ALGORITMOS DE PROGRAMAÇÃO LINEAR

Programação Linear Concreta

Paulo Feofiloff

Professor do
Departamento de Ciência da Computação do
Instituto de Matemática e Estatística da
Universidade de São Paulo

novembro de 1997
revisto em 27.7.1999
reformatado em 11.9.2005

Prefácio

O problema básico de *programação linear*[1] consiste no seguinte: dada uma matriz A e vetores b e c, encontrar um vetor x tal que

$$x \geq 0\,, \quad Ax = b \quad \text{e} \quad cx \text{ é mínimo}\,.$$

O livro discute este problema, suas variantes e generalizações, e a correspondente teoria da dualidade.

O que. Nosso ponto de partida é o algoritmo de Gauss-Jordan e o algoritmo Simplex. Toda a teoria da programação linear é deduzida desses dois algoritmos. Do ponto de vista do Simplex, o problema básico tem a seguinte forma: transformar uma matriz dada (a matriz resulta da justaposição de A, b e c) em outra equivalente que contenha um certo "padrão" ou "desenho".

Examinaremos também um representante da família de algoritmos polinomiais de programação linear que surgiu em meados da década 1970–1980. O algoritmo que discutiremos — uma variante do célebre algoritmo do elipsóide — não é uma alternativa prática para o Simplex,[2] mas tem profundas implicações teóricas.

O livro não trata dos aspectos mais práticos da programação linear. Assim, por exemplo, o livro não se ocupa das implementações *aproximadas* do Simplex, que representam números racionais em notação ponto flutuante; em particular, o livro não trata das heurísticas que procuram reduzir os erros de arredondamento de tais implementações. O livro também não trata das dificuldades práticas associadas com a manipulação de matrizes muito grandes, nem de algoritmos especiais para matrizes esparsas.[3] Finalmente, o livro não trata de *modelagem*, que é a arte de reduzir certos problemas de otimização a problemas de programação linear. Todos esses tópicos são muito importantes na prática, mas estão além dos objetivos do livro (e da competência do autor).

Como. A atitude do livro é mais matemática e conceitual que tecnológica. Em outra dimensão, a atitude é mais *algébrica* que geométrica. O enfoque é

[1] Neste contexto, o termo *programação* significa *planejamento*. Não se trata de uma referência à programação de computadores.

[2] Outros algoritmos da família, entretanto, competem com o Simplex.

[3] O leitor interessado nesses tópicos deve consultar os livros de Chvátal [Chv83] e Golub e Van Loan [GL96].

algorítmico: toda a teoria é derivada dos algoritmos, particularmente do Simplex.

Os algoritmos são descritos de maneira precisa, em linguagem razoavelmente informal. Para tornar isso possível, é necessário introduzir definições limpas para os conceitos de matriz e vetor e uma notação suficientemente poderosa para descrever partes desses objetos.

O livro procura dizer com precisão *o que* cada algoritmo faz e não apenas *como* faz o que faz. O comportamento dos algoritmos é descrito por meio de *invariantes*, e o seu desempenho de pior caso é analisado.

O universo natural da programação linear é o dos números *racionais*. O livro supõe, portanto, que dispomos de um agente computacional capaz de executar aritmética racional. Uma das versões do Simplex (capítulo 12 manipula em separado os numeradores e denominadores dos números racionais e portanto só usa aritmética *inteira*. Segue daí uma versão do Teorema da Dualidade que especifica delimitações superiores para o número de dígitos das soluções do problema de programação linear.

O livro evita o uso indiscriminado de ferramentas da álgebra linear, porque tais ferramentas são, em geral, mais sofisticadas que as situações concretas que é preciso enfrentar. O livro evita também as "hipóteses simplificadoras" (por exemplo, a hipótese de que nossas matrizes têm posto pleno e a hipótese de que dispomos de uma "solução viável" ao iniciar a execução do Simplex) tão comuns em outros textos sobre o assunto. Tais hipóteses pouco contribuiriam para simplificar a discussão.

Para quem. Este livro é dirigido a qualquer pessoa que queira compreender as interconexões lógicas entre as várias peças desse quebra-cabeças que é a programação linear. Em particular, o texto se destina a estudantes de graduação e pós-graduação em matemática aplicada, computação e engenharia. O livro não tem pré-requisitos formais, mas exige uma certa maturidade matemática.

Versões preliminares do livro foram usadas em vários oferecimentos da disciplina Programação Linear nos cursos de graduação e pós-graduação em Ciência da Computação no Instituto de Matemática e Estatística da Universidade de São Paulo. O subtítulo do livro —*Programação Linear Concreta* —é uma alusão ao *Concrete Mathematics* [GKP94] de Graham, Knuth e Patashnik Para explicar o título, o prefácio daquele livro diz:

> The course title "Concrete Mathematics" was originally intended as an antidote to "Abstract Mathematics" [...]. Abstract mathematics is a wonderful subject, [...] But [...] The goal of generalization had become so fashionable that a generation of mathematicians has become unable to relish beauty in the particular, to enjoy the challenge of solving quantitative problems, or to appreciate the value of technique.

Dados técnicos. A elaboração do livro contou com o apoio dos projetos *Aspectos Estruturais e Algorítmicos de Objetos Combinatórios* (FAPESP 96/04505-2) e *Complexity of Discrete Structures* (ProNEx 107/97). O livro foi escrito em

LaTeX nas instalações do Instituto de Matemática e Estatística da Universidade de São Paulo. Informações atualizadas sobre o texto poderão ser encontradas no endereço `http://www.ime.usp.br/~pf/prog-lin/` da teia *WWW*.

São Paulo, 1999–2005

P. F.

Sumário

Capítulo 1

Vetores e Matrizes

"Vetor? É uma espécie de linha reta com uma flecha na ponta."
"Matriz? Acho que é onde fica a sede da empresa."

Este capítulo faz um resumo de conceitos elementares de álgebra linear e introduz as convenções de notação que usaremos nos capítulos subseqüentes. O conteúdo do capítulo é muito simples, mas algumas das definições e convenções de notação merecem atenção pois são pouco usuais.

1.1 Vetores

Um **vetor** é uma função que leva um conjunto finito arbitrário — o conjunto de **índices** — no conjunto dos números reais (mas não há mal em restringir a atenção aos números racionais). Convém não presumir qualquer relação de ordem sobre o conjunto de índices. Se o conjunto de índices de um vetor x é N, diremos que x está definido **sobre** N.

Se x é um vetor sobre um conjunto N e n é um elemento de N então $x_{[n]}$ denota o **componente** n **de** x, isto é, o valor da função x em n. Se Q é uma parte de N então

$x_{[n]}$

$$x_{[Q]}$$

denota a restrição de x a Q, ou seja, o vetor cujo componente q é $x_{[q]}$ para cada q em Q. Note a distinção entre $x_{[n]}$ e $x_{[\{n\}]}$: o primeiro é um número, enquanto o segundo é um vetor (com um só componente).

Um vetor x sobre N é **nulo** se $x_{[n]} = 0$ para todo n em N. O vetor nulo será denotado por o, qualquer que seja o seu conjunto de índices.

vetor nulo
o

Se x é um vetor e λ é um número então λx é o vetor que se obtém mediante multiplicação de cada componente de x por λ. Analogamente, x/λ é o vetor que se obtém dividindo por λ cada componente de x.

Se x e y são vetores sobre um mesmo conjunto de índices e $x_{[n]} \geq y_{[n]}$ para cada n, dizemos que $x \geq y$. Analogamente, dizemos que

$x \geq y$

$$x > y$$

se $x_{[n]} > y_{[n]}$ para todo n. As relações $\leq$ e $<$ são definidas de modo análogo.

4	3	1	2
13	22	11	14

11 14 22 13

Figura 1.1: Duas representações de um vetor sobre $1, 2, 3, 4$. Na segunda, fica subentendido que os índices são 1, 2, 3 e 4 da esquerda para a direita.

1	11
2	14
4	13
3	22

11
14
22
13

Figura 1.2: Mais duas representações do mesmo vetor. Na segunda, fica subentendido que os índices são 1, 2, 3 e 4 de cima para baixo.

1.2 Matrizes

Uma **matriz** é uma função que leva o produto cartesiano de dois conjuntos finitos no conjunto dos números reais (poderíamos restingir a definição ao conjunto dos racionais). Convém não presumir qualquer relação de ordem sobre os conjuntos de índices.

Se uma matriz A tem domínio $M \times N$, dizemos que M é o conjunto de **índices de linhas** e N é o conjunto de **índices de colunas** de A. Dizemos também que A é uma matriz definida **sobre** $M \times N$.

Se m e n são elementos de M e N respectivamente então $A_{[m,n]}$ denota o **componente** m, n **de** A, ou seja, o valor de A em m, n. Se P e Q são partes de M e N respectivamente então $A[m,n]$

$$A_{[P,Q]}$$

é a restrição de A a $P \times Q$. Usaremos a abreviatura $A_{[P,\,]}$ (leia "A P tudo") para $A_{[P,N]}$ e a abreviatura $A_{[\,,Q]}$ para $A_{[M,Q]}$. Se m é um elemento de M então $A[P,\,]$ $A[\,,Q]$

$$A_{[m,Q]}$$

é o *vetor* sobre Q cujo componente q é $A_{[m,q]}$ para todo q em Q. Usaremos a abreviatura $A_{[m,\,]}$ para $A_{[m,N]}$ e diremos que esse vetor é a **linha** m de A. $A[m,\,]$ linha

Analogamente, para qualquer parte P de M e qualquer elemento n de N, a expressão $A_{[P,n]}$ denota o vetor cujo componente p é $A_{[p,n]}$ para cada p em P. $A[P,n]$

	1	2	3	4	5	6
7	1	0	0	1	2	3
8	0	1	0	4	5	6
9	0	0	1	7	8	9

	4	2	5	3	1	6
7	1	0	2	0	1	3
9	7	0	8	1	0	9
8	4	1	5	0	0	6

	2	4	5	3	1	6
7	1	0	2	0	1	3
9	7	0	8	1	0	9
8	4	1	5	0	0	6

Figura 1.3: Representação de três matrizes sobre os mesmos conjuntos de índices. Os índices de linhas são 7, 8 e 9. Os índices de colunas são 1, 2, 3, 4, 5 e 6. As duas primeiras figuras representam a mesma matriz; a terceira representa uma matriz diferente.

$$\begin{array}{rrrrr} 99 & -4 & 12 & 13 & 77 \\ 98 & 14 & 19 & -7 & 88 \\ 32 & 11 & 22 & 9 & -6 \end{array}$$

Figura 1.4: Quando os índices de uma matriz não estão indicados explicitamente, fica subentendido que os índices das linhas são $1, 2, 3, \ldots$ de cima para baixo e que os índices das colunas são $1, 2, 3, \ldots$ da esquerda para a direita.

	a	b	c	d	e	f	g
f	3	1	9	0	0	4	9
g	4	0	8	1	0	5	9
h	5	0	7	0	1	6	9

	b	d	e
f	1	0	0
g	0	1	0
h	0	0	1

Figura 1.5: A primeira figura representa uma matriz A. A segunda representa a matriz $A[\,, Q]$, onde Q é o conjunto composto pelos índices b, d, e.

Usaremos a abreviatura $A[\,, n]$ para $A[M, n]$ e diremos que este vetor é a **coluna** n de A. $A[\,, n]$ coluna

Convém não confundir as expressões $A[P, n]$ e $A[P, \{n\}]$: a primeira denota um vetor, enquanto a segunda denota uma matriz (com uma só coluna). Alguns livros mais antigos fazem essa confusão conscientemente, e usam a expressão "vetor coluna" para designar qualquer matriz dotada de uma única coluna e a expressão "vetor linha" para uma matriz com uma única linha.

Uma matriz A é **nula** se $A[m, n] = 0$ para todo par m, n. A matriz nula será denotada por O, quaisquer que sejam os seus conjuntos de índices. matriz nula O

Se A é uma matriz e λ é um número então λA é a matriz que se obtém quando cada componente de A é multiplicado por λ. Analogamente, A/λ é o vetor que se obtém dividindo por λ cada componente de A. λA A/λ

1.3 Produtos

Matrizes e vetores podem ser multiplicados entre si. A versão mais básica dessa operação de multiplicação envolve dois vetores.

Produto vetor-por-vetor. Para quaisquer vetores x e y sobre um mesmo conjunto N, o **produto** de x por y é o número

$$\sum_{n \in N} x[n]\, y[n]\,,$$

que denotaremos por $x \cdot y$. É óbvio que $x \cdot y = y \cdot x$. Ademais, para qualquer parte Q de N,

$x \cdot y$

$$x \cdot y \;=\; x[Q] \cdot y[Q] \;+\; x[N{-}Q] \cdot y[N{-}Q]\,.$$

a	b	c	d	e
11	22	33	44	55

e	b	d	c	a
35	41	37	39	43

Figura 1.6: Se x e y são os vetores definidos pela figura então $x \cdot y = 11 \cdot 43 + 22 \cdot 41 + 33 \cdot 39 + 44 \cdot 37 + 55 \cdot 35 = 6215$. Imagine que a, b, c, d, e são os modelos de um produto fabricado por certa empresa e que $y[n]$ é o lucro sobre cada unidade do modelo n. Se foram fabricadas $x[n]$ unidades do modelo n então $x \cdot y$ é o lucro total.

Produtos matriz-por-vetor e vetor-por-matriz. Para qualquer matriz A sobre $M \times N$ e qualquer vetor x sobre N, o **produto de A por x** é o vetor $A \cdot x$ definido pela expressão

produto matriz por vetor $A \cdot x$

$$(A \cdot x)[m] = A[m, \,] \cdot x$$

para cada m em M. É claro que $A \cdot x$ é um vetor sobre M. Analogamente, para qualquer vetor y sobre M, o **produto de y por A** é o vetor $y \cdot A$ definido pela expressão

produto vetor por matriz $y \cdot A$

$$(y \cdot A)[n] = y \cdot A[\,, n]$$

para cada n em N. É fácil verificar que, para qualquer parte P de M e qualquer parte Q de N,

$$(A \cdot x)[P] = A[P, \,] \cdot x \qquad \text{e} \qquad (y \cdot A)[Q] = y \cdot A[\,, Q]\,.$$

É menos fácil verificar a propriedade associativa

$$y \cdot (A \cdot x) = (y \cdot A) \cdot x\,.$$

Produto matriz-por-matriz. Para qualquer matriz A sobre $L \times M$ e qualquer matriz B sobre $M \times N$, o **produto de** A **por** B é a matriz $A \cdot B$ sobre $L \times N$ definida pela expressão $A \cdot B$

$$(A \cdot B)_{[l,n]} = A_{[l,\,]} \cdot B_{[\,,n]}$$

para cada l em L e cada n em N. É fácil verificar que, para qualquer parte P de L e qualquer parte Q de N,

$$(A \cdot B)_{[P,Q]} = A_{[P,\,]} \cdot B_{[\,,Q]}\,.$$

É menos fácil verificar a propriedade associativa propriedade associativa

$$(A \cdot B) \cdot C = A \cdot (B \cdot C)\,,$$

válida para quaisquer matrizes A, B e C cujos conjuntos de índices permitam definir os produtos $A \cdot B$ e $B \cdot C$. Analogamente, $A \cdot (B \cdot x) = (A \cdot B) \cdot x$ e $(y \cdot A) \cdot B = y \cdot (A \cdot B)$ para quaisquer vetores x e y, desde que cada um dos produtos faça sentido.

Notação. Vamos apelar, muitas vezes, ao "princípio universal da preguiça" e escrever xy, Ax, yA e AB no lugar de $x \cdot y$, $A \cdot x$, $y \cdot A$ e $A \cdot B$ respectivamente. xy Ax yA AB

O operador de indexação $[\,,\,]$ tem precedência sobre o operador de multiplicação. Assim, expressões da forma $BA_{[P,Q]}$ e $yA_{[P,Q]}$ devem ser entendidas como $B \cdot (A_{[P,Q]})$ e $y \cdot (A_{[P,Q]})$ respectivamente. Em certas condições, os dois operadores comutam: se os produtos BA e yA fazem sentido então $BA_{[P,Q]}$

$$(BA)_{[\,,Q]} = B \cdot (A_{[\,,Q]}) \quad \text{e} \quad (yA)_{[\,,Q]} = y \cdot (A_{[\,,Q]})\,.$$

1.4 Matrizes inversíveis

O problema mais básico da álgebra linear é o da inversão das operações de multiplicação que definimos acima: dada uma matriz A e um vetor b,

$$\text{encontrar um vetor } x \text{ tal que } Ax = b\,.$$

Analogamente, dado um vetor c, encontrar um vetor y tal que $yA = c$. Ou ainda, dadas matrizes A e B, encontrar uma matriz X tal que $AX = B$; analogamente, dadas matrizes A e C, encontrar uma matriz Y tal que $YA = C$. Estes problemas levam naturalmente aos seguintes conceitos.

Uma matriz I sobre $M \times N$ é a **identidade** se $M = N$ e, para cada par i, j de elementos de M, identidade

$$I_{[i,\,j]} \;=\; \text{se } i = j \text{ então } 1 \text{ senão } 0\,.$$

Toda matriz identidade será denotada por I, quaisquer que sejam seus conjuntos de índices. I

	a	b	d	c
a	1	0	0	0
b	0	1	0	0
c	0	0	1	0
d	0	0	0	1

Figura 1.7: Esta matriz não é a identidade.

Uma **inversa esquerda** de uma matriz A é uma matriz E tal que $EA = I$. Uma matriz A é **inversível pela esquerda** se possui uma inversa esquerda. A **inversa direita** de uma matriz A é uma matriz D tal que $AD = I$. Uma matriz A é **inversível pela direita** se possui uma inversa direita.

inversa esquerda

inversa direita

Se uma matriz tem uma inversa esquerda e uma inversa direita então as duas inversas são iguais. De fato, se $AD = I$ e $EA = I$ então

$$E = E(AD) = (EA)D = D\,.$$

Ademais, as inversas são únicas. De fato, para qualquer matriz D' tal que $AD' = I$ tem-se $D' = (EA)D' = E(AD') = E = D$. Analogamente, para qualquer E' tal que $E'A = I$ tem-se $E' = E$.

A propósito, eis um fato fundamental mas não-trivial: uma matriz que tenha o mesmo número de linhas e colunas tem inversa direita se e só se tem inversa esquerda. Este fato será demonstrado, implicitamente, no próximo capítulo.

Os problemas que mencionamos no início da seção podem ser imediatamente resolvidos se tivermos uma matriz inversa apropriada. Por exemplo, se A tem uma inversa esquerda e direita E, então o vetor $x = Eb$ satisfaz a equação $Ax = b$.

$$\begin{bmatrix} 1 & 2 & 2 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 1 & -1 & 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} 1 & -2 & 0 & 0 \\ 0 & 1 & -1/2 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & -1 & 1 & 1 \\ 1 & 2 & 3 & 4 \end{bmatrix}$$

Figura 1.8: A primeira matriz é uma inversa esquerda da segunda (e a segunda é uma inversa direita da primeira).

