613-4822-1.

[Fil+07] Ricardo Shirota Filho et al. "Multilinear and Integer Programming for Markov Decision Processes with Imprecise Probabilities". Em: *Proceedings of the 5th International Symposium on Imprecise Probability: Theories and Applications*. 2007.

[FT91] Drew Fudenberg e Jean Tirole. *Game Theory*. MIT Press, 1991. ISBN: 978-0262061414.

[GL00] Marco Cesar Goldbarg e Henrique Pacca L. Luna. *Otimização Combinatória e Programação Linear: modelos e algoritmos*. Campus, 2000. ISBN: 85-352-0541-1.

[GNS09] Igor Griva, Stephen G. Nash e Ariela Sofer. *Linear and Nonlinear optimization*. SIAM, 2009. ISBN: 978-0898716610.

[Gom58] R. E. Gomory. "Outline of an Algorithm for Integer Solution to Linear Programs". Em: *Bulletin of the American Mathematical Society* 64.5 (1958).

[How60] Ronald A. Howard. *Dynamic Programming and Markov Processes*. MIT Press, 1960.

[How71] R. A. Howard. *Semi-Markovian decision processes*. Proc. Intern. Stat. Inst. (Ottawa, Canada, 1963). 1971.

[IE94] C. C. White III e H. K. El-Deib. "Markov decision processes with imprecise transition probabilities". Em: *Operations Research* 42.4 (1994), pp. 739–749.

[IN07] Hideaki Itoh e Kiyohiko Nakamura. "Partially observable Markov decision processes with imprecise parameters". Em: *Artificial Intelligence* 171.8-9 (2007), pp. 453–490.

[Jew63] W. S. Jewell. "Markov-renewal programming I: Formulation, finite return models; Markov-renewal programming II, infinite return models, example". Em: *Operations Research* 11 (1963), pp. 938–971.

[Kar84] Narendra Karmarkar. "A New Polynomial Time Algorithm for Linear Programming". Em: *Combinatorica* 4.4 (1984), pp. 373–395.

[Kuh55] H. W. Kuhn. "The Hungarian Method for the Assignment Problem". Em: *Naval Research in Logistic Quarterly* 2 (1955), pp. 83–97.

[KV12] Bernhard Korte e Jens Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer, 2012. ISBN: 978-3642244872.

[Lay07] Steven R. Lay. *Convex Sets and Their Applications*. Dover, 2007. ISBN: 978-0486458038.

[LD60] A. H. Land e A. G. Doig. "An automatic method of solving discrete programming problems". Em: *Econometrica* 28.3 (1960), pp. 497–520.

[Lim+06] Dag Mendonça Lima et al. *Tabela Brasileira de Composição de Alimentos – TACO*. 2006.

[Lue10] David G. Luenberger. *Linear and Nonlinear Programming*. Springer, 2010. ISBN: 978-1441945044.

[MG06] Jiri Matousek e Bernd Gärtner. *Understanding and Using Linear Programming*. Springer, 2006. ISBN: 978-3540306979.

[NM07] John von Neumann e Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 2007. ISBN: 978-0691130613.

[NR10] Peter Norvig e Stuart Russel. *Artificial Intelligence: a modern approach*. Pearson, 2010. ISBN: 0-13-604259-7.

[NW06] Jorge Nocedal e Stephen Wright. *Numerical Optimization*. Springer, 2006. ISBN: 978-0387303031.

[OR94] Martin J. Osborne e Ariel Rubinstein. *A Course in Game Theory*. MIT Press, 1994. ISBN: 978-0262650403.

[Pow11] Warren B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley, 2011. ISBN: 978-0470604458.

[PS91] Panos M. Pardalos e Vavasis Stephen A. "Quadratic programming with one negative eigenvalue is NP-hard". Em: *Journal of Global Optimization* 1.1 (1991), pp. 15–22.

[PS98] Christos Papadimitriou e Kenneth Steiglitz. *Combinatorial Optimization*. Dover, 1998. ISBN: 0-486-40258-4.

[PT87] Christos H. Papadimitriou e John N. Tsitsiklis. "The complexity of Markov decision processes". Em: *Mathematics of Operations Research* 12.3 (1987), pp. 441–450.

[Put05] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, 2005. ISBN: 978-0471727828.

[Roc96] Ralph Tyrell Rockafellar. *Convex Analysis*. Princeton University Press, 1996. ISBN: 978-0691015866.

[Rus06] Andrzej Ruszczyski. *Nonlinear Optimization*. Princeton University Press, 2006. ISBN: 978-0-691-11915-1.

[SB98] Richard S. Sutton e Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998. ISBN: 978-0262193986.

[Sch03] Alexander Schrijver. *Combinatorial Optimization*. Springer, 2003. ISBN: 978-3540443896.

[Sch98] Alexander Schrijver. *Theory of Linear and Integer Programming*. Wiley, 1998. ISBN: 978-0471982326.

[Sin06] S. M. Sinha. *Mathematical Programming: theory and methods*. Elsevier, 2006. ISBN: 978-8131203767.

[SL70] J. K. Satia e R. E. Lave. "Markovian decision processes with uncertain transition probabilities". Em: *Operations Research* 21 (1970), pp. 728–740.

[Sta97] I. N. Stancu-Minasian. *Fractional Programming: theory, methods and applications*. Kluwer, 1997. ISBN: 978-94-010-6504-7.

[Ste94] Robert F. Stengel. *Optimal Control and Estimation*. Dover, 1994. ISBN: 978-0486682006.

[Van07] Robert J. Vanderbei. *Linear Programming: foundations and extensions*. Springer, 2007. ISBN: 978-0387743875.

[WN99] Laurence A. Wolsey e George L. Nemhauser. *Integer and Combinatorial Optimization*. Wiley-Insterscience, 1999. ISBN: 978-0471359432.

[Wol59] Philip Wolfe. "The Simplex Method for Quadratic Programming". Em: *Econometrica* 27.3 (1959).

[Wol98] Laurence Wolsey. *Integer Programming*. Wiley-Insterscience, 1998. ISBN: 978-0471283669.

Índice Remissivo