1.5 Transposição

A **transposta** de uma matriz A sobre $M \times N$ é a matriz $\widetilde{A}$ definida pelas equações

transposta $\widetilde{A}$

$$\widetilde{A}_{[n,\,m]} = A_{[m,\,n]}$$

$$\begin{array}{cccc} 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 2 \end{array}$$

Figura 1.9: Uma matriz não-inversível.

para cada par m, n. Portanto, $\widetilde{A}$ é uma matriz sobre $N \times M$. É claro que a transposta de $\widetilde{A}$ é A. É fácil verificar que

$$Ax = x\widetilde{A}$$

para todo vetor x tal que o produto de A por x esteja definido. Também é fácil verificar que

$$\widetilde{AB} = \widetilde{B}\widetilde{A}$$

para toda matriz B tal que o produto de A por B esteja definido.

1.6 Matrizes de bijeção

A seguinte generalização do conceito de matriz identidade é muito útil. Uma matriz J sobre $M \times N$ é **de bijeção**[1] se existe uma função bijetora φ de M em N tal que

matriz de bijeção

$$J_{[m,\,n]} \;=\; \text{se } \varphi(m) = n \text{ então } 1 \text{ senão } 0\,.$$

Portanto, uma matriz com componentes 0 e 1 é de bijeção se cada uma de suas colunas tem exatamente um 1 e cada uma de suas linhas tem exatamente um 1. É óbvio que $|M| = |N|$ se existe uma matriz de bijeção sobre $M \times N$.

A transposta de uma matriz de bijeção sobre $M \times N$ é uma matriz de bijeção sobre $N \times M$. Essa segunda matriz é inversa da primeira, como mostraremos a seguir.

Fato *Se J é uma matriz de bijeção então $J\,\widetilde{J} = I$ e $\widetilde{J}\,J = I$.*

DEMONSTRAÇÃO: Para qualquer par i, j de índices de linhas de J, o componente i, j de $J\widetilde{J}$ é o produto de duas linhas de J:

$$(J\widetilde{J})_{[i,\,j]} \;=\; J_{[i,\,]}\widetilde{J}_{[\,,j]} \;=\; J_{[i,\,]}J_{[\,j,\,]}\,.$$

Como J é matriz de bijeção, $J_{[i,\,]}J_{[\,j,\,]}$ é igual a 1 ou 0 conforme $i = j$ ou $i \neq j$. Isso mostra que $J\widetilde{J} = I$. O mesmo raciocínio, com $\widetilde{J}$ no papel de J, mostra que $\widetilde{J}J = I$. □

[1] Generaliza o conceito de matriz de permutação; uma **matriz de permutação** é uma matriz de bijeção cujo conjunto de índices de linhas é idêntico ao conjunto de índices de colunas.

Qual o resultado da multiplicação de uma matriz arbitrária por uma matriz de bijeção? Suponha que J é uma matriz de bijeção sobre $M \times N$. Digamos que $J_{[m,n]} = 1$ para algum m em M e algum n em N. Então, para qualquer matriz A cujo conjunto de índices de linhas seja N, a linha m de JA é idêntica à linha n de A:

$$(JA)_{[m,\,]} \;=\; A_{[n,\,]}\,.$$

Analogamente, para qualquer matriz B que tenha colunas indexadas por M, a coluna n de BJ é idêntica à coluna m de B. Em suma, a pré-multiplicação de A por J apenas redefine os nomes das linhas de A, e a pós-multiplicação de B por J apenas redefine os nomes das colunas de B.

$$\begin{array}{cccc} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{array}$$

Figura 1.10: Uma matriz de bijeção.

1.7 Matrizes diagonais

Uma matriz D sobre $M \times N$ é **diagonal** se $M = N$ e $D_{[m,n]} = 0$ sempre que $m \neq n$. Em particular, toda matriz identidade é diagonal. diagonal

Se D é uma matriz diagonal tal que $D_{[m,m]} \neq 0$ para todo m então a matriz diagonal E definida pelas equações

$$E_{[m,m]} = 1/D_{[m,m]}$$

é uma inversa esquerda e também uma inversa direita de D. Portanto, E é a *única* inversa de D. Por outro lado, se D é uma matriz diagonal e $D_{[m,m]} = 0$ para algum m então é fácil verificar que D não tem inversa.

1.8 Matrizes elementares

Uma **matriz-coluna** coincide com a identidade em todas as colunas, exceto talvez uma. Em outras palavras, uma matriz F sobre $M \times M$ é uma matriz-coluna se existe k em M tal que matriz-coluna

$$F_{[M,\,M-k]} = I_{[M,\,M-k]}\,,$$

onde $M - k$ é uma abreviatura de $M - \{k\}$. Diremos que k é a **coluna saliente** da matriz. $M-k$ coluna saliente

Fato *Para qualquer matriz-coluna F com coluna saliente k, se $F_{[k,k]}$ não é nulo então a matriz-coluna G com coluna saliente k definida pelas equações*

$$G_{[k,k]} = \frac{1}{F_{[k,k]}} \quad \text{e} \quad G_{[m,k]} = \frac{-F_{[m,k]}}{F_{[k,k]}}$$

para cada m em $M - k$ é uma inversa esquerda e também uma inversa direita de F.

DEMONSTRAÇÃO: Mostremos inicialmente que $GF = I$. Na coluna k temos $(GF)_{[k,k]} = G_{[k,\,]}F_{[\,,k]} = G_{[k,k]}F_{[k,k]} = F_{[k,k]}/F_{[k,k]} = 1$; ademais,

$$\begin{aligned}(GF)_{[m,k]} &= G_{[m,\,]}F_{[\,,k]} \\ &= G_{[m,m]}\,F_{[m,k]} + G_{[m,k]}\,F_{[k,k]} \\ &= F_{[m,k]} - F_{[k,k]}\,F_{[m,k]}/F_{[k,k]} \\ &= 0\end{aligned}$$

para cada m em $M-k$. Portanto, a coluna k de GF é igual à coluna k da matriz identidade. Para concluir, considere as colunas distintas de k:

$$(GF)_{[\,,M-k]} = GF_{[\,,M-k]} = GI_{[\,,M-k]} = G_{[\,,M-k]} = I_{[\,,M-k]}\,.$$

Portanto, $GF = I$. Para mostrar que $FG = I$, basta observar que as mesmas regras que definem G a partir de F também geram F a partir de G. □

Nas condições da proposição acima, G é a única inversa (esquerda e direita) de F. Também é fácil verificar que se $F_{[k,k]} = 0$ para algum k então F não tem inversa.

$$\begin{array}{ccccc} 1 & 0 & 0 & 4 & 0 \\ 0 & 1 & 0 & 5 & 0 \\ 0 & 0 & 1 & 6 & 0 \\ 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 8 & 1 \end{array} \qquad \begin{array}{ccccc} 1 & 0 & 0 & -4/7 & 0 \\ 0 & 1 & 0 & -5/7 & 0 \\ 0 & 0 & 1 & -6/7 & 0 \\ 0 & 0 & 0 & 1/7 & 0 \\ 0 & 0 & 0 & -8/7 & 1 \end{array}$$

Figura 1.11: Matrizes-coluna com coluna saliente 4. Uma é inversa da outra.

Uma matriz F sobre $M \times M$ é uma **matriz-linha** se existe h em M tal que $F_{[M-h,\,M]} = I_{[M-h,\,M]}$. Diremos que h é a **linha saliente** de F. Uma observação análoga à que demonstramos acima vale para matrizes-linha: se F é uma matriz-linha com linha saliente h e $F_{[h,h]} \neq 0$ então a matriz-linha G com linha saliente h definida pelas equações

matriz-linha

$$G_{[h,h]} = 1/F_{[h,h]} \qquad \text{e} \qquad G_{[h,n]} = -F_{[h,n]}/F_{[h,h]}$$

para cada n em $M-h$ é a única inversa esquerda de F e também a única inversa direita de F.

Diremos que uma matriz é **elementar** se for uma matriz-coluna ou uma matriz-linha. Note que os conjuntos de índices de linhas e colunas de uma matriz elementar são idênticos. Matrizes elementares e suas inversas terão um papel de destaque nos capítulos subseqüentes.

matriz elementar

1.9 Combinações lineares

Suponha que $a_1, .., a_k$ são vetores sobre um mesmo conjunto de índices. Uma **combinação linear** desses vetores é qualquer vetor da forma

$$\lambda_1 a_1 + \cdots + \lambda_k a_k \,,$$

onde $\lambda_1, .., \lambda_k$ são números. Esses números são os **coeficientes** da combinação linear.

Suponha que A é uma matriz sobre $M \times N$. Para todo vetor x sobre N, o vetor Ax é uma combinação linear das colunas de A com coeficientes $x_{[j]}$, isto é,

$$Ax = \textstyle\sum_{j \in N} A_{[\,,j]}\, x_{[j]} \,.$$

Analogamente, para todo vetor y sobre M, o vetor yA é uma combinação linear das linhas de A, isto é,

$$yA = \textstyle\sum_{i \in M} y_{[i]}\, A_{[i,\,]} \,.$$

Se A e B são matrizes tais que o produto AB faz sentido então cada coluna de AB é uma combinação linear das colunas de A e cada linha de AB é uma combinação linear das linhas de B:

$$(AB)_{[\,,j]} = A\, B_{[\,,j]} \qquad \text{e} \qquad (AB)_{[i,\,]} = A_{[i,\,]}\, B \,.$$

Exercícios

1.1 Demonstre a propriedade associativa do produto de matrizes: se cada um dos produtos faz sentido, então $A(BC) = (AB)C$.

1.2 Mostre que o produto de matrizes não é comutativo: AB é, em geral, diferente de BA (mesmo que os dois produtos estejam definidos).

1.3 Suponha que x e y são vetores e que A e B são matrizes. Quantas operações de multiplicação são necessárias para calcular xy? para calcular Ax? yA? AB?

1.4 Suponha que $AB = I$ e $BC = I$. Mostre que B é inversa direita de C.

1.5 Seja A a primeira das matrizes abaixo. Encontre uma matriz de bijeção J tal que AJ seja a segunda das matrizes. Encontre uma matriz de bijeção J tal que JA seja a terceira matriz.

	a	b	c	d	e
f	11	12	13	14	15
g	21	22	23	24	25
k	31	32	33	34	35

	b	c	a	f	h
f	11	12	13	14	15
g	21	22	23	24	25
k	31	32	33	34	35

	a	b	c	d	e
g	11	12	13	14	15
k	21	22	23	24	25
i	31	32	33	34	35

1.6 Sejam F e G matrizes sobre $M \times M$ e D uma matriz sobre $M \times N$. Suponha que $FG = I$ e que a matriz $E = GD$ é de bijeção. Verifique que $D\widetilde{E}G = I$.

1.7 Suponha que A é uma matriz de bijeção sobre $M \times N$ e b é um vetor arbitrário sobre M. Verifique que existe um e um só vetor x tal que $Ax = b$.

Parte I

Algoritmos Básicos

Capítulo 2

Algoritmo de Gauss-Jordan

Encontre números x_1, x_2, x_3 e x_4 que satisfaçam as equações

$$\begin{aligned} d_{11}\,x_1 + d_{12}\,x_2 + d_{13}\,x_3 + d_{14}\,x_4 &= b_1 \\ d_{21}\,x_1 + d_{22}\,x_2 + d_{23}\,x_3 + d_{24}\,x_4 &= b_2 \\ d_{31}\,x_1 + d_{32}\,x_2 + d_{33}\,x_3 + d_{34}\,x_4 &= b_3 \end{aligned}$$

O algoritmo de Gauss-Jordan[1] é a ferramenta básica da álgebra linear. O algoritmo transforma qualquer matriz em uma matriz equivalente dotada de um certo "desenho" ou "padrão", que descreveremos na seção 2.1.

Ao estudar qualquer algoritmo, é preciso enfrentar duas perguntas: *o que* o algoritmo faz? *como* faz o que faz? No caso do algoritmo de Gauss-Jordan, ao contrário do que ocorre com outros algoritmos célebres, é mais fácil tratar da segunda pergunta. Assim, começaremos com um esboço de *como* o algoritmo funciona.

2.1 Matrizes escalonadas

matriz escalonada

Uma matriz E sobre $M \times N$ é **escalonada** se existem uma parte P de M e uma parte Q de N tais que

$$E_{[P,\,Q]} \text{ é uma matriz de bijeção} \quad \text{e} \quad E_{[M-P,\,N]} = O\,.$$

bases

Os conjuntos P e Q são as **bases** da matriz; o conjunto Q é a **base de colunas** e P é a **base de linhas**. É óbvio que toda matriz escalonada tem uma única base de linhas, mas pode ter várias bases de colunas distintas. (Convém lembrar que não estamos fazendo quaisquer restrições sobre os valores relativos de $|M|$ e $|N|$. Também não estamos presumindo qualquer relação de ordem em M ou N.)

[1] Referências ao célebre Carl Friedrich Gauss (1777–1855) e ao (menos célebre) Wilhelm Jordan (1842–1899), que popularizou o algoritmo [Jor20].

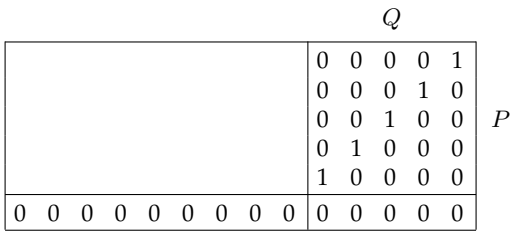

Figura 2.1: Matriz escalonada.

Figura 2.2: Exemplos de matrizes escalonadas. A primeira tem bases $\emptyset$ e $\emptyset$. A segunda tem base de linhas $1, 2, 3, 4$ e base de colunas $1, 2, 3, 4$. Na última, a base de linhas é $\{2, 3\}$ e há duas bases de colunas distintas: $\{1, 3\}$ e $\{1, 6\}$.

2.2 Esboço do algoritmo

O algoritmo de Gauss-Jordan recebe uma matriz D sobre $M \times N$ e transforma D numa matriz escalonada. Cada iteração do algoritmo começa com uma parte P de M e uma matriz E. A primeira iteração começa com $P = \emptyset$ e $E = D$. Cada iteração consiste no seguinte:

D
$M \times N$
P
E

CASO 1: $E_{[M-P,\,]} \neq O$.
Escolha h em $M - P$ e k em N de modo que $E_{[h,k]} \neq 0$.
Seja E' a matriz definida pelas equações $E'_{[h,\,]} = E_{[h,\,]}/E_{[h,k]}$ e $E'_{[i,\,]} = E_{[i,\,]} - E_{[i,k]}\, E_{[h,\,]}/E_{[h,k]}$ para cada i em $M - h$.
Comece nova iteração com $P + h$ e E' nos papéis de P e E.

CASO 2: $E_{[M-P,\,]} = O$.
Devolva E e pare. □

A expressão $P + h$ é uma abreviatura de $P \cup \{h\}$. É claro que se $P = M$ no início de uma iteração então aplica-se o caso 2.

No início de cada iteração existe uma parte Q de N tal que $E_{[P,Q]}$ é uma matriz de bijeção e $E_{[M-P,Q]}$ é nula. A demonstração desta propriedade será feita mais adiante, depois que o algoritmo for reescrito de modo mais completo. Se a propriedade vale no início da última iteração então é óbvio que a matriz E é escalonada.

$$\begin{array}{rrrr} 1 & 1 & 2 & 0 \\ 2 & 1 & 5 & -1 \\ 2 & 2 & 4 & 0 \\ -1 & 1 & -3 & 1 \end{array}$$

$$\begin{array}{rrrr} 0 & 1/2 & -1/2 & 1/2 \\ 1 & 1/2 & 5/2 & -1/2 \\ 0 & 1 & -1 & 1 \\ 0 & 3/2 & -1/2 & 1/2 \end{array}$$

$$\begin{array}{rrrr} 0 & 1 & -1 & 1 \\ 1 & 0 & 3 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{array}$$

$$\begin{array}{rrrr} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{array}$$

Figura 2.3: Aplicação do esboço do algoritmo de Gauss-Jordan. A figura descreve a matriz E no início de sucessivas iterações. A última matriz é escalonada.

$$\begin{array}{rrrr} 1 & 1 & 0 & 2 \\ 1 & 2 & 0 & 5 \\ 2 & 2 & 1 & 4 \\ 1 & -1 & 0 & -3 \end{array} \qquad \begin{array}{rrrr} -1 & 1 & 0 & 1 \\ 8 & -5 & 0 & -3 \\ -2 & 0 & 1 & 0 \\ -3 & 2 & 0 & 1 \end{array} \qquad \begin{array}{rrrr} 1 & 1 & 2 & 0 \\ 2 & 1 & 5 & -1 \\ 2 & 2 & 4 & 0 \\ -1 & 1 & -3 & 1 \end{array}$$

Figura 2.4: A figura define matrizes F, G e D. Verifique que FG é a identidade e que GD é escalonada.

$$\begin{array}{rrrrr} 1 & 0 & 0 & 0 & \\ 0 & 1 & 0 & 0 & \\ 0 & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & \end{array} \qquad \begin{array}{rrrrr} 1 & 0 & 2 & 3 & 4 \\ 0 & 1 & 4 & 5 & 6 \\ 6 & 7 & 8 & 9 & 0 \\ 9 & 8 & 7 & 6 & 5 \end{array} \qquad \begin{array}{rrrrr} 1 & 0 & 2 & 3 & 4 \\ 0 & 1 & 4 & 5 & 6 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}$$

Figura 2.5: A figura define matrizes G e D e exibe GD. Observe que GD é escalonada mas não existe F tal que $FG = I$.

2.3 Algoritmo

Para dizer, exatamente, *o que* o algoritmo faz é preciso especificar a relação entre as matrizes E e D. A matriz E é equivalente à matriz D no seguinte sentido: existe uma matriz inversível G tal que

$$E \;=\; GD\,.$$

O esboço da seção anterior não devolve G, o que impede o usuário de conferir a equivalência entre E e D. A versão do algoritmo que descreveremos abaixo devolve G e sua inversa F; o usuário pode então, ao preço de duas multiplicações de matrizes, verificar que G é inversível e que GD é escalonada.

Algoritmo de Gauss-Jordan *Recebe uma matriz D sobre $M \times N$ e devolve matrizes F e G sobre $M \times M$ tais que $FG = I$ e GD é escalonada.*

Cada iteração começa com matrizes F, G e E e uma parte P de M. No início da primeira iteração, $F = G = I$, $E = D$ e $P = \emptyset$. Cada iteração consiste no seguinte: F G

CASO 1: $E_{[h,\,]} \neq o$ *para algum* h *em* $M - P$. h
Escolha k em N tal que $E_{[h,k]} \neq 0$. k
Seja F', G', E' o resultado da pivotação de F, G, E em torno de h, k.
Comece nova iteração com F', G', E' e $P + h$
nos papéis de F, G, E e P.

CASO 2: $E_{[M-P,\,]} = O$.
Devolva F, G e pare. □[2]

A operação de pivotação a que se refere o texto do algoritmo é definida da seguinte maneira: dados elementos h de M e k de N, o **resultado da pivotação de F, G, E em torno de h, k** é o terno F', G', E' de matrizes definido pelas equações pivotação

$$\begin{aligned}
F'_{[\,,h]} &= D_{[\,,k]}\,, & F'_{[\,,i]} &= F_{[\,,i]} \qquad\qquad\;\;, \\
G'_{[h,\,]} &= \alpha_h\, G_{[h,\,]}\,, & G'_{[i,\,]} &= G_{[i,\,]} + \alpha_i\, G_{[h,\,]}\,, \\
E'_{[h,\,]} &= \alpha_h\, E_{[h,\,]}\,, & E'_{[i,\,]} &= E_{[i,\,]} + \alpha_i\, E_{[h,\,]}\,,
\end{aligned}$$

para cada i em $M - h$, onde

$$\alpha_h = 1 \,/\, E_{[h,k]} \quad \text{e} \quad \alpha_i = -E_{[i,k]} \,/\, E_{[h,k]}\,.$$

Número de iterações. É claro que o algoritmo de Gauss-Jordan converge — ou seja, sua execução pára depois de um número finito de iterações — pois P aumenta a cada ocorrência do caso 1. O número total de iterações é, no máximo, $|M|$. converge

[2] Veja exercício 2.5.

$$
\begin{array}{cccc|cccc|cccc}
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 2 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 1 & 5 & -1 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 2 & 2 & 4 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & -1 & 1 & -3 & 1
\end{array}

\begin{array}{cccc|cccc|cccc}
1 & 1 & 0 & 0 & 1 & -1/2 & 0 & 0 & 0 & 1/2 & -1/2 & 1/2 \\
0 & 2 & 0 & 0 & 0 & 1/2 & 0 & 0 & 1 & 1/2 & 5/2 & -1/2 \\
0 & 2 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & 1 & -1 & 1 \\
0 & -1 & 0 & 1 & 0 & 1/2 & 0 & 1 & 0 & 3/2 & -1/2 & 1/2
\end{array}

\begin{array}{cccc|cccc|cccc}
1 & 1 & 0 & 0 & 2 & -1 & 0 & 0 & 0 & 1 & -1 & 1 \\
1 & 2 & 0 & 0 & -1 & 1 & 0 & 0 & 1 & 0 & 3 & -1 \\
2 & 2 & 1 & 0 & -2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & -1 & 0 & 1 & -3 & 2 & 0 & 1 & 0 & 0 & 1 & -1
\end{array}

\begin{array}{cccc|cccc|cccc}
1 & 1 & 0 & 2 & -1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
1 & 2 & 0 & 5 & 8 & -5 & 0 & -3 & 1 & 0 & 0 & 2 \\
2 & 2 & 1 & 4 & -2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & -1 & 0 & -3 & -3 & 2 & 0 & 1 & 0 & 0 & 1 & -1
\end{array}
$$

Figura 2.6: Exemplo de aplicação do algoritmo de Gauss-Jordan. A figura registra os valores de F, G e E no início de sucessivas iterações.

$$
\begin{array}{cccc|cccc|cccc}
1 & 0 & 0 & 0 & 2 & 1 & 2 & 4 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & 2 & 1 & 3 \\
0 & 0 & 1 & 0 & 2 & 1 & 0 & 5 & -1 & 4 \\
0 & 0 & 0 & 1 & -1 & -1 & 1 & -2 & 1 & 3
\end{array}

\begin{array}{cccc|cccc|cccc}
1/2 & 0 & 0 & 0 & 1 & 1/2 & 1 & 2 & 0 & 1/2 \\
-1/2 & 1 & 0 & 0 & 0 & 1/2 & 0 & 0 & 1 & 5/2 \\
-1 & 0 & 1 & 0 & 0 & 0 & -2 & 1 & -1 & 3 \\
1/2 & 0 & 0 & 1 & 0 & -1/2 & 2 & 0 & 1 & 7/2
\end{array}

\begin{array}{cccc|cccc|cccc}
1 & -1 & 0 & 0 & 1 & 0 & 1 & 2 & -1 & -2 \\
-1 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 5 \\
-1 & 0 & 1 & 0 & 0 & 0 & -2 & 1 & -1 & 3 \\
0 & 1 & 0 & 1 & 0 & 0 & 2 & 0 & 2 & 6
\end{array}

\begin{array}{cccc|cccc|cccc}
1/2 & -1 & 1/2 & 0 & 1 & 0 & 0 & 5/2 & -3/2 & -1/2 \\
-1 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 5 \\
1/2 & 0 & -1/2 & 0 & 0 & 0 & 1 & -1/2 & 1/2 & -3/2 \\
-1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 9
\end{array}

\begin{array}{cccc|cccc|cccc}
3 & -7/2 & -2 & -5/2 & 1 & 0 & 0 & 0 & -4 & -23 \\
-1 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 5 \\
0 & 1/2 & 0 & 1/2 & 0 & 0 & 1 & 0 & 1 & 3 \\
-1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 9
\end{array}
$$

Figura 2.7: Exemplo de aplicação do algoritmo de Gauss-Jordan. A figura registra os valores de G e E no início de sucessivas iterações (F foi omitida por falta de espaço). Observe como a matriz identidade que estava inicialmente em G move-se para a direita, invadindo E.

2.4 Análise do algoritmo

A chave para entender como e por que o algoritmo funciona está na seguinte lista de propriedades. As propriedades valem no início de cada iteração e são, por isso mesmo, chamadas invariantes.

Invariantes *No início de cada iteração do algoritmo,*

(i1) $E_{[P,Q]}$ *é uma matriz de bijeção e* $E_{[M-P,Q]} = O$ *,*

(i2) $FG = I$ *e*

(i3) $GD = E$ *,*

onde Q *é uma parte de* N *(que poderíamos chamar* base de colunas *da iteração).*

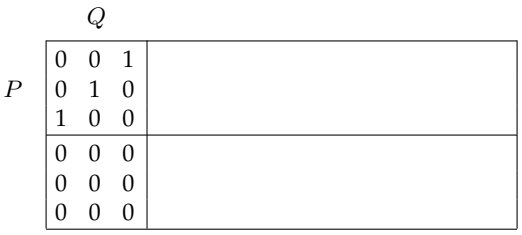

Figura 2.8: Matriz E no início de uma iteração do algoritmo de Gauss-Jordan.

Essas propriedades valem, em particular, no início da última iteração, quando ocorre o caso 2. Nesse caso, E é escalonada em virtude de (i1) e da definição do caso 2; além disso, $FG = I$ em virtude de (i2). Portanto, ao devolver F e G o algoritmo estará se comportanto como prometeu.

DEMONSTRAÇÃO DOS INVARIANTES: É evidente que as propriedades valem no início da primeira iteração (com P e Q vazios). Suponha agora que as propriedades valem no início de uma iteração qualquer que não a última. Então ocorre o caso 1 (com k em $N - Q$) e as propriedades passam a valer com

$$F',\ G',\ E',\ P + h \text{ e } Q + k$$

nos papéis de F, G, E, P e Q. Para demonstrar esta afirmação basta verificar que no fim do caso 1 tem-se

$$\begin{aligned} E'_{[\,,Q]} &= E_{[\,,Q]}\,, && \text{(2.a)}\\ E'_{[\,,k]} &= I_{[\,,h]}\,, && \text{(2.b)}\\ F'G' &= I\,, && \text{(2.c)}\\ E' &= G'D\,. && \text{(2.d)} \end{aligned}$$

	Q			k
P	0	0	1	0
	0	1	0	0
	1	0	0	0
h	0	0	0	0
	0	0	0	1
	0	0	0	0

Figura 2.9: Matriz E' no fim do caso 1 do algoritmo de Gauss-Jordan.

A demonstração de (2.a) é elementar. Por definição da operação de pivotação, temos

$$E'[h,\,] = \alpha_h E[h,\,] \quad \text{e} \quad E'[i,\,] = E[i,\,] + \alpha_i E[h,\,]$$

para cada i em $M - h$. Como o vetor $E[h,Q]$ é nulo em virtude de (i1), temos $E'[\,,Q] = E[\,,Q]$.

Antes de empreender as demonstrações de (2.b) a (2.d), é conveniente dar uma representação matricial à operação de pivotação. Seja $\check{F}$ a matriz elementar (veja seção 1.8) sobre $M \times M$ cuja coluna saliente, h, é igual a $E[\,,k]$, isto é, $\check{F}$

$$\check{F}[\,,h] = E[\,,k] \quad \text{e} \quad \check{F}[\,,M-h] = I[\,,M-h]\,.$$

Seja $\check{G}$ a inversa de $\check{F}$, isto é, a matriz elementar com coluna saliente h definida pelas equações $\check{G}$

$$\check{G}[h,h] = 1/E[h,k] \quad \text{e} \quad \check{G}[i,h] = -E[i,k]/E[h,k]$$

para cada i em $M - h$. Observe que $\check{F}\check{G} = \check{G}\check{F} = I$. Observe também que

$$F' \;=\; F\check{F}\,, \quad G' \;=\; \check{G}G \quad \text{e} \quad E' \;=\; \check{G}E\,.$$

As duas últimas identidades são óbvias. A primeira merece uma verificação mais cuidadosa: na coluna h temos

$$F'[\,,h] = D[\,,k] = FG \cdot D[\,,k] = F \cdot E[\,,k] = F \cdot \check{F}[\,,h]$$

e nas demais colunas temos

$$F'[\,,M-h] = F[\,,M-h] = F \cdot I[\,,M-h] = F \cdot \check{F}[\,,M-h]\,.$$

Portanto, o resultado da pivotação de F, G, E em torno de h, k é o terno de matrizes $F\check{F}$, $\check{G}G$, $\check{G}E$.

Agora podemos cuidar das demonstrações de (2.b) a (2.d). A demonstração de (2.b) decorre das igualdades $\check{G}\check{F} = I$ e $E' = \check{G}E$: para cada i em M,

$$\begin{aligned} E'[i,k] &= \check{G}[i,\,]E[\,,k] \\ &= \check{G}[i,\,]\check{F}[\,,h] \\ &= (\check{G}\check{F})[i,h] \\ &= I[i,h]\,. \end{aligned}$$

A demonstração de (2.c) é fácil:

$$F'G' = (F\check{F})(\check{G}G) = F(\check{F}\check{G})G = FG = I\,.$$

A prova de (2.d) é igualmente fácil: $E' = \check{G}E = \check{G}(GD) = (\check{G}G)D = G'D$. □

2.5 Mais invariantes

O algoritmo de Gauss-Jordan tem mais quatro invariantes, além dos que discutimos na seção anterior. Não é necessário ter consciência desses invariantes adicionais para compreender o funcionamento do algoritmo; mas eles são um prenúncio de invariantes fundamentais do Simplex, cujo estudo empreenderemos a partir do próximo capítulo.

Invariantes *No início de cada iteração do algoritmo,*

(i4) $G_{[\,,\,M-P]} = I_{[\,,\,M-P]}$,

(i5) $F_{[\,,\,M-P]} = I_{[\,,\,M-P]}$ e $F_{[\,,\,P]} = D_{[\,,\,Q]}\,\widetilde{J}$,

onde $\widetilde{J}$ *é a transposta da matriz de bijeção* $E_{[P,\,Q]}$.

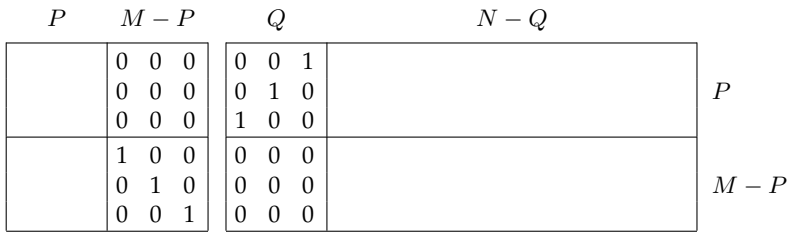

Figura 2.10: Matrizes G e E no início de uma iteração do algoritmo de Gauss-Jordan. A justaposição de G e E contém uma matriz de bijeção que ocupa todas as linhas.

DEMONSTRAÇÃO: É óbvio que (i4) vale no início da primeira iteração. Para demonstrar que o invariante permanece válido no início das demais iterações, basta provar que no fim de cada ocorrência do caso 1 temos

$$G'_{[\,,\,M-P-h]} = G_{[\,,\,M-P-h]}\,,$$

onde $M-P-h$ é uma abreviatura de $(M-P)-\{h\}$. A demonstração é análoga à de (2.a). Por definição,

$$G'_{[h,\,]} = \alpha_h G_{[h,\,]} \quad \text{e} \quad G'_{[i,\,]} = G_{[i,\,]} + \alpha_i G_{[h,\,]}$$

para cada i em $M - h$. Mas $G_{[h,\, M-P-h]}$ é nulo em virtude de (i4). Logo, $G'_{[\,,\, M-P-h]} = G_{[\,,\, M-P-h]}$.

O invariante (i5) segue imediatamente da maneira como F' é definida a partir de F em cada iteração. A multiplicação por $\widetilde{J}$ apenas troca os nomes das colunas que estão em Q de modo que $E_{[P,\,Q]}\,\widetilde{J}$ seja a matriz identidade sobre $P \times P$. O invariante mostra que a variável F não precisa ser atualizada a cada iteração: ela pode muito bem ser calculada no caso 2, imediatamente antes do fim da execução do algoritmo. □

Invariantes *No início de cada iteração do algoritmo,*

(i6) $D_{[M-P,\,Q]} = -G_{[M-P,\,P]}\,D_{[P,\,Q]}$ *e*
(i7) $D_{[P,\,Q]}\,\widetilde{J}\,G_{[P,\,P]} = I$,

onde $\widetilde{J}$ é a transposta da matriz de bijeção $E_{[P,\,Q]}$.

DEMONSTRAÇÃO: Seja i um elemento de $M - P$. Em virtude de (i4), para cada i em $M - P$,

$$\begin{aligned} G_{[i,\,P]}D_{[P,\,Q]} + D_{[i,\,Q]} &= G_{[i,\,P]}D_{[P,\,Q]} + G_{[i,\,M-P]}D_{[M-P,\,Q]} \\ &= G_{[i,\,M]}D_{[M,\,Q]} \\ &= (GD)_{[i,\,Q]} \\ &= E_{[i,\,Q]} \\ &= o\,, \end{aligned}$$

onde as duas últimas igualdades são conseqüência de (i3) e (i1) respectivamente. Logo, $G_{[i,\,P]}D_{[P,\,Q]} = -D_{[i,\,Q]}$. Isto demonstra (i6). Agora considere (i7). Em virtude da segunda parte de (i5),

$$D_{[P,\,Q]}\,\widetilde{J}\,G_{[P,\,P]} = F_{[P,\,P]}\,G_{[P,\,P]}\,.$$

Mas $F_{[P,\,P]}G_{[P,\,P]} = I$ por força de (i2), (i4) e da primeira parte de (i5). □

O invariante (i6) mostra que, para cada i em $M - P$, o vetor $D_{[i,\,Q]}$ é uma combinação linear das linhas da matriz $D_{[P,\,Q]}$. O invariante (i7) mostra que $\widetilde{J}\,G_{[P,\,P]}$ é uma inversa direita de $D_{[P,\,Q]}$. A propósito, os invariantes (i3) e (i4) mostram que $\widetilde{J}\,G_{[P,\,P]}$ é uma inversa esquerda de $D_{[P,\,Q]}$.

2.6 Número de operações aritméticas

Não é difícil estimar, em termos dos parâmetros $m = |M|$ e $n = |N|$, o número de operações aritméticas que o algoritmo executa. É possível implementar o algoritmo (veja exercício 2.6, página 24) de modo que cada pivotação exija apenas mn multiplicações e divisões e menos que mn adições e subtrações. Como o

algoritmo executa no máximo m tais pivotações, o número total de operações aritméticas é menor que

$$2m^2n\,.$$

(A título de comparação, observe que a multiplicação de G por D requer m^2n multiplicações e outras tantas adições.)

O consumo total de tempo do algoritmo depende não só do número de operações aritméticas mas também do tempo dispendido em cada operação. Antes de estimar esse tempo, é preciso entender que tipo de números o algoritmo manipula. É razoável restringir nosso universo aos números racionais: como o algoritmo envolve apenas as operações de soma, subtração, multiplicação e divisão, ele transforma números racionais em outros números racionais. Assim, se os componentes da matriz dada D são racionais então os componentes das matrizes F, G e E serão sempre racionais.

números racionais

Cada número racional tem a forma α/δ, onde α é um inteiro e δ um inteiro não-nulo. O número α/δ é representado pelo par ordenado $\langle\alpha, \delta\rangle$; o primeiro elemento do par é o **numerador** e o segundo é o **denominador** da representação. O custo de uma operação aritmética sobre números racionais depende da magnitude dos numeradores e denominadores dos números envolvidos. Será necessário, portanto, obter uma delimitação superior para os valores absolutos dos numeradores e denominadores gerados pelo algoritmo. Faremos isto no capítulo 11. Podemos adiantar que esses números são, em geral, muito maiores que os numeradores e denominadores dos componentes da matriz dada D.

numerador
denominador

2.7 Conclusão

O algoritmo de Gauss-Jordan transforma qualquer matriz dada em uma matriz escalonada equivalente. O algoritmo, juntamente com sua análise, constitui prova do seguinte teorema:

> *Para toda matriz D, existem matrizes F e G tais que $FG = I$ e GD é escalonada.*

Vale também o seguinte adendo: se os componentes de D são números racionais então existem matrizes *racionais* F e G com as propriedades citadas.

O algoritmo de Gauss-Jordan é muitas vezes executado de modo apenas aproximado: os números são representados "em ponto flutuante", com um número fixo de dígitos, e as operações aritméticas são executadas com erro de arredondamento. Os erros podem mascarar completamente os resultados; números que deveriam ser nulos, por exemplo, podem se apresentar como diferentes de zero, e o algoritmo pode deixar de reconhecer o caso 2. Há uma grande coleção de truques que visam controlar, em alguma medida, tais erros de arredondamento [Chv83].

$$\begin{array}{rrrrrrrrrr}
3 & 2 & 1 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
4 & 3 & 2 & 5 & 6 & 7 & 8 & 9 & 73 & 1 \\
5 & 6 & 3 & 4 & 7 & 9 & 8 & 10 & 1 & 2 \\
1 & 5 & 4 & 7 & 5 & 10 & 9 & 6 & 2 & 3 \\
32 & 6 & 5 & 8 & 9 & 10 & 9 & 2 & 7 & 4 \\
8 & 7 & 6 & 9 & 10 & 5 & 2 & 3 & 4 & 5 \\
9 & 8 & 7 & 11 & 3 & 2 & 1 & 4 & 5 & 8
\end{array}$$

$$\begin{array}{rrrrrrr}
-629/246 & 39615/16892 & -6827/25338 & 1687/12669 & 298/12669 & 955/50676 & -1970/12669 \\
173/246 & -11887/16892 & 10079/25338 & -2704/12669 & -748/12669 & -5203/50676 & 2309/12669 \\
2/41 & -531/8446 & 3/4223 & -72/4223 & 155/4223 & -113/8446 & 24/4223 \\
130/123 & -7651/8446 & -1445/12669 & 896/12669 & -52/12669 & 937/25338 & 1109/12669 \\
-22/123 & 1281/4223 & -115/12669 & -1463/12669 & -311/12669 & 1462/12669 & -920/12669 \\
67/82 & -20105/16892 & -125/8446 & 1500/4223 & 290/4223 & 1893/16892 & -500/4223 \\
-73/82 & 20509/16892 & 75/8446 & -900/4223 & -174/4223 & -2825/16892 & 300/4223
\end{array}$$

$$\begin{array}{rrrrrrrrr}
0 & 0 & 1 & 0 & 0 & 0 & 0 & -14885/8446 & 7482125/50676 & -309364/12669 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 19365/8446 & -2279561/50676 & 91123/12669 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & -907/4223 & -33379/8446 & 2114/4223 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 165/4223 & -1419745/25338 & 133228/12669 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 2205/4223 & 256175/12669 & -24730/12669 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & -18515/8446 & -1326005/16892 & 33380/4223 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 19555/8446 & 1344593/16892 & -36920/4223
\end{array}$$

Figura 2.11: Ao receber a primeira matriz da figura, digamos D, o algoritmo de Gauss-Jordan poderá devolver a segunda matriz no papel de G. A terceira matriz é GD.

2.8 Aplicação: Sistemas de equações

Considere o seguinte problema: *Dada uma matriz A sobre $M \times N$ e um vetor b sobre M, encontrar um vetor x tal que $Ax = b$.* Para resolver o problema, comece por submeter A ao algoritmo de Gauss-Jordan. O algoritmo devolverá matrizes F e G tais que $FG = I$ e GA é escalonada.

Digamos que as bases de GA são P e Q e suponha inicialmente que o vetor $(Gb)_{[M-P]}$ é nulo. Seja x o único vetor que satisfaz as equações

$$x_{[N-Q]} = o \quad \text{e} \quad (GA)_{[P,Q]}\, x_{[Q]} = (Gb)_{[P]}\,.$$

vetor básico

Este é o **vetor básico** associado à base Q. É claro que $(GA)x = Gb$, donde $F(GA)x = F(Gb)$ e portanto $Ax = b$.[3]

Suponha agora que $(Gb)_{[h]}$ não é nulo para algum h em $M - P$. É claro que nesse caso não existe x tal que $(GA)x = Gb$. Nosso problema original também não tem solução, como passamos a demonstrar. Seja g o vetor $G_{[h,\,]}$ e observe

[3] Esse método de cálculo de x não é o mais eficiente. É possível obter x com apenas um terço do número de multiplicações se o algoritmo de Gauss-Jordan for modificado de modo a produzir uma matriz triangular no lugar da matriz de bijeção $(GA)[P,Q]$. Essa variante do algoritmo é conhecida como **método de eliminação** de Gauss [Chv83, cap.6] [CLRS01, cap.31].

que

$$gA = (GA)_{[h,\,]} = 0 \quad \text{enquanto} \quad gb = (Gb)_{[h]} \neq 0\,.$$

Se existisse um vetor x tal que $Ax = b$ teríamos a contradição $0 = (gA)x = g(Ax) = gb \neq 0$.

O vetor g constitui, portanto, um "certificado" facilmente verificável de que a equação $Ax = b$ não tem solução.

Exercícios

2.1 Escreva um algoritmo que decide se uma dada matriz é escalonada e em caso afirmativo devolve o seu par de bases.

2.2 Mostre que $GF = I$ no início de cada iteração do algoritmo de Gauss-Jordan.

2.3 O algoritmo descrito no texto devolve apenas as matrizes F e G. Escreva uma versão que devolva também a matriz escalonada E e suas bases P e Q.

2.4 Escreva uma versão do algoritmo de Gauss-Jordan em que a matriz F seja calculada somente na última iteração.

2.5 Escreva o algoritmo de Gauss-Jordan em uma linguagem mais formal, mais próxima de PASCAL ou C. (Veja solução parcial E.1 no apêndice E.) Programe o algoritmo em um computador.

2.6 Escreva uma versão um pouco mais eficiente do algoritmo de Gauss-Jordan: execute apenas implicitamente a parte da operação de pivotação que afeta a base de colunas Q e a coluna k.

2.7 Suponha que a execução do algoritmo de Gauss-Jordan é interrompida no início de uma iteração e que uma pivotação é executada em torno de um elemento h de P (e não de $M - P$, como é usual) e um elemento k de $N - Q$. É claro que isso só faz sentido se $E_{[h,\,k]} \neq 0$. Qual o efeito de tal pivotação? A execução do algoritmo pode ser retomada depois da pivotação excepcional?

2.8 Suponha que A e B são matrizes sobre $M \times M$ e que P é uma parte de M. Suponha ainda que $AB = I$ e $A_{[\,,\,M-P]} = B_{[\,,\,M-P]} = I_{[\,,\,M-P]}$. Mostre que $A_{[P,\,P]}B_{[P,\,P]} = I$.

2.9 Mostre que no início de cada iteração do algoritmo de Gauss-Jordan a matriz $G_{[M-P,\,P]}$ é completamente determinada pelas matrizes $D_{[M-P,\,Q]}$, $E_{[P,\,Q]}$ e $G_{[P,\,P]}$.

2.10 Sejam $\dot{G}$, $\dot{P}$ e $\dot{Q}$ os valores das variáveis G, P e Q no início de uma iteração. Sejam $\ddot{G}$, $\ddot{P}$ e $\ddot{Q}$ os valores das variáveis G, P e Q no início de outra iteração. Mostre que se $\dot{P} = \ddot{P}$ e $\dot{Q} = \ddot{Q}$ então $\dot{G}_{[M-\dot{P},\,]} = \ddot{G}_{[M-\ddot{P},\,]}$.

2.11 Suponha que G_1 e G_2 são matrizes inversíveis tais que G_1D e G_2D são escalonadas. Mostre que quaisquer bases de colunas, digamos Q_1 e Q_2, das duas matrizes escalonadas têm a mesma cardinalidade. Essa cardinalidade comum é o **posto** da matriz D. A propósito, diz-se que D tem **posto pleno** se seu posto é igual ao número de linhas da matriz. (Solução no apêndice E, página 192.)

2.12 Encontre números $x_1, \ldots, x_4$ que satisfaçam as equações abaixo.

$$\begin{aligned} x_1 &+ x_2 &+ x_3 &+ 3x_4 &= 6 \\ 3x_1 &+ 4x_2 &+ x_3 &+ 8x_4 &= 18 \\ -x_1 &+ x_2 &- 4x_3 &- 4x_4 &= -5 \end{aligned}$$

2.13 Resolva o seguinte problema: dada uma matriz A sobre $M \times N$, um vetor b sobre M e um vetor c sobre N, encontrar x tal que $Ax = b$ e cx é mínimo. O problema sempre tem solução? (Veja apêndice E, página 192.)

Capítulo 3

Introdução ao Simplex

Encontre números não-negativos x_1, x_2, x_3 e x_4 que satisfaçam as equações

$$d_{11}\,x_1 + d_{12}\,x_2 + d_{13}\,x_3 + d_{14}\,x_4 = d_{15}$$
$$d_{21}\,x_1 + d_{22}\,x_2 + d_{23}\,x_3 + d_{24}\,x_4 = d_{25}$$
$$d_{31}\,x_1 + d_{32}\,x_2 + d_{33}\,x_3 + d_{34}\,x_4 = d_{35}$$

e minimizem a soma $d_{41}\,x_1 + d_{42}\,x_2 + d_{43}\,x_3 + d_{44}\,x_4$

O algoritmo Simplex é a ferramenta básica da programação linear. O objetivo do algoritmo é transformar uma matriz dada em outra equivalente que contenha um certo "desenho" ou "padrão", que descreveremos na seção 3.1.

Este capítulo faz um esboço do Simplex, destacando seu parentesco com o algoritmo de Gauss-Jordan discutido no capítulo anterior. Nosso esboço contém todos os elementos básicos do Simplex, mas não chega a ser um algoritmo pois em geral não converge. Os dois próximos capítulos mostrarão como refinar o esboço para transformá-lo num algoritmo.

Como no estudo de qualquer algoritmo, é preciso enfrentar duas perguntas: *o que* faz o Simplex? *como* faz o que faz? Este capítulo trata principalmente da segunda pergunta. A primeira será respondida no próximo capítulo. Uma terceira pergunta — *para que* serve o Simplex? — será adiada até o capítulo 7, ainda que isso possa tornar um tanto árida e indigesta a tarefa de entender o conceito de matriz simples.

3.1 Matrizes simples

Os dados do Simplex são uma matriz sobre $M \times N$, um elemento n de N e um elemento m de M. Diremos que n é a **coluna especial** e que m é a **linha especial** da matriz. Nas figuras, a coluna especial será sempre a que estiver mais à direita e a linha especial será sempre a última. A propósito, não estamos fazendo quaisquer restrições sobre os valores relativos de $|M|$ e $|N|$ e não estamos presumindo qualquer relação de ordem em M ou N.

n m
coluna especial
linha especial

Nosso estudo começa com uma descrição das características da matriz que o Simplex calcula. Diremos que uma matriz E é **simples com relação ao par** n, m **de índices** se for de um dos três tipos definidos abaixo: simples solúvel, simples inviável ou simples ilimitada. Não vamos nos preocupar, por enquanto, com as conotações das palavras "solúvel", "inviável" e "ilimitada"; elas serão justificadas no capítulo 7. As definições poderão parecer indigestas, mas deverão ficar mais naturais depois que fizermos um esboço do Simplex.

matriz simples

Matriz simples solúvel. Dada uma matriz E sobre $M \times N$ e elementos n e m de N e M respectivamente, diremos que E é **simples solúvel** com relação ao par n, m se existem partes P de $M - m$ e Q de $N - n$ tais que

$$\begin{aligned} & E_{[P,\,Q]} \text{ é de bijeção}\,, \quad E_{[P,\,n]} \geq o\,, \\ & E_{[M-m-P,\,N]} = O\,, \\ E_{[m,\,N-n-Q]} \geq o\,, \quad & E_{[m,\,Q]} = o\,. \end{aligned}$$

A expressão $M - m - P$ é uma abreviatura de $(M - \{m\}) - P$; analogamente, $N-n-Q$ é uma abreviatura de $(N-\{n\})-Q$. É óbvio que a matriz $E_{[M-m,\,N-n]}$ é escalonada. O conjunto P é a **base de linhas** e o conjunto Q é uma **base de colunas** da matriz. É fácil verificar que a base de linhas é única, mas podem existir várias bases de colunas.

$M - m - P$
$N - n - Q$
bases

Há uma certa simetria entre $E_{[m,\,N-n]}$ e $E_{[M-m,\,n]}$ na definição de matriz simples solúvel: a condição $E_{[P,\,n]} \geq o$ corresponde à condição $E_{[m,\,Q]} = o$; e a condição $E_{[M-m-P,\,n]} = o$ corresponde à condição $E_{[m,\,N-n-Q]} \geq o$.

Matriz simples inviável. Uma matriz E sobre $M \times N$ é **simples inviável** com relação à coluna n e linha m se existe h em $M - m$ tal que

linha de inviabilidade

$$\begin{aligned} E_{[h,\,N-n]} \leq o \quad \text{e} \quad E_{[h,\,n]} > 0 \quad \text{ou} \\ E_{[h,\,N-n]} \geq o \quad \text{e} \quad E_{[h,\,n]} < 0\,. \end{aligned}$$

O elemento h de $M - m$ é o índice de uma **linha de inviabilidade**.

Matriz simples ilimitada. Uma matriz E sobre $M \times N$ é **simples ilimitada** com relação à coluna n e linha m se existe uma parte P de $M - m$, uma parte Q de $N - n$ e um elemento k de $N - n - Q$ tais que

coluna de ilimitação

$$\begin{aligned} E_{[P,\,k]} \leq o\,, \quad & E_{[P,\,Q]} \text{ é de bijeção}\,, \quad & E_{[P,\,n]} \geq o\,, \\ E_{[M-m-P,\,k]} = o\,, \quad & E_{[M-m-P,\,Q]} = O\,, \quad & E_{[M-m-P,\,n]} = o\,, \\ E_{[m,\,k]} < 0\,, \quad & E_{[m,\,Q]} = o\,. \end{aligned}$$

Os conjuntos P e Q são as **bases** da matriz e k é o índice de uma **coluna de ilimitação**.

Há uma certa simetria entre a definição de matriz simples inviável e a de matriz simples ilimitada: a condição "$E_{[h,\,N-n]} \leq o$ e $E_{[h,\,n]} > 0$ ou $E_{[h,\,N-n]} \geq o$ e $E_{[h,\,n]} < 0$" da primeira corresponde à condição "$E_{[M-m,\,k]} \leq o$ e $E_{[m,\,k]} < 0$" da segunda.

												Q			n
P												0	0	1	$\geq$
												0	1	0	$\geq$
												1	0	0	$\geq$
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
m	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	0	0	0	

Figura 3.1: Matriz simples solúvel.

															n	
	$\leq$	$\leq$	$\leq$	$\leq$	$\leq$	$\leq$	$\leq$	$\leq$	$\leq$	$\leq$	$\leq$	$\leq$	$\leq$	$\leq$	$>$	h
m																

Figura 3.2: Matriz simples inviável.

	k			Q			n
P	$\leq$			0	0	1	$\geq$
	$\leq$			0	1	0	$\geq$
	$\leq$			1	0	0	$\geq$
	0			0	0	0	0
	0			0	0	0	0
	0			0	0	0	0
m	$<$			0	0	0	

Figura 3.3: Matriz simples ilimitada.

Matriz simples. Uma matriz E é **simples** com relação aos índices n e m se for de um dos três tipos descritos acima. As três definições são mutuamente exclusivas: uma matriz não pode ser, ao mesmo tempo, simples solúvel e inviável, nem simples solúvel e ilimitada, nem simples inviável e ilimitada.

3.2 Esboço do Simplex

Suponha dada uma matriz D sobre $M \times N$ e elementos n e m de N e M respectivamente. O objetivo do Simplex é transformar D, por meio de sucessivas D

$$\begin{array}{rrrrrrrrr}
1 & 99 & 0 & 0 & 99 & 0 & 0 & 99 & 88 \\
0 & 99 & 0 & 0 & 99 & 0 & 1 & 99 & 88 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/2 & 0 & 0 & 1/2 & 1 & 0 & 1/2 & 0 \\
0 & -99 & 1 & 1 & -99 & 0 & 0 & -99 & 88 \\
0 & 77 & 0 & 0 & 77 & 0 & 0 & 77 & -66
\end{array}$$

Figura 3.4: Esta matriz é simples solúvel com relação à coluna 9 e linha 6. A base de linhas é composta de $1, 2, 4, 5$. Há duas bases de colunas: uma contém $1, 3, 6, 7$ e outra contém $1, 4, 6, 7$.

$$\begin{array}{rrrrrrrrr}
1 & 99 & 0 & 99 & 99 & 0 & 0 & 99 & 88 \\
0 & -99 & 0 & -99 & -99 & 0 & -99 & -99 & 88 \\
0 & 0 & 0 & 99 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/2 & 0 & 1/2 & 1/2 & 1 & 0 & 1/2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
-99 & 99 & 0 & -99 & 99 & 0 & 0 & 99 & -66
\end{array}$$

Figura 3.5: Matriz simples inviável (coluna especial 9 e linha especial 6). Há duas linhas de inviabilidade: 2 e 5.

$$\begin{array}{rrrrrrrrr}
1 & 99 & 0 & 0 & 99 & 0 & 0 & 99 & 88 \\
0 & -99 & 0 & -99 & -99 & 0 & 1 & 99 & 88 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/2 & 0 & -1/2 & 1/2 & 1 & 0 & 1/2 & 0 \\
0 & 99 & 1 & -99 & 99 & 0 & 0 & 99 & 88 \\
0 & 77 & 0 & -77 & 77 & 0 & 0 & 77 & -66
\end{array}$$

Figura 3.6: Matriz simples ilimitada. A coluna de ilimitação é 4.

operações de pivotação, numa matriz que seja simples em relação a n e m.

Cada iteração do Simplex começa com uma matriz E e uma parte P de $M - m$ tais que E P

$$f_{[P]} \geq o\,,$$
$$E_{[P,Q]} \text{ é uma matriz de bijeção e } E_{[M-P,Q]} = O\,,$$

onde f denota o vetor $E_{[\,,n]}$ e Q é uma parte de $N-n$. As operações executadas durante uma iteração modificam E e P de modo a preservar essas propriedades. f Q A primeira iteração começa com $E = D$ e $P = \emptyset$. Cada iteração consiste no seguinte:

CASO 1: $E_{[h,k]} > 0$ e $f_{[h]} \geq 0$ ou $E_{[h,k]} < 0$ e $f_{[h]} \leq 0$
para algum h *em* $M - m - P$ *e algum* k *em* $N - n$. h k
Seja P^* o conjunto de todos os p em P para os quais $E_{[p,k]} > 0$. P^*

CASO 1A: $f_{[h]}/E_{[h,k]} \leq f_{[p]}/E_{[p,k]}$ *para todo* p *em* P^*.
Seja E' o resultado da pivotação de E em torno de h, k.
Comece nova iteração com $P + h$ e E' nos papéis de P e E.

CASO 1B: $f_{[h]}/E_{[h,k]} > f_{[p]}/E_{[p,k]}$ *para algum* p *em* P^*.
Escolha p em P^* de modo que $f_{[p]}/E_{[p,k]}$ seja mínimo. p
Seja E' o resultado da pivotação de E em torno de p, k.
Comece nova iteração com E' no papel de E (sem alterar P).

CASO 2: $E_{[h,N-n]} \leq o$ e $f_{[h]} > 0$ ou $E_{[h,N-n]} \geq o$ e $f_{[h]} < 0$
para algum h *em* $M - m - P$. h
Devolva E e pare (a matriz E é simples inviável).

CASO 3: $E_{[M-m-P,N]} = O$, $f_{[M-m-P]} = o$ e $E_{[m,k]} < 0$
paraalgum k *em* $N - n$. k
Seja P^* o conjunto de todos os p em P para os quais $E_{[p,k]} > 0$. P^*

CASO 3A: P^* *é vazio*.
Devolva E e pare (a matriz E é simples ilimitada).

CASO 3B: P^* *não é vazio*.
Escolha p em P^* de modo que $f_{[p]}/E_{[p,k]}$ seja mínimo. p
Seja E' o resultado da pivotação de E em torno de p, k.
Comece nova iteração com E' no papel de E (sem alterar P).

CASO 4: $E_{[M-m-P,N]} = O$, $f_{[M-m-P]} = o$ e $E_{[m,N-n]} \geq o$.
Devolva E e pare (a matriz E é simples solúvel). □

A operação de pivotação a que se referem os casos 1A, 1B e 3B é definida como no algoritmo de Gauss-Jordan (seção 2.3): dados elementos h de $M - m$ e k de $N - n$ (as pivotações jamais ocorrem na linha m e jamais na coluna n), o **resultado da pivotação de** E **em torno de** h, k é a matriz E' definida pelas pivotação

equações

$$E'_{[h,\,]} = \frac{1}{E_{[h,k]}}\, E_{[h,\,]} \quad \text{e} \quad E'_{[i,\,]} = E_{[i,\,]} - \frac{E_{[i,k]}}{E_{[h,k]}}\, E_{[h,\,]}$$

para cada i em $M - h$.

É claro que os casos 1A e 1B devem ser entendidos como subcasos do caso 1; analogamente, 3A e 3B são subcasos de 3. Os casos 1B e 3B são formalmente idênticos. A definição de p nesses casos deve ser entendida da seguinte maneira: escolha *qualquer* p em P^* que satisfaça a condição $f_{[p]}/E_{[p,k]} \le f_{[i]}/E_{[i,k]}$ para todo i em P^*.

	Q				n
P	0	0	1		$\ge$
	0	1	0		$\ge$
	1	0	0		$\ge$
	0	0	0		
	0	0	0		
	0	0	0		
m	0	0	0		

Figura 3.7: Matriz E no início de uma iteração.

A estrutura de casos. Em cada iteração, pelo menos um dos quatro casos se aplica, como passamos a mostrar. Se os casos 1 e 2 não valem para um elemento h de $M - m - P$ então

$$E_{[h,\,N-n]} = o \quad \text{e} \quad f_{[h]} = 0\,.$$

Por outro lado, se essa condição é verdadeira para todo h em $M - m - P$ então é óbvio que vale o caso 3 ou o caso 4.

Como se vê, os quatro casos se complementam naturalmente. Essa complementaridade determina a "lógica interna" do Simplex e justifica nossas definições de matriz simples solúvel, inviável e ilimitada.

Nossa linguagem algorítmica. Em nossa linguagem de descrição de algoritmos, a ordem em que os casos são enunciados é irrelevante: em cada iteração, qualquer dos casos aplicáveis pode ser executado. O mesmo vale para os subcasos dentro de cada caso. Ademais, a definição de um caso não supõe que os demais não se aplicam. É possível mesmo que os casos não sejam mutuamente exclusivos (embora isso não ocorra no esboço do Simplex acima).

A propósito, as expressões lógicas da forma "X e Y ou W e Z" que aparecem na definição de alguns casos devem ser entendidas como "(X e Y) ou (W e Z)".

$$
\begin{array}{rrrrr}
1 & 1 & 5 & -1 & 5 \\
0 & 1 & -1 & 1 & 4 \\
0 & -1 & -3 & -1 & -8 \\
0 & -1 & 1 & 2 & 3
\end{array}
$$

$$
\begin{array}{rrrrr}
1 & 0 & 6 & -2 & 1 \\
0 & 1 & -1 & 1 & 4 \\
0 & 0 & -4 & 0 & -4 \\
0 & 0 & 0 & 3 & 7
\end{array}
$$

$$
\begin{array}{rrrrr}
1/6 & 0 & 1 & -1/3 & 1/6 \\
1/6 & 1 & 0 & 2/3 & 25/6 \\
2/3 & 0 & 0 & -4/3 & -10/3 \\
0 & 0 & 0 & 3 & 7
\end{array}
$$

$$
\begin{array}{rrrrr}
0 & 0 & 1 & 0 & 1 \\
1/2 & 1 & 0 & 0 & 5/2 \\
-1/2 & 0 & 0 & 1 & 5/2 \\
3/2 & 0 & 0 & 0 & -1/2
\end{array}
$$

Figura 3.8: Exemplo de aplicação do Simplex. A figura registra a matriz E no início de sucessivas iterações. Neste exemplo, a primeira iteração começa com $P = \{1\}$ e $Q = \{1\}$. O subcaso 1A se aplica com $h, k = 2, 2$. Na segunda iteração, o caso 1 se aplica com $h, k = 3, 3$; o subcaso 1A não se aplica, mas o subcaso 1B vale com $p = 1$. No início da terceira iteração, a matriz E não é mais simples que no início da iteração anterior, mas em algum sentido é melhor que aquela: depois de uma pivotação em torno de $3, 4$ obtemos uma matriz simples solúvel.

Terminologia tradicional. Convém mencionar alguns termos consagrados pelo uso em discussões sobre o Simplex. Nos casos 1 e 3, dizemos que a coluna k **entra na base**. Nos casos 1B e 3B, dizemos que **sai da base** a coluna q definida pela relação $E_{[p,\,q]} = 1$.

entra na base

sai da base

Qual o critério para a escolha da coluna k que deverá entrar na base? No caso 1, basta que $E_{[h,\,k]}$ não seja nulo e $f_{[h]}/E_{[h,\,k]}$ não seja negativo; no caso 3, basta que $E_{[m,\,k]}$ seja negativo.[1] Qual o critério para a escolha da coluna q que deverá sair da base nos casos 1B e 3B? Como a matriz $E_{[P,\,Q]}$ estabelece uma bijeção entre P e Q, o critério de escolha de q se confunde com o critério de escolha de p e p é escolhido em P^* de modo que $f_{[p]}/E_{[p,\,k]}$ seja mínimo.

A propósito, o esboço que fizemos acima corresponde, essencialmente, à chamada **versão tabular** do Simplex.

[1] Dizer que α é *negativo* é o mesmo que dizer "$\alpha < 0$". Analogamente, α é *positivo* se $\alpha > 0$.

$$
\begin{array}{rrrrr}
2 & 2 & 4 & 0 & 24 \\
2 & 0 & 5 & -1 & -10 \\
-1 & 1 & -2 & 1 & 2 \\
1 & 1 & 1 & 1 & 0
\end{array}
\qquad
\begin{array}{rrrrr}
1 & 1 & 2 & 0 & 12 \\
0 & -2 & 1 & -1 & -34 \\
0 & 2 & 0 & 1 & 14 \\
0 & 0 & -1 & 1 & -12
\end{array}
\qquad
\begin{array}{rrrrr}
1 & 1 & 2 & 0 & 12 \\
0 & 2 & -1 & 1 & 34 \\
0 & 0 & 1 & 0 & -20 \\
0 & -2 & 0 & 0 & -46
\end{array}
$$

Figura 3.9: Exemplo de aplicação do Simplex. A figura registra a matriz E no início de sucessivas iterações. A matriz resultante é simples inviável.

$$
\begin{array}{rrrrr}
2 & 2 & 4 & 0 & 24 \\
2 & 0 & 5 & -1 & 10 \\
-1 & 1 & -2 & 1 & 2 \\
1 & 1 & 1 & 1 & 0
\end{array}
\qquad
\begin{array}{rrrrr}
1 & 1 & 2 & 0 & 12 \\
0 & -2 & 1 & -1 & -14 \\
0 & 2 & 0 & 1 & 14 \\
0 & 0 & -1 & 1 & -12
\end{array}
\qquad
\begin{array}{rrrrr}
1 & 0 & 5/2 & -1/2 & 5 \\
0 & 1 & -1/2 & 1/2 & 7 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & -1 & 1 & -12
\end{array}
\qquad
\begin{array}{rrrrr}
1 & 0 & 0 & -1/2 & 5 \\
0 & 1 & 0 & 1/2 & 7 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & -12
\end{array}
$$

$$
\begin{array}{rrrrr}
2 & 2 & 4 & -8 & 24 \\
2 & 0 & 5 & -4 & 10 \\
-1 & 1 & -2 & 0 & 2 \\
1 & 1 & 1 & -8 & 0
\end{array}
\qquad
\begin{array}{rrrrr}
1 & 1 & 2 & -4 & 12 \\
0 & -2 & 1 & 4 & -14 \\
0 & 2 & 0 & -4 & 14 \\
0 & 0 & -1 & -4 & -12
\end{array}
\qquad
\begin{array}{rrrrr}
1 & 0 & 5/2 & -2 & 5 \\
0 & 1 & -1/2 & -2 & 7 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & -1 & -4 & -12
\end{array}
\qquad
\begin{array}{rrrrr}
1 & 0 & 0 & -2 & 5 \\
0 & 1 & 0 & -2 & 7 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & -4 & -12
\end{array}
$$

Figura 3.10: Mais dois exemplos de aplicação do Simplex, ligeiramente diferentes do anterior. No primeiro (à esquerda), a última matriz é simples solúvel. No segundo, a última matriz é simples ilimitada.

3.3 Análise

Como já anunciamos acima, o Simplex gira em torno de duas propriedades (compare com a seção 2.4:

Invariantes *No início de cada iteração,*

(i0) $f[P] \geq o$,
(i1) $E_{[P,Q]}$ *é de bijeção e* $E_{[M-P,Q]} = O$,

onde Q *é uma parte de* N .

Os dois invariantes são trivialmente verdadeiros (com P e Q vazios) no início da primeira iteração. Suponha agora que eles valem no início de uma iteração qualquer que não a última. É claro que nessa iteração ocorre o caso 1 ou o caso 3 e k está necessariamente em $N - Q$. É preciso mostrar que no fim do caso 1A os invariantes permanecem válidos com E', $P + h$ e $Q + k$ nos papéis de E, P e Q e que no fim dos casos 1B e 3B os invariantes permanecem válidos com E' e $Q - q + k$ nos papéis de E e Q, onde q é o único elemento de Q tal que $E_{[p,q]} = 1$. Trocando em miúdos, basta mostrar que no fim do caso 1A temos

$$\begin{aligned} f'_{[P+h]} &\geq o\,, && \text{(3.a)}\\ E'_{[\,,Q]} &= E_{[\,,Q]}\,, && \text{(3.b)}\\ E'_{[\,,k]} &= I_{[\,,h]}\,, && \text{(3.c)} \end{aligned}$$

e que no fim dos casos 1B e 3B temos

$$\begin{aligned} f'_{[P]} &\geq o\,, && \text{(3.d)}\\ E'_{[\,,Q-q]} &= E_{[\,,Q-q]}\,, && \text{(3.e)}\\ E'_{[\,,k]} &= I_{[\,,p]}\,, && \text{(3.f)} \end{aligned}$$

onde $f' = E'_{[\,,n]}$ e I é a matriz identidade. As demonstrações de (3.a) e (3.d) explicam a estrutura de casos do Simplex e a elaborada escolha de p nos casos 1B e 3B.

DEMONSTRAÇÃO DE (3.a): Considere inicialmente o componente h de f'. Em virtude da definição do caso 1,

$$f_{[h]}/E_{[h,k]} \geq 0\,.$$

Como a pivotação no caso 1A ocorre em torno de h, k temos $f'_{[h]} = f_{[h]}/E_{[h,k]}$ e portanto $f'_{[h]} \geq 0$. Resta considerar $f'_{[i]}$ com i em P. Suponha inicialmente que $E_{[i,k]}$ não é positivo. Então $f'_{[i]} \geq 0$ pois

$$f'_{[i]} \;=\; f_{[i]} - E_{[i,k]}\,\frac{f_{[h]}}{E_{[h,k]}}$$

	Q		k		n	
	0	0	1	0	$\geq$	
P	0	1	0	0	$\geq$	
	1	0	0	0	$\geq$	
	0	0	0	1	$\geq$	h
	0	0	0	0		
	0	0	0	0		
m	0	0	0	0		

Figura 3.11: Matriz E' no fim do caso 1A.

e $f[i] \geq 0$ em virtude de (i0). Suponha agora que $E[i,k]$ é positivo. Então $f'[i] \geq 0$ porque

$$f'[i] \;=\; E[i,k]\,(\frac{f[i]}{E[i,k]} - \frac{f[h]}{E[h,k]})$$

e a expressão entre parênteses não é negativa, em virtude da definição do caso 1A. Em suma, $f'[i] \geq 0$ para cada i em $P + h$. □

DEMONSTRAÇÃO DE (3.d): Como a pivotação nos casos 1B e 3B ocorre em torno de p, k, temos $f'[p] = f[p]/E[p,k]$. Mas $f[p] \geq 0$ em virtude de (i0) e $E[p,k] > 0$ pois $p \in P^*$. Logo,

$$f'[p] \geq 0\,.$$

Resta considerar $f'[i]$ com i em $P - p$. Suponha inicialmente que $E[i,k]$ não é positivo. Então $f'[i] \geq 0$ pois

$$f'[i] \;=\; f[i] - E[i,k]\,\frac{f[p]}{E[p,k]}$$

e $f[i] \geq 0$. Suponha agora que $E[i,k]$ é positivo. Então $f'[i] \geq 0$ porque

$$f'[i] \;=\; E[i,k]\,(\frac{f[i]}{E[i,k]} - \frac{f[p]}{E[p,k]})$$

e a expressão entre parênteses não é negativa em virtude da maneira como p foi escolhido. □

As demonstrações de (3.b), (3.c), (3.e) e (3.f) são análogas às demonstrações das relações correspondentes no algoritmo de Gauss-Jordan. A título de ilustração, vamos examinar as duas últimas.

DEMONSTRAÇÃO DE (3.e): Como $E[p, Q-q] = 0$ em virtude de (i1), a definição da operação de pivotação garante que

$$E'[p, Q-q] = E[p, Q-q] \quad \text{e} \quad E'[i, Q-q] = E[i, Q-q]$$

para cada i em $M - p$. □

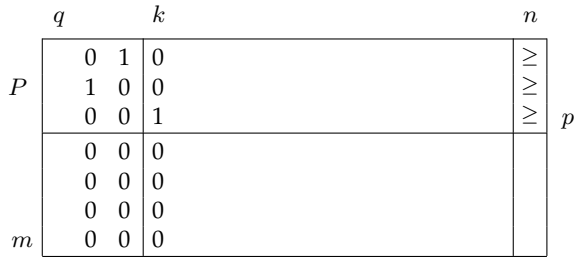

Figura 3.12: Matriz E' no fim dos casos 1B e 3B.

DEMONSTRAÇÃO DE (3.f): Convém representar a operação de pivotação de forma matricial, como já fizemos ao analisar o algoritmo de Gauss-Jordan.

Seja $\check{F}$ a matriz elementar sobre $M \times M$ cuja coluna saliente, p, é igual a $E[\,,k]$. Seja $\check{G}$ a inversa de $\check{F}$, isto é, a matriz elementar com coluna saliente p definida pelas equações $\check{G}[p,p] = 1/E[p,k]$ e $\check{G}[i,p] = -E[i,k]/E[p,k]$ para cada i em $M - p$. Observe que $\check{F}$ $\check{G}$

$$\check{F}\,\check{G} = \check{G}\,\check{F} = I \quad \text{e} \quad E' = \check{G}\,E\,.$$

Agora, para cada i em M,

$$E'[i,k] \;=\; \check{G}[i,\,]E[\,,k] \;=\; \check{G}[i,\,]\check{F}[\,,p] \;=\; (\check{G}\check{F})[i,p] \;=\; I[i,p]\,.$$

Logo, $E'[\,,k] = I[\,,p]$. □

Como acabamos de mostrar, os invariantes (i0) e (i1) valem no início de cada iteração. Em particular, os invariantes valem no início da última iteração. Portanto, a matriz E que o Simplex devolve é simples com relação ao par n, m: no caso 2, E é simples inviável (com linha de inviabilidade h); no caso 3A, E é simples ilimitada (com coluna de ilimitação k); no caso 4, E é simples solúvel (com bases P e Q).

3.4 Convergência

Nosso esboço do Simplex freqüentemente não converge, ou seja, freqüentemente não atinge um dos casos terminais (2, 3A ou 4). Enquanto estiver ocorrendo o caso 1A, é óbvio que estamos fazendo progresso, pois P aumenta. Mas é possível que haja uma seqüência infinita de ocorrências dos casos 1B ou 3B. Isto acontece, em especial, se o caso 1 ocorre em duas iterações consecutivas com um mesmo valor de P e valores distintos de h, como mostra a figura 3.13.

A convergência melhora se insistirmos na mesma linha h, iteração após iteração, enquanto P não se alterar. A justificativa para esta política está na seguinte observação, a ser demonstrada no próximo capítulo: enquanto estiver

$$\begin{array}{rrrr} 1 & 2 & 10 & 2 \\ 0 & 1 & 12 & 2 \\ 0 & -6 & 13 & 3 \\ 0 & 0 & 0 & 0 \end{array}$$

$$\begin{array}{rrrr} 1/2 & 1 & 5 & 1 \\ -1/2 & 0 & 7 & 1 \\ 3 & 0 & 43 & 9 \\ 0 & 0 & 0 & 0 \end{array}$$

$$\begin{array}{rrrr} 1 & 2 & 10 & 2 \\ 0 & 1 & 12 & 2 \\ 0 & -6 & 13 & 3 \\ 0 & 0 & 0 & 0 \end{array}$$

Figura 3.13: Exemplo de não-convergência do esboço do Simplex na seção 3.2. A figura registra o valor de E no início de três iterações consecutivas. Em todas, 1 é o único elemento de P. Na primeira iteração, o caso 1 se aplica com $h, k = 2, 2$ e o subcaso 1B se aplica com $p = 1$. Na segunda iteração, o caso 1 se aplica com $h, k = 3, 1$ e o subcaso 1B se aplica com $p = 1$. No início da terceira iteração temos exatamente a mesma configuração que no início da primeira. Este ciclo poderá se repetir *ad æternum*.

acontecendo o caso 1 com um mesmo valor de P e um mesmo valor de h, o valor absoluto de $f[h]$ diminui ou, na pior das hipóteses, não aumenta. Em outras palavras, numa seqüência de ocorrências do caso 1B, a seqüência de valores de $f[h]$ é monotônica, isto é, crescente ou decrescente.[2]

Exercícios

3.1 Escreva um algoritmo que decida se uma dada matriz E é simples com relação a um dado par n, m de índices.

3.2 Considere o esboço do Simplex feito na seção 3.2. Qual dos casos se aplica numa iteração que começa com $P = M - m$?

3.3 Escreva uma versão especializada do Simplex para matrizes com uma só linha. Escreva uma versão especializada do Simplex para matrizes com apenas duas linhas. Use as duas matrizes abaixo como teste, com $n = 6$ e $m = 2$. (Veja também o exercício 4.3.)

$$\begin{array}{rrrrrr} 2 & 2 & 2 & 2 & 2 & 0 \\ 2 & 1 & 0 & -1 & -2 & 0 \end{array} \qquad \begin{array}{rrrrrr} 2 & 2 & 2 & -2 & 2 & 0 \\ 2 & 1 & 0 & -1 & -2 & 0 \end{array}$$

[2] Diremos que uma seqüência $\alpha_1, \alpha_2, \alpha_3, \ldots$ é *crescente* se $\alpha_1 \leq \alpha_2 \leq \alpha_3 \leq \cdots$ e *decrescente* se $\alpha_1 \geq \alpha_2 \geq \alpha_3 \geq \cdots$.

3.4 Aplique o Simplex descrito na seção 3.2 à matriz abaixo (como de hábito, a coluna especial é a que está mais à direita e a linha especial é a última). Aplique o Simplex várias vezes, procurando obter soluções diferentes.

$$\begin{array}{rrrrrrrr}
1 & 2 & 1 & 0 & 1 & 0 & 0 & 12 \\
2 & 5 & 0 & -1 & 0 & 1 & 0 & 10 \\
-1 & -3 & 1 & 1 & 0 & 0 & 1 & 2 \\
-1 & -2 & -1 & 0 & 0 & 1 & 0 & 0
\end{array}$$

3.5 Suponha que no início de uma iteração do Simplex aplica-se o caso 3 e k é escolhido, por engano, de modo que $E[m,k] \geq 0$. Suponha que a iteração é executada com este valor de k. Como isso afeta a iteração corrente? Como isso afeta o andamento das iterações subseqüentes?

Capítulo 4

Heurística Simplex

"Que a heurística apresente a sua versão dos fatos!"

Este capítulo faz uma descrição completa da heurística Simplex. Esta versão do Simplex não merece o rótulo *algoritmo* pois nem sempre converge.[1] Mas os exemplos de não-convergência são muito mais raros que os do esboço que fizemos no capítulo anterior (seção 3.2). O próximo capítulo mostrará como refinar a heurística para transformá-la num algoritmo.

4.1 Introdução

A heurística Simplex recebe uma matriz D, o índice n de uma coluna e o índice m de uma linha e calcula uma matriz E que é simples com relação ao par n,m. Para dizer *o que*, exatamente, o Simplex faz é preciso especificar a relação entre as matrizes E e D. Essa relação é um pouco mais restritiva que a do algoritmo de Gauss-Jordan (capítulo 2): existem matrizes F e G tais que

$$E = GD\,, \quad G_{[\,,m]} = I_{[\,,m]} \quad \text{e} \quad FG = I\,,$$

onde I é a matriz identidade. A versão da heurística que descreveremos a seguir devolve F e G. De posse dessas matrizes, basta calcular os produtos FG e GD e verificar se, de fato, o primeiro é igual à matriz identidade e o segundo é simples com relação a n,m.

Como $E = GD$, cada linha de E é uma combinação linear das linhas de D. Como $G_{[\,,m]} = I_{[\,,m]}$, cada linha de $E_{[M-m,\,]}$ é uma combinação linear das linhas de $D_{[M-m,\,]}$ e o vetor $E_{[m,\,]}$ é a soma de $D_{[m,\,]}$ com uma combinação linear das linhas de $D_{[M-m,\,]}$.

[1] Uma *heurística* é um procedimento de tentativa-e-erro. No presente texto, o termo é usado para designar um procedimento computacional que (ao contrário de um algoritmo) nem sempre converge, mas produz os resultados desejados quando converge. Um procedimento computacional *converge* se sua execução termina depois de um número finito de iterações, quaisquer que sejam os dados.

$$\begin{array}{cccc} 1 & 2 & 2 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 1 & -1 & 1 \end{array} \qquad \begin{array}{cccc} 1 & -2 & 0 & 0 \\ 0 & 1 & -1/2 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & -1 & 1 & 1 \end{array} \qquad \begin{array}{ccccc} 2 & 2 & 4 & 0 & 24 \\ 1 & 1 & 2 & 0 & 12 \\ 2 & 0 & 5 & -1 & 10 \\ -1 & 1 & -3 & 1 & 2 \end{array}$$

Figura 4.1: A figura especifica matrizes F, G e D. Verifique que $FG = I$, que $G[\,,4] = I[\,,4]$ e que a matriz GD é simples com relação à coluna 5 e linha 4.

4.2 A heurística

A heurística Simplex adota a política, já sugerida no capítulo anterior (veja seção 3.4), de insistir numa mesma linha h enquanto isso fizer sentido. Essa política melhora muito a convergência, embora não seja suficiente para garanti-la. Para implementar a política, introduzimos duas novas variáveis: L e h.

Heurística Simplex *Recebe uma matriz D sobre $M \times N$ e elementos n e m de N e M respectivamente; se convergir, devolve matrizes F e G sobre $M \times M$ tais que $FG = I$, $G[\,,m] = I[\,,m]$ e GD é simples (solúvel, inviável, ou ilimitada) com relação ao par n, m.* D n m

Cada iteração começa com uma parte L de $M - m$, um elemento h de $M - L$, e matrizes F, G e E. A primeira iteração começa com $F = G = I$, $E = D$, $L = \emptyset$ e com h em M, se possível[2] distinto de m. Cada iteração adota a abreviatura $f = E[\,,n]$ e consiste no seguinte: L h f

ALTERNATIVA I: *h é diferente de m.*

CASO I.1: *$E[h,k] > 0$ e $f[h] \geq 0$ ou $E[h,k] < 0$ e $f[h] \leq 0$ para algum k em $N - n$.* k
Seja L^* o conjunto de todos os p em L para os quais $E[p,k] > 0$. L^*

CASO I.1A: *$f[h]/E[h,k] \leq f[p]/E[p,k]$ para todo p em L^*.*
Seja F', G', E' o resultado da pivotação de F, G, E em torno de h, k.
Seja L' o conjunto $L + h$.
Escolha h' em $M - L'$, se possível distinto de m.
Comece nova iteração com L', h', F', G', E'
nos papéis de L, h, F, G, E.

CASO I.1B: *$f[h]/E[h,k] > f[p]/E[p,k]$ para algum p em L^*.*
Escolha qualquer p em L^* tal que $f[p]/E[p,k]$ é mínimo. p
Seja F', G', E' o resultado da pivotação de F, G, E em torno de p, k.
Comece nova iteração com F', G' e E' nos papéis de F, G e E.

[2] Ou seja, se $M - m$ não é vazio.

CASO I.2: $E_{[h,\,N-n]} \leq o$ *e* $f_{[h]} > 0$ *ou* $E_{[h,\,N-n]} \geq o$ *e* $f_{[h]} < 0$.
Devolva F e G e pare.

CASO I.3: $E_{[h,\,N-n]} = o$ *e* $f_{[h]} = 0$.
Seja L' o conjunto $L + h$.
Escolha h' em $M - L'$, se possível distinto de m.
Comece nova iteração com L' e h' nos papéis de L e h.

ALTERNATIVA II: h *é igual a* m .

CASO II.1: $E_{[h,\,k]} < 0$ *para algum* k *em* $N - n$. k
Seja L^* o conjunto de todos os p em L para os quais $E_{[p,\,k]} > 0$. L^*

CASO II.1A: L^* *é vazio* .
Devolva F e G e pare.

CASO II.1B: L^* *não é vazio* .
Escolha qualquer p em L^* tal que $f_{[p]}/E_{[p,\,k]}$ é mínimo. p
Seja F', G', E' o resultado da pivotação de F, G, E em torno de p, k.
Comece nova iteração com F', G' e E' nos papéis de F, G e E.

CASO II.2: $E_{[h,\,N-n]} \geq o$.
Devolva F e G e pare. □[3]

A operação de pivotação é definida como no algoritmo de Gauss-Jordan: dados elementos h de $M - m$ e k de $N - n$, o **resultado da pivotação de** F, G, E **em torno de** h, k é o terno F', G', E' de matrizes definido pelas equações pivotação

$$\begin{aligned} F'_{[\,,\,h]} &= D_{[\,,\,k]}\,, & F'_{[\,,\,i]} &= F_{[\,,\,i]}\,, \\ G'_{[h,\,]} &= \alpha_h\, G_{[h,\,]}\,, & G'_{[i,\,]} &= G_{[i,\,]} + \alpha_i\, G_{[h,\,]}\,, \\ E'_{[h,\,]} &= \alpha_h\, E_{[h,\,]}\,, & E'_{[i,\,]} &= E_{[i,\,]} + \alpha_i\, E_{[h,\,]}\,, \end{aligned}$$

onde $\alpha_h = 1/E_{[h,\,k]}$ e $\alpha_i = -E_{[i,\,k]}/E_{[h,\,k]}$ para cada i em $M - h$. Como já vimos em capítulos anteriores,

$$F' = F\check{F}\,, \quad G' = \check{G}G \quad \text{e} \quad E' = \check{G}E\,,$$

onde $\check{F}$ é a matriz elementar com coluna saliente h definida pela equação $\check{F}_{[\,,\,h]} = E_{[\,,\,k]}$ e $\check{G}$ é a inversa de $\check{F}$. $\check{F}$ $\check{G}$

Observações. É óbvio que uma das duas grandes alternativas se aplica em cada iteração. Dentro de cada alternativa pelo menos um dos casos se aplica, como verificamos no capítulo anterior.

A escolha de h no início da primeira iteração e nos casos I.1A e I.3 deve ser entendida assim: escolha h de modo que $h = m$ somente se $M - L = \{m\}$.

A definição de p nos casos I.1B e II.1B deve ser entendida da seguinte maneira: escolha qualquer p em L^* que satisfaça a condição $f_{[p]}/E_{[p,\,k]} \leq f_{[i]}/E_{[i,\,k]}$ para todo i em L^*.

[3] Ver solução do exercício 4.2 no apêndice E, página 192.

$$
\begin{array}{cccc|cccc|cccccc}
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 2 & 4 & 0 & 24 & \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 5 & -1 & 10 & \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & -1 & 1 & -2 & 1 & 2 & \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & \\
\hline
2 & 0 & 0 & 0 & 1/2 & 0 & 0 & 0 & 1 & 1 & 2 & 0 & 12 & L \\
2 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & -2 & 1 & -1 & -14 & \\
-1 & 0 & 1 & 0 & 1/2 & 0 & 1 & 0 & 0 & 2 & 0 & 1 & 14 & \\
1 & 0 & 0 & 1 & -1/2 & 0 & 0 & 1 & 0 & 0 & -1 & 1 & -12 & \\
\hline
2 & 2 & 0 & 0 & 0 & 1/2 & 0 & 0 & 1 & 0 & 5/2 & -1/2 & 5 & L \\
2 & 0 & 0 & 0 & 1/2 & -1/2 & 0 & 0 & 0 & 1 & -1/2 & 1/2 & 7 & L \\
-1 & 1 & 1 & 0 & -1/2 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & \\
1 & 1 & 0 & 1 & -1/2 & 0 & 0 & 1 & 0 & 0 & -1 & 1 & -12 & \\
\hline
2 & 2 & 4 & 0 & 5/4 & -2 & -5/2 & 0 & 1 & 0 & 0 & -1/2 & 5 & L \\
2 & 0 & 5 & 0 & 1/4 & 0 & 1/2 & 0 & 0 & 1 & 0 & 1/2 & 7 & L \\
-1 & 1 & -2 & 0 & -1/2 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & L \\
1 & 1 & 1 & 1 & -1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & -12 &
\end{array}
$$

Figura 4.2: Exemplo de aplicação do Simplex. A figura registra os valores de F, G e E no início de cada iteração. O rótulo "L" indica as linhas que estão em L. A última matriz E é simples solúvel com relação à coluna 5 e linha 4.

$$
\begin{array}{ccccccccc}
3 & 2 & 1 & 4 & 5 & 6 & 7 & 8 & 10 \\
4 & 3 & 2 & 5 & 6 & 7 & -8 & -9 & 11 \\
5 & 6 & 3 & 4 & 7 & -9 & 8 & -10 & 9 \\
1 & -5 & 4 & 7 & 5 & -10 & 9 & 6 & 8 \\
9 & 8 & 7 & 11 & 3 & 2 & 1 & 4 & 0
\end{array}
$$

$$
\begin{array}{ccccc}
157/7473 & 334/7473 & -277/7473 & -170/7473 & 0 \\
466/7473 & -389/7473 & -13/7473 & 19/7473 & 0 \\
545/2491 & -332/2491 & 514/2491 & -368/2491 & 0 \\
-1343/7473 & 1522/7473 & -1486/7473 & 1597/7473 & 0 \\
7690/7473 & -12866/7473 & 12326/7473 & -13991/7473 & 1
\end{array}
$$

$$
\begin{array}{ccccccccc}
84/2491 & 168/2491 & -686/7473 & 0 & 0 & 1 & -1773/2491 & 0 & 1391/7473 \\
-68/2491 & -136/2491 & -275/7473 & 0 & 0 & 0 & 2147/2491 & 1 & 416/7473 \\
2509/2491 & 5018/2491 & -49/2491 & 0 & 1 & 0 & 7271/2491 & 0 & 3480/2491 \\
-1258/2491 & -5007/2491 & 3631/7473 & 1 & 0 & 0 & -6364/2491 & 0 & 2714/7473 \\
28834/2491 & 60159/2491 & 15283/7473 & 0 & 0 & 0 & 45640/2491 & 0 & -65620/7473
\end{array}
$$

Figura 4.3: Exemplo de aplicação do Simplex. Ao receber a matriz D (topo da figura) e os índices 9 (coluna) e 5 (linha) o Simplex executou 10 iterações e devolveu a matriz G (meio da figura). A matriz GD (parte inferior da figura) é simples solúvel com relação à coluna 9 e linha 5.

Fases. Qualquer execução da heurística consiste em uma seqüência de ocorrências da alternativa I seguida de uma seqüência de ocorrências da alternativa II. Enquanto estiver acontecendo a alternativa I, dizemos que a heurística está na **primeira fase**; enquanto estiver acontecendo a alternativa II, dizemos que a heurística está na **segunda fase**. A propósito, a alternativa II ocorre quando e somente quando $L = M - m$. fases

4.3 Análise

Para entender como e por que a heurística Simplex funciona, basta verificar as seguintes propriedades (compare com as seções 3.3, 2.4 e 2.5):

Invariantes *No início de cada iteração da heurística, existe uma parte P de L e uma parte Q de $N - n$ tais que* Q

(i0) $f_{[P]} \geq o$,
(i1) $E_{[P,Q]}$ *é de bijeção*, $E_{[M-P,Q]} = O$ e $E_{[L-P,\,]} = O$,
(i2) $F\,G = I$,
(i3) $G\,D = E$,
(i4) $G_{[\,,M-P]} = I_{[\,,M-P]}$,

onde f é o vetor $E_{[\,,n]}$.

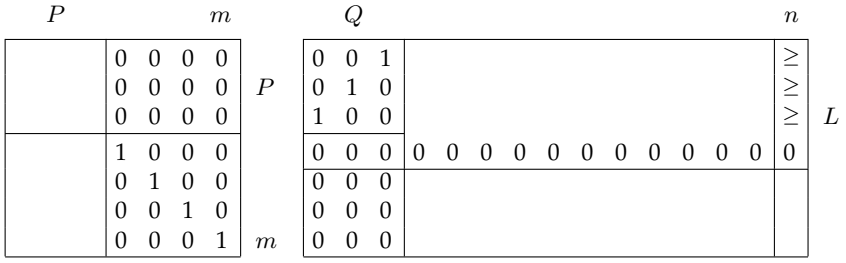

Figura 4.4: Matrizes G e E no início de uma iteração da heurística Simplex.

É evidente que esses invariantes valem (com P e Q vazios) no início da primeira iteração. Suponha agora que os invariantes valem no início de uma iteração qualquer que não a última. É claro que nessa iteração ocorre o caso I.1A, o caso I.1B, o caso I.3 ou o caso II.1B. É óbvio que depois de uma ocorrência do caso I.3 os invariantes permanecem válidos; para tratar dos demais casos, adote a notação

$$f' = E'_{[\,,n]} \quad \text{e} \quad P' = P + h\,.$$

Seja q é o único elemento de Q tal que $E_{[p,\,q]} = 1$. Basta mostrar agora que no fim do caso I.1A

$$f'_{[P']} \geq o\,, \tag{4.a}$$
$$E'_{[\,,Q]} = E_{[\,,Q]} \quad \text{e} \quad E'_{[\,,k]} = I_{[\,,h]}\,, \tag{4.b}$$
$$E'_{[L'-P',\,]} = E_{[L-P,\,]}\,, \tag{4.c}$$
$$G'_{[\,,M-P']} = G_{[\,,M-P']}\,, \tag{4.d}$$

que no fim dos casos I.1B e II.1B

$$f'_{[P]} \geq o\,, \tag{4.e}$$
$$E'_{[\,,Q-q]} = E_{[\,,Q-q]} \quad \text{e} \quad E'_{[\,,k]} = I_{[\,,q]}\,, \tag{4.f}$$
$$E'_{[L-P,\,]} = E_{[L-P,\,]}\,, \tag{4.g}$$
$$G'_{[\,,M-P]} = G_{[\,,M-P]}\,, \tag{4.h}$$

e que no fim de quaisquer dos casos

$$F'\,G' = I \quad \text{e} \quad G'\,D' = E'\,. \tag{4.i}$$

As propriedades (4.c) e (4.g) são triviais. As propriedades (4.a), (4.b), (4.e) e (4.f) já foram demonstradas na seção 3.3. As demonstrações das demais propriedades são análogas às que fizemos ao discutir o algoritmo de Gauss-Jordan.

Os invariantes valem, em particular, no início da última iteração. Portanto, na última iteração, a matriz E é simples inviável (caso I.2) ou simples ilimitada (caso II.1A) ou simples solúvel (caso II.2) com relação ao par n, m. Em qualquer desses casos, $G_{[\,,m]} = I_{[\,,m]}$ em virtude de (i4). Além disso, $FG = I$ em virtude de (i2). Assim, ao devolver F e G, a heurística está se comportando como prometeu.

4.4 Mais três invariantes

Os três invariantes que examinaremos a seguir (compare com a seção 2.5) parecem irrelevantes no presente capítulo, mas serão importantes para a análise do mecanismo de convergência que introduziremos no próximo capítulo (seção 5.5).

Invariantes *No início de cada iteração da heurística,*

(i5) $F_{[\,,M-P]} = I_{[\,,M-P]}$ e $F_{[\,,P]} = D_{[\,,Q]}\,\widetilde{J}$,
(i6) $D_{[M-P,\,Q]} = -G_{[M-P,\,P]}\,D_{[P,\,Q]}$,
(i7) $D_{[P,\,Q]}\,\widetilde{J}\,G_{[P,\,P]} = I$,

onde $\widetilde{J}$ é a transposta da matriz de bijeção $E_{[P,\,Q]}$. $\widetilde{J}$

A demonstração dessas propriedades é análoga à dos invariantes correspondentes do algoritmo de Gauss-Jordan.

O invariante (i5) descreve completamente a matriz F. Em particular, mostra que não é preciso atualizar F a cada iteração: a matriz pode ser calculada imediatamente antes do fim da execução da heurística. É claro que para isso é preciso dispor das variáveis P e Q.

O invariante (i6) mostra que, para cada i em $M - P$, o vetor $D_{[i,\,Q]}$ é uma combinação linear das linhas da matriz $D_{[P,\,Q]}$. O invariante (i7) mostra que $\widetilde{J}\,G_{[P,\,P]}$ é uma inversa direita de $D_{[P,\,Q]}$. A propósito, os invariantes (i3) e (i4) mostram que $\widetilde{J}\,G_{[P,\,P]}$ é uma inversa esquerda de $D_{[P,\,Q]}$.

Esses invariantes sugerem que em certas condições podemos tomar um pequeno atalho antes da primeira iteração. Suponha, por exemplo, que existe uma parte Q de $N - n$ tal que $D_{[M-m,\,Q]}$ é de bijeção e $D_{[M-m,\,n]} \geq o$ (mas $D_{[m,\,Q]}$ não é nula). Seja P o conjunto $M - m$ e G a matriz definida por

$$\begin{aligned} &G_{[P,\,P]} = I_{[P,\,P]}\,, \quad G_{[\,,\,M-P]} = I_{[\,,\,M-P]} \quad \text{e} \\ &G_{[M-P,\,P]} = -D_{[M-P,\,Q]}\,\widetilde{J}\,, \end{aligned}$$

onde $\widetilde{J}$ é a transposta de $D_{[P,\,Q]}$. Então a primeira iteração pode começar com $E = GD$ e $L = M - m$.

$$\begin{array}{cccccccc} 0 & 1 & 0 & 4 & 4 & 4 & 4 & 1 \\ 0 & 0 & 1 & 4 & 4 & 4 & 4 & 1 \\ 1 & 0 & 0 & 4 & 4 & 4 & 4 & 1 \\ 9 & 8 & 7 & 6 & 5 & 4 & 3 & 1 \end{array}$$

Figura 4.5: Seja D a matriz da figura. Adote $n = 8$ e $m = 4$. Calcule matrizes F_0 e G_0 que permitam começar a primeira iteração do Simplex com $L = M - m$, $F = F_0$, $G = G_0$ e $E = G_0 D$.

4.5 Convergência

A heurística Simplex nem sempre converge. A figura 4.6 dá um exemplo: o caso II.1B se aplica em todas as iterações; no início da sétima iteração, estamos exatamente na mesma situação que no início da primeira; este ciclo poderá se repetir *ad æternum*. A experiência sugere que tais exemplos de "ciclagem" são raros.[4]

É importante examinar o fenômeno mais de perto. É óbvio que nos casos I.1A e I.3 a heurística está fazendo progresso, pois L aumenta. Há algum

[4] Entretanto, há quem afirme [Sch86, p.138] que a ocorrência de ciclagem em problemas práticos é mais freqüente do que se imagina, mas é mascarada pelo efeito dos erros de arredondamento da aritmética ponto flutuante.

$$
\begin{array}{rrrrrrrr}
1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & -1/2 & -3/2 & 1/2 & 0 \\
0 & 0 & 1 & 9 & -5/2 & -11/2 & 1/2 & 0 \\
0 & 0 & 0 & 24 & 9 & 57 & -10 & 1
\end{array}
$$

$$
\begin{array}{rrrrrrrr}
1 & 0 & -2 & -18 & 5 & 11 & 0 & 1 \\
0 & 1 & -1 & -8 & 2 & 4 & 0 & 0 \\
0 & 0 & 2 & 18 & -5 & -11 & 1 & 0 \\
0 & 0 & 20 & 204 & -41 & -53 & 0 & 1
\end{array}
$$

$$
\begin{array}{rrrrrrrr}
1 & -11/4 & 3/4 & 4 & -1/2 & 0 & 0 & 1 \\
0 & 1/4 & -1/4 & -2 & 1/2 & 1 & 0 & 0 \\
0 & 11/4 & -3/4 & -4 & 1/2 & 0 & 1 & 0 \\
0 & 53/4 & 27/4 & 98 & -29/2 & 0 & 0 & 1
\end{array}
$$

$$
\begin{array}{rrrrrrrr}
1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & -5/2 & 1/2 & 2 & 0 & 1 & -1 & 0 \\
0 & 11/2 & -3/2 & -8 & 1 & 0 & 2 & 0 \\
0 & 93 & -15 & -18 & 0 & 0 & 29 & 1
\end{array}
$$

$$
\begin{array}{rrrrrrrr}
1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & -5/4 & 1/4 & 1 & 0 & 1/2 & -1/2 & 0 \\
0 & -9/2 & 1/2 & 0 & 1 & 4 & -2 & 0 \\
0 & 141/2 & -21/2 & 0 & 0 & 9 & 20 & 1
\end{array}
$$

$$
\begin{array}{rrrrrrrr}
1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & -1/2 & -3/2 & 1/2 & 0 \\
0 & -9 & 1 & 0 & 2 & 8 & -4 & 0 \\
0 & -24 & 0 & 0 & 21 & 93 & -22 & 1
\end{array}
$$

$$
\begin{array}{rrrrrrrr}
1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & -1/2 & -3/2 & 1/2 & 0 \\
0 & 0 & 1 & 9 & -5/2 & -11/2 & 1/2 & 0 \\
0 & 0 & 0 & 24 & 9 & 57 & -10 & 1
\end{array}
$$

Figura 4.6: Um exemplo [Chv83] de não-convergência da heurística Simplex. A figura registra a matriz E no início de sucessivas iterações. Em todas as iterações os elementos de L são $1, 2, 3$ e h é 4. No início da sétima iteração, estamos exatamente na mesma situação que no início da primeira.

progresso nos demais casos? Não é difícil verificar que no caso I.1B

$$0 < f'[h] \leq f[h] \quad \text{ou} \quad 0 > f'[h] \geq f[h] \tag{4.j}$$

e no caso II.1B

$$f'[h] \geq f[h]\,, \tag{4.k}$$

onde $f' = E'[\,, n]$. Demonstraremos uma generalização dessas desigualdades no próximo capítulo. Imagine agora que estamos diante de uma seqüência de ocorrências do caso I.1B ou do caso II.1B. Suponha, por um instante, que as desigualdades em (4.j) e (4.k) são estritas ao longo da execução da heurística. Então a seqüência de valores de $f[h]$ será estritamente monotônica, isto é, estritamente

crescente ou estritamente decrescente.[5] Nessas condições, cada iteração da seqüência começará com um valor diferente de Q, como mostraremos no próximo capítulo recorrendo aos invariantes (i6) e (i7). Como o número de possíveis valores de Q é finito, o número de iterações será finito, e portanto a heurística convergirá.

Em suma, a heurística Simplex converge se as desigualdades em (4.j) e (4.k) forem sempre estritas. Mas essa observação é um tanto vazia, porque não há como prever se a condição estará satisfeita em todas as iterações.

A heurística corre o risco de não convergir quando $f'_{[h]} = f_{[h]}$. É fácil verificar que isso acontece quando $f_{[p]} = 0$. Diz-se que matrizes com essa propriedade são **degeneradas**. É fácil verificar também que $f_{[p]} = 0$ quando, em alguma iteração anterior, houve empate na definição de p, ou seja, quando $f_{[p]}/E_{[p,k]}$ é mínimo para mais de um p em L^*. O capítulo seguinte introduzirá uma política para a escolha de p no caso de empates.

matriz degenerada

empate

4.6 Conclusão

A heurística Simplex recebe uma matriz D, um índice n de coluna e um índice m de linha e procura determinar uma matriz inversível G tal que $G_{[\,,m]} = I_{[\,,m]}$ e GD é simples com relação a n, m. A heurística nem sempre converge; mas quando converge, produz o resultado desejado. Esta situação pode colocar em dúvida a própria existência de uma matriz G com as propriedades desejadas.

4.7 Apêndice: Simplex Revisto

Cada iteração do Simplex usa apenas a linha h e as colunas k e n da matriz E. Podemos, portanto, tentar economizar algumas operações aritméticas eliminando a variável E e calculando os vetores

$$G_{[h,\,]}\,D\,, \quad GD_{[\,,k]} \quad \text{e} \quad GD_{[\,,n]}$$

apenas quando isso se fizer necessário. Esses ajustes levam a uma variante da heurística conhecida como Simplex Revisto. Vamos exibir abaixo uma versão do Simplex Revisto que se aplica a sistemas D, n, m dotadas da seguinte propriedade (veja exercício 4.4, página 48): existe uma parte Q_0 de $N - n$ tal que

$$\begin{aligned} &D_{[M-m,\,Q_0]} \text{ é de bijeção}\,,\\ &D_{[m,\,Q_0]} = o \quad \text{e} \quad D_{[M-m,\,n]} \geq o\,. \end{aligned}$$

É claro que essa versão corresponde à segunda fase da heurística geral. Cada iteração começa com uma parte Q de $N - n$, uma matriz G e um vetor f. A primeira iteração começa com $Q = Q_0$, $G = I$ e $f = D_{[\,,n]}$. Cada iteração consiste no seguinte:

[5] Uma seqüência $\alpha_1, \alpha_2, \alpha_3, \ldots$ é *estritamente* decrescente se $\alpha_1 > \alpha_2 > \alpha_3 > \cdots$.

CASO 1: $G_{[m,\,]}D_{[\,,k]} < 0$ *para algum* k *em* $N - n - Q$.
Seja a o vetor $GD_{[\,,k]}$.
Seja P^* o conjunto de todos os p em $M - m$ para os quais $a_{[p]} > 0$.

CASO 1A: P^* *é vazio* .
Devolva G e pare (a matriz GD é simples ilimitada).

CASO 1B: P^* *não é vazio* .
Escolha qualquer p em P^* tal que $f_{[p]}/a_{[p]}$ é mínimo.
Seja q o elemento de Q tal que $E_{[p,q]} = 1$.
Seja G', f' o resultado da pivotação de G, f em torno de p, k.
Comece nova iteração com $Q - q + k$, G', f' nos papéis de Q, G, f.

CASO 2: $G_{[m,\,]}D_{[\,,k]} \geq 0$ *para todo* k *em* $N - n - Q$.
Devolva G e pare (a matriz GD é simples solúvel). □

O resultado da pivotação do par G, f em torno de h, k é o par G', f' definido da maneira óbvia: $G'_{[h,\,]} = G_{[h,\,]}/a_{[h]}$, $f'_{[h]} = f_{[h]}/a_{[h]}$ e, para cada i em $M - h$,

$$G'_{[i,\,]} = G_{[i,\,]} - a_{[i]}\, G_{[h,\,]}/a_{[h]} \quad \text{e} \quad f'_{[i]} = f_{[i]} - a_{[i]}\, f_{[h]}/a_{[h]}\,.$$

No início de cada iteração, $f_{[M-m]} \geq o$, $(GD)_{[m,Q]}$ é nulo e $(GD)_{[M-m,Q]}$ é de bijeção. Ademais, G é inversível e $G_{[\,,m]} = I_{[\,,m]}$.

Tal como a versão básica, a versão revista da heurística pode não convergir. O número de operações aritméticas que a versão revista executa em cada iteração é, em geral, da mesma ordem que o mesmo número de operações da versão básica. Entretanto, se $|N|$ é muito maior que $|M|$ e se a matriz D é esparsa (isto é, tem poucos componentes não-nulos) então a versão revista faz menos operações aritméticas que a versão básica. Além disso, a versão revista é, em geral, menos sensível aos erros de arredondamento quando os números são representados em ponto flutuante.

Exercícios

4.1 Escreva uma versão da heurística Simplex que devolva, além de F e G, também a matriz E e, quando apropriado, as bases P e Q, o índice h de uma linha de inviabilidade e o índice k de uma coluna de ilimitação.

4.2 Escreva a heurística Simplex em uma linguagem mais formal, mais próxima de PASCAL ou C (veja apêndice E, página 192). Programe a heurística em um computador.

4.3 Escreva uma versão especializada da heurística Simplex para matrizes com apenas duas linhas. Mostre que essa versão sempre converge. (Solução no apêndice E.)

4.4 (SEGUNDA FASE DO SIMPLEX) Escreva uma versão especializada da heurística Simplex para o caso em que D tem a seguinte forma: existe uma

parte Q_0 de $N - n$ tal que

$$D_{[M-m,\,Q_0]} \text{ é de bijeção}\,,\ D_{[m,\,Q_0]} = o \ \text{ e } \ D_{[M-m,\,n]} \geq o\,.$$

É claro que o valor inicial de Q é Q_0. É claro que $P = M - m$ em todas as iterações.

4.5 (PRIMEIRA FASE DO SIMPLEX) Adote as seguintes definições, um pouco diferentes das que usamos no texto. Dada uma matriz E sobre $M \times N$ e um elemento n de N, diremos que E é **simples solúvel** se existem partes P de M e Q de $N - n$ tais que

$$E_{[P,\,Q]} \text{ é de bijeção}\,,\ E_{[M-P,\,]} = o \ \text{ e } \ E_{[\,,\,n]} \geq o\,.$$

Uma matriz E é **simples inviável** se existe um elemento h de M tal que $E_{[h,\,N-n]} \leq o$ e $E_{[h,\,n]}$ é positivo ou $E_{[h,\,N-n]} \geq o$ e $E_{[h,\,n]}$ é negativo.

Escreva uma heurística Simplex-Primeira-Fase que receba uma matriz E e um índice n e devolva matrizes F e G tais que $FG = I$ e a matriz GD é simples no sentido da nova definição.

4.6 (MUDANÇA DE BASE) Seja E uma matriz sobre $M \times N$ e sejam m e n elementos de M e N respectivamente. Suponha que E é simples solúvel com relação a n, m e tem base de linhas P e base de colunas Q. Suponha ainda que

$$\begin{aligned} &E_{[m,\,k]} = 0 \ \text{ para algum } k \text{ em } N - n - Q \ \text{ e} \\ &E_{[p,\,k]} > 0 \ \text{ para algum } p \text{ em } P\,. \end{aligned}$$

Calcule matrizes F e G tais que $FG = I$, $G_{[\,,\,m]} = I_{[\,,\,m]}$ e a matriz GE é simples solúvel com relação a n, m e tem base de colunas $Q - q + k$, onde q é um elemento de Q.

4.7 Deduza dos invariantes (i4), (i6) e (i7) que no início de cada iteração da heurística, para todo i em $M - P$, vale a identidade

$$E_{[i,\,]} = D_{[i,\,]} - D_{[i,\,Q]}\, \widetilde{E}_{[Q,\,P]}\, E_{[P,\,]}\,.$$

(Essa identidade é útil na análise da regra de Bland, a ser mencionada no próximo capítulo.)

4.8 Aplique a heurística Simplex à matriz abaixo [Bea55] com $n = 8$ e $m = 4$. Mostre que a heurística não converge quando os elementos de Q são $1, 6, 7$, depois $1, 2, 7$, depois $2, 3, 7$, depois $3, 4, 7$, depois $4, 5, 7$ e finalmente $5, 6, 7$.

$$\begin{array}{rrrrrrrr} 1/4 & -60 & -1/25 & 9 & 1 & 0 & 0 & 0 \\ 1/2 & -90 & -1/50 & 3 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ -3/4 & 150 & -1/50 & 6 & 0 & 0 & 0 & 0 \end{array}$$

Capítulo 5

Algoritmo Simplex

Para transformar a heurística Simplex (seção 4.2) num algoritmo é preciso introduzir um mecanismo que force a convergência (veja seção 4.5). O mecanismo que discutiremos neste capítulo é um refinamento da regra que a heurística usa, em cada iteração, para escolher a linha em torno da qual fará a próxima pivotação. (Outro mecanismo de convergência será mencionado na seção 5.9.)

5.1 Ordem lexicográfica

Seja N um conjunto de índices e Δ uma enumeração dos elementos de N, isto é, uma seqüência em que cada elemento de N aparece uma e uma só vez. (Em outras palavras, Δ é uma ordem total sobre N.) Para quaisquer vetores x e y sobre N, diremos que x é **lexicograficamente menor que** y em relação a Δ se existe k em N tal que

N
Δ

$$x[k] < y[k]$$

mas $x[j] = y[j]$ para todo j que precede k em Δ. Escreveremos

$\lessdot$

$$x \lessdot y$$

para dizer que x é lexicograficamente menor que y em relação a Δ. Assim, a enumeração Δ induz uma relação de **ordem lexicográfica** $\preceq$ no conjunto dos vetores sobre N:

$\preceq$

$$x \preceq y \quad \text{se} \quad x \lessdot y \text{ ou } x = y.$$

A relação $\succeq$ é definida da maneira óbvia: $x \succeq y$ se $y \preceq x$. Diremos que um vetor x é **lexicograficamente positivo** se $x \succ o$.

lexicograficamente positivo

É importante distinguir a desigualdade lexicográfica $x \preceq y$ da desigualdade usual $x \leq y$ (veja seção 1.1), que significa que $x[i] \leq y[i]$ para todo i. Se $x \leq y$ então $x \preceq y$, mas a recíproca não é verdadeira.

$$\begin{array}{rcrrrr} u & = & 0 & 11 & -99 & -88 \\ v & = & 0 & 0 & 0 & 0 \end{array} \qquad \begin{array}{rcrrrr} x & = & -88 & 33 & 44 & 11 \\ y & = & -88 & 33 & 33 & 99 \\ z & = & -88 & 33 & 44 & 12 \end{array}$$

Figura 5.1: Adote a enumeração $\langle 1, 2, 3, 4\rangle$ do conjunto de índices. O vetor u é lexicograficamente maior que v. O vetor x é lexicograficamente maior que y e lexicograficamente menor que z.

5.2 Regra lexicográfica

O capítulo anterior mostrou que a heurística Simplex corre o risco de não convergir sempre que houver mais de uma candidata para a linha de pivotação, ou seja, sempre que houver mais de um p em L^* tal que $E_{[p,n]}/E_{[p,k]}$ é mínimo (veja a seção 4.5). A regra lexicográfica que descreveremos a seguir força a convergência ao prescrever uma maneira de escolher uma das candidatas. Em virtude dessa regra, o algoritmo "anda" sempre numa mesma "direção", arbitrária mas fixa. A regra lexicográfica depende da presença de uma enumeração Δ de N cujo primeiro elemento é n e consiste no seguinte:

> Nos casos I.1B e II.1B da heurística Simplex, escolha p em L^* de modo que o vetor
>
> $$\frac{1}{E_{[p,k]}}\, E_{[p,\,]}$$
>
> seja lexicograficamente mínimo com relação à enumeração Δ. A enumeração não se altera durante as ocorrências dos casos I.1B e II.1B, mas é reajustada no fim de cada ocorrência do caso I.1A de modo que, para todo p em P, o vetor $E_{[p,\,]}$ seja lexicograficamente positivo.

Como n é o primeiro elemento de Δ, a escolha de p garante que o número $E_{[p,n]}/E_{[p,k]}$ será mínimo. Assim, a regra lexicográfica é um refinamento da regra que usamos na heurística do capítulo anterior. Como veremos adiante, a regra lexicográfica é satisfeita por um único p.

5.3 Algoritmo

Se acrescentarmos a regra lexicográfica à heurística Simplex (veja seção 4.2) obteremos o seguinte

> **Algoritmo Simplex Lexicográfico** *Recebe uma matriz D sobre $M \times N$ e elementos n e m de N e M respectivamente; devolve matrizes F e G sobre $M \times M$ tais que $FG = I$, $G_{[\,,m]} = I_{[\,,m]}$ e GD é simples com relação a n, m.* D n m

Cada iteração começa com uma enumeração Δ de N, uma parte L de $M-m$, um elemento h de $M-L$, uma parte P de L, uma parte Q de $N-n$ e matrizes F, G e E. Δ Q

Na primeira iteração, Δ é qualquer enumeração que comece com n. Ademais, $L = P = \emptyset$, $Q = \emptyset$, $F = G = I$, $E = D$ e h é um elemento de M, se possível distinto de m. Cada iteração consiste no seguinte, com $f = E_{[\,,n]}$: f

ALTERNATIVA I: *h é diferente de m.*

CASO I.1: $E_{[h,\,k]} > 0$ *e* $f_{[h]} \geq 0$ *ou* $E_{[h,\,k]} < 0$ *e* $f_{[h]} \leq 0$
para algum k em $N-n$. k
Seja L^* o conjunto de todos os p em L tais que $E_{[p,\,k]} > 0$. L^*

CASO I.1A: $f_{[h]}/E_{[h,\,k]} \leq f_{[p]}/E_{[p,\,k]}$ *para todo p em L^*.*
Defina Δ', L', h', F', G' e E' como indicado abaixo. Δ'
Comece nova iteração com Δ', L', h', $P+h$, $Q+k$, F', G', E'
nos papéis de Δ, L, h, P, Q, F, G, E.

CASO I.1B: $f_{[h]}/E_{[h,\,k]} > f_{[p]}/E_{[p,\,k]}$ *para algum p em L^*.*
Defina q, F', G' e E' como indicado abaixo.
Comece nova iteração com $Q-q+k$, F', G', E'
nos papéis de Q, F, G, E.

CASO I.2: $E_{[h,\,N-n]} \leq o$ *e* $f_{[h]} > 0$ *ou* $E_{[h,\,N-n]} \geq o$ *e* $f_{[h]} < 0$.
Devolva F e G e pare.

CASO I.3: $E_{[h,\,N-n]} = o$ *e* $f_{[h]} = 0$.
Seja L' o conjunto $L+h$.
Escolha h' em $M-L'$ se possível distinto de m.
Comece nova iteração com L' e h' nos papéis de L e h.

ALTERNATIVA II: *h é igual a m.*

CASO II.1: $E_{[h,\,k]} < 0$ *para algum k em $N-n$.* k
Seja L^* o conjunto de todos os p em L tais que $E_{[p,\,k]} > 0$. L^*

CASO II.1A: *L^* é vazio.*
Devolva F e G e pare.

CASO II.1B: *L^* não é vazio.*
Defina q, F', G' e E' como indicado abaixo.
Comece nova iteração com $Q-q+k$, F', G', E'
nos papéis de Q, F, G, E.

CASO II.2: $E_{[h,\,N-n]} \geq o$.
Devolva F e G e pare. □

Resta descrever os detalhes dos casos I.1A, I.1B e II.1B. O caso I.1A consiste no seguinte:

Seja F', G', E' o resultado da pivotação de F, G, E em torno de h, k.

Adote uma enumeração Δ' de N na qual n seja o primeiro elemento e os elementos de $Q + k$ venham logo a seguir (em qualquer ordem). Seja L' o conjunto $L + h$. Escolha h' em $M - L'$, se possível distinto de m. Δ'

(A operação de pivotação é definida exatamente como no capítulo anterior.) Os casos I.1B e II.1B são formalmente idênticos e consistem no seguinte:

Escolha p em L^* de modo que o vetor $E_{[p,\,]}/E_{[p,k]}$ seja lexicograficamente mínimo com relação à enumeração Δ. Seja q o elemento de Q tal que $E_{[p,q]} = 1$. Seja F', G', E' o resultado da pivotação de F, G, E em torno de p, k. p

A escolha lexicográfica de p deve ser entendida assim: escolha p em L^* de modo que

$$\frac{1}{E_{[p,k]}} E_{[p,\,]} \;\preceq\; \frac{1}{E_{[i,k]}} E_{[i,\,]} \tag{5.a}$$

para todo i em L^*. A propósito, L^* é parte de P, uma vez que a matriz $E_{[L-P,\,]}$ é nula em virtude do invariante (i1) abaixo.

5.4 Análise

O comportamento do algoritmo pode ser descrito pelas seguintes propriedades (compare com as seções 4.3 e 4.4):

Invariantes *No início de cada iteração do algoritmo,*

(i0) $E_{[i,\,]} \succcurlyeq o$ *para cada* i *em* P,[1]
(i1) $E_{[P,Q]}$ *é de bijeção,* $E_{[M-P,Q]} = O$ *e* $E_{[L-P,\,]} = O$,
(i2) $FG = I$,
(i3) $GD = E$,
(i4) $G_{[\,,M-P]} = I_{[\,,M-P]}$,
(i5) $F_{[\,,M-P]} = I_{[\,,M-P]}$ *e* $F_{[\,,P]} = D_{[\,,Q]}\,\widetilde{J}$,
(i6) $D_{[M-P,Q]} = -G_{[M-P,P]}\,D_{[P,Q]}$,
(i7) $D_{[P,Q]}\,\widetilde{J}\,G_{[P,P]} = I$,

onde $\widetilde{J}$ *é a transposta da matriz de bijeção* $E_{[P,Q]}$. $\widetilde{J}$

Os invariantes (i1) a (i7) são idênticos aos que demonstramos ao analisar a heurística Simplex. O invariante (i0) generaliza o invariante correspondente da heurística. Note que a desigualdade lexicográfica em (i0) é estrita; compare

[1] No entanto, a proposição "os elementos de Q precedem os demais elementos de $N - n$ na enumeração Δ" não é invariante!

b	c	d	e	f	g	h	a
1	0	0	0	0	0	1	1
0	1	0	1	$-1/2$	$-3/2$	$1/2$	0
0	0	1	9	$-5/2$	$-11/2$	$1/2$	0
0	0	0	24	9	57	-10	1

$$\tfrac{1}{E[3,7]}E[3,\,] \;\lessdot\; \tfrac{1}{E[2,7]}E[2,\,] \;\lessdot\; \tfrac{1}{E[1,7]}E[1,\,]$$

1	0	-2	-18	5	11	0	1
0	1	-1	-8	2	4	0	0
0	0	2	18	-5	-11	1	0
0	0	20	204	-41	-53	0	1

$$\tfrac{1}{E[2,6]}E[2,\,] \;\lessdot\; \tfrac{1}{E[1,6]}E[1,\,]$$

1	$-11/4$	$3/4$	4	$-1/2$	0	0	1
0	$1/4$	$-1/4$	-2	$1/2$	1	0	0
0	$11/4$	$-3/4$	-4	$1/2$	0	1	0
0	$53/4$	$27/4$	98	$-29/2$	0	0	1

$$\tfrac{1}{E[2,5]}E[2,\,] \;\lessdot\; \tfrac{1}{E[3,5]}E[3,\,]$$

1	$-5/2$	$1/2$	2	0	1	0	1
0	$1/2$	$-1/2$	-4	1	2	0	0
0	$5/2$	$-1/2$	-2	0	-1	1	0
0	$41/2$	$-1/2$	40	0	29	0	1

2	-5	1	4	0	2	0	2
1	-2	0	-2	1	3	0	1
1	0	0	0	0	0	1	1
1	18	0	42	0	30	0	2

Figura 5.2: Exemplo de aplicação do Simplex Lexicográfico (compare com a figura 4.6). Na primeira iteração, as linhas 1, 2 e 3 estão em L e em P e as colunas 1, 2 e 3 estão em Q. As letras que encabeçam as colunas, tomadas em ordem alfabética, indicam a enumeração de N (a mesma em todas as iterações). O caso II.1B ocorre em todas as iterações, exceto a última. Indicamos as desigualdades lexicográficas que ditam a escolha de p em cada iteração. A execução do algoritmo termina no caso II.2.

isso com nossa discussão na seção 4.5 sobre a não-convergência da heurística, especialmente no caso de sistemas degenerados.

É claro que o invariante (i0) vale no início da primeira iteração. Suponha agora que ele vale no início de uma iteração qualquer que não a última. Seja $\preceq'$ a ordem lexicográfica induzida por Δ'. É preciso mostrar então que no fim do caso I.1A teremos

$$E'_{[i,\,]} \succcurlyeq' o \;\text{ para cada } i \text{ em } P + h\,, \tag{5.b}$$

e que no fim dos casos I.1B e II.1B teremos

$$E'_{[i,\,]} \succcurlyeq o \;\text{ para cada } i \text{ em } P\,. \tag{5.c}$$

DEMONSTRAÇÃO DE (5.b): Já mostramos, ao analisar a heurística Simplex, que

$$E'_{[P+h,\,n]} = f'_{[P+h]} \geq o$$

e que $E'_{[P+h,\,Q+k]}$ é uma matriz de bijeção. Na enumeração Δ', o primeiro elemento é n e os elementos de $Q + k$ precedem todos demais elementos de N. Portanto, o vetor $E'_{[i,\,]}$ é lexicograficamente positivo. □

DEMONSTRAÇÃO DE (5.c): É preciso mostrar que, para todo i em P, o vetor $E'_{[i,\,]}$ é lexicograficamente positivo com relação à enumeração Δ. Considere inicialmente o caso $i = p$. Então

$$E'_{[p,\,]} = \frac{1}{E_{[p,\,k]}}\, E_{[p,\,]}\,.$$

Como $E_{[p,\,k]}$ é positivo e $E_{[p,\,]}$ é lexicograficamente positivo, $E'_{[p,\,]}$ também é lexicograficamente positivo. Suponha agora que i é diferente de p e $E_{[i,\,k]}$ não é positivo. Por definição,

$$E'_{[i,\,]} \;=\; E_{[i,\,]} - \frac{E_{[i,\,k]}}{E_{[p,\,k]}}\, E_{[p,\,]} \;=\; E_{[i,\,]} - E_{[i,\,k]}\, E'_{[p,\,]}\,.$$

Como $E_{[i,\,]}$ e $E'_{[p,\,]}$ são lexicograficamente positivos, também $E'_{[i,\,]}$ é lexicograficamente positivo. Suponha, finalmente, que i é diferente de p e $E_{[i,\,k]}$ é positivo. Por definição,

$$E'_{[i,\,]} \;=\; E_{[i,\,k]}\, \big(\frac{1}{E_{[i,\,k]}} E_{[i,\,]} - \frac{1}{E_{[p,\,k]}} E_{[p,\,]}\big)\,.$$

Portanto, $E'_{[i,\,]} \succeq o$ em virtude de (5.a). Ademais, a expressão entre parênteses não pode ser nula uma vez que i e p são elementos distintos de P e $E_{[P,\,Q]}$ é uma matriz de bijeção. Portanto,

$$E'_{[i,\,]} \succcurlyeq o\,,$$

como queríamos demonstrar. □

5.5 Convergência

Enquanto estiver ocorrendo o caso I.1A ou o caso I.3, o algoritmo está fazendo progresso, pois L aumenta. A situação é mais delicada quando ocorre o caso I.1B ou o caso II.1B. Em que sentido o algoritmo está fazendo progresso nesses casos? A resposta a esta pergunta está nas seguintes observações, que dependem crucialmente do invariante (i0).

Fato 5.1 *No fim de cada ocorrência do caso* I.1B*,*

$$o \lessdot E'_{[h,\,]} \lessdot E_{[h,\,]} \qquad \textit{ou} \qquad o \gtrdot E'_{[h,\,]} \gtrdot E_{[h,\,]}\,.$$

No fim de cada ocorrência do caso II.1B*,* $E'_{[h,\,]} \gtrdot E_{[h,\,]}$.

DEMONSTRAÇÃO: Por definição do caso I.1B, temos $f_{[h]}/E_{[h,k]} > f_{[p]}/E_{[p,k]}$. Logo,

$$\frac{1}{E_{[h,k]}} E_{[h,\,]} \gtrdot \frac{1}{E_{[p,k]}} E_{[p,\,]}\,.$$

Suponha $E_{[h,k]}$ é positivo. Então

$$E_{[h,\,]} \gtrdot \frac{E_{[h,k]}}{E_{[p,k]}}\, E_{[p,\,]}\,.$$

Por outro lado, como $E_{[p,k]}$ é positivo, o invariante (i0) garante que

$$\frac{E_{[h,k]}}{E_{[p,k]}}\, E_{[p,\,]} \gtrdot o\,.$$

Como

$$E'_{[h,\,]} = E_{[h,\,]} - \frac{E_{[h,k]}}{E_{[p,k]}}\, E_{[p,\,]}\,,$$

concluímos que $o \lessdot E'_{[h,\,]} \lessdot E_{[h,\,]}$. Se $E_{[h,k]}$ for negativo, um raciocício análogo mostra que $o \gtrdot E'_{[h,\,]} \gtrdot E_{[h,\,]}$. Finalmente, considere o caso II.1B. Como $E_{[h,k]}$ é negativo, $E_{[p,k]}$ é positivo e

$$E'_{[h,\,]} \;=\; E_{[h,\,]} - \frac{E_{[h,k]}}{E_{[p,k]}}\, E_{[p,\,]}\,,$$

o invariante (i0) aplicado ao vetor $E_{[p,\,]}$ garante que $E'_{[h,\,]} \gtrdot E_{[h,\,]}$. □

Podemos mostrar agora por que o algoritmo converge. Imagine que estamos diante de uma seqüência de iterações em que ocorrem somente o caso I.1B ou somente o caso II.1B, donde L e P permanecem constantes. Sejam $\dot{Q}$ e $\dot{E}$ os valores das variáveis Q e E no início da primeira das iterações da seqüência e defina $\ddot{Q}$ e $\ddot{E}$ analogamente para a última das iterações da seqüência. Como a enumeração Δ não se altera ao longo da seqüência de iterações em questão, o fato 5.1 garante que

$$\dot{E}_{[h,\,]} \lessdot \ddot{E}_{[h,\,]} \quad \text{ou} \quad \dot{E}_{[h,\,]} \gtrdot \ddot{E}_{[h,\,]}\,,$$

donde $\dot{E}_{[h,\,]} \neq \ddot{E}_{[h,\,]}$. Nessas condições, o lema abaixo garante que $\dot{Q} \neq \ddot{Q}$. Como o número de subconjuntos de N é finito, nossa seqüência de iterações é necessariamente finita.

Lema 5.2 (da repetição de bases) *Se $\dot{Q} = \ddot{Q}$ então $\dot{E}_{[h,\,]} = \ddot{E}_{[h,\,]}$.*

DEMONSTRAÇÃO: Sejam $\dot{G}$ e $\ddot{G}$ os valores da variável G no início das duas iterações em questão. Seja Q o valor comum de $\dot{Q}$ e $\ddot{Q}$. Em virtude do invariante (i6),

$$\dot{G}_{[h,\,P]}\, D_{[P,\,Q]} = \ddot{G}_{[h,\,P]}\, D_{[P,\,Q]}\,,$$

pois ambas as expressões são iguais a $-D_{[h,\,Q]}$. Em virtude do invariante (i7), $D_{[P,\,Q]}$ tem uma inversa direita:

$$D_{[P,\,Q]}\, \widetilde{J}\, \dot{G}_{[P,\,P]} = I\,,$$

onde $\widetilde{J}$ é a transposta da matriz de bijeção $\dot{E}_{[P,\,Q]}$. Logo, $\dot{G}_{[h,\,P]} = \ddot{G}_{[h,\,P]}$. Em virtude de (i4),

$$\dot{G}_{[h,\,]} = \ddot{G}_{[h,\,]}\,.$$

Finalmente, em virtude de (i3), $\dot{E}_{[h,\,]} = \dot{G}_{[h,\,]}\, D = \ddot{G}_{[h,\,]}\, D = \ddot{E}_{[h,\,]}$. □

Em suma, se L e P permanecem constantes em alguma seqüência de iterações então cada iteração começa com um valor diferente de Q. Assim, o comprimento da seqüência não excede o número de subconjuntos de $N-n$ que têm $|P|$ elementos. Segue daí que o número total de iterações que o algoritmo executa é limitado pelo número total de subconjuntos de $N - n$, ou seja, por

$$2^{|N|-1}\,.$$

Se ajustarmos a notação de modo que N seja o conjunto $\{1, 2, \ldots, n\}$, teremos $n = |N|$ e poderemos dizer que o número total de iterações do algoritmo é limitado por

$$2^{n-1}\,.$$

Essa delimitação sugere que o número de iterações pode crescer explosivamente em função de n. (Por exemplo, se somarmos 10 a n o valor de 2^{n-1} será multiplicado por mais de 1000.) Mas a experiência mostra que, em geral, o número de iterações fica bem abaixo desse limite; o estudo deste fenômeno é muito interessante (veja Borgwardt [Bor87]), mas não cabe aqui.

O algoritmo pode, de fato, executar cerca de 2^{n-1} iterações. Klee e Minty [KM72], e também Avis e Chvátal [AC78], construíram uma família de exemplos com n colunas e $(n+1)/2$ linhas que forçam o algoritmo a executar cerca de $2^{(n-1)/2}$ iterações. Reproduzimos nas figuras 5.3 a 5.5 o resultado da aplicação do algoritmo ao membro $n = 9$ da família.

i	h	g	f	e	d	c	b	a	
1	1	0	0	0	0	0	0	5	
4	0	1	1	0	0	0	0	25	
8	0	4	0	1	1	0	0	125	
16	0	8	0	4	0	1	1	625	
8	0	4	0	2	0	1	0	0	

b	c	d	e	f	g	h	i	a	
1	1	0	0	0	0	0	0	5	L
0	−4	1	1	0	0	0	0	5	
0	−8	4	0	1	1	0	0	85	
0	−16	8	0	4	0	1	1	545	
0	−8	4	0	2	0	1	0	−40	

b	d	c	e	f	g	h	i	a	
1	1	0	0	0	0	0	0	5	L
0	−4	1	1	0	0	0	0	5	L
0	8	0	−4	1	1	0	0	65	
0	16	0	−8	4	0	1	1	505	
0	8	0	−4	2	0	1	0	−60	

1	1	0	0	0	0	0	0	5	L
4	0	1	1	0	0	0	0	25	L
−8	0	0	−4	1	1	0	0	25	
−16	0	0	−8	4	0	1	1	425	
−8	0	0	−4	2	0	1	0	−100	

e	b	c	f	d	g	h	i	a	
1	1	0	0	0	0	0	0	5	L
4	0	1	1	0	0	0	0	25	L
−8	0	0	−4	1	1	0	0	25	L
16	0	0	8	0	−4	1	1	325	
8	0	0	4	0	−2	1	0	−150	

1	1	0	0	0	0	0	0	5	L
0	−4	1	1	0	0	0	0	5	L
0	8	0	−4	1	1	0	0	65	L
0	−16	0	8	0	−4	1	1	245	
0	−8	0	4	0	−2	1	0	−190	

1	1	0	0	0	0	0	0	5	L
0	−4	1	1	0	0	0	0	5	L
0	−8	4	0	1	1	0	0	85	L
0	16	−8	0	0	−4	1	1	205	
0	8	−4	0	0	−2	1	0	−210	

Figura 5.3: Continua na próxima figura.

1	1	0	0	0	0	0	0	5	L
4	0	1	1	0	0	0	0	25	L
8	0	4	0	1	1	0	0	125	L
−16	0	−8	0	0	−4	1	1	125	
−8	0	−4	0	0	−2	1	0	−250	

f	*b*	*g*	*c*	*d*	*h*	*e*	*i*	*a*	
1	1	0	0	0	0	0	0	5	L
4	0	1	1	0	0	0	0	25	L
8	0	4	0	1	1	0	0	125	L
−16	0	−8	0	0	−4	1	1	125	L
8	0	4	0	0	2	0	−1	−375	

1	1	0	0	0	0	0	0	5	L
4	0	1	1	0	0	0	0	25	L
8	0	4	0	1	1	0	0	125	L
−16	0	−8	0	0	−4	1	1	125	L
−8	0	−4	0	0	−2	1	0	−250	

1	1	0	0	0	0	0	0	5	L
0	−4	1	1	0	0	0	0	5	L
0	−8	4	0	1	1	0	0	85	L
0	16	−8	0	0	−4	1	1	205	L
0	8	−4	0	0	−2	1	0	−210	

1	1	0	0	0	0	0	0	5	L
0	−4	1	1	0	0	0	0	5	L
0	8	0	−4	1	1	0	0	65	L
0	−16	0	8	0	−4	1	1	245	L
0	−8	0	4	0	−2	1	0	−190	

1	1	0	0	0	0	0	0	5	L
4	0	1	1	0	0	0	0	25	L
−8	0	0	−4	1	1	0	0	25	L
16	0	0	8	0	−4	1	1	325	L
8	0	0	4	0	−2	1	0	−150	

1	1	0	0	0	0	0	0	5	L
4	0	1	1	0	0	0	0	25	L
−8	0	0	−4	1	1	0	0	25	L
−16	0	0	−8	4	0	1	1	425	L
−8	0	0	−4	2	0	1	0	−100	

1	1	0	0	0	0	0	0	5	L
0	−4	1	1	0	0	0	0	5	L
0	8	0	−4	1	1	0	0	65	L
0	16	0	−8	4	0	1	1	505	L
0	8	0	−4	2	0	1	0	−60	

Figura 5.4: Continua na próxima figura.

$$
\begin{array}{rrrrrrrrrl}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & L \\
0 & -4 & 1 & 1 & 0 & 0 & 0 & 0 & 5 & L \\
0 & -8 & 4 & 0 & 1 & 1 & 0 & 0 & 85 & L \\
0 & -16 & 8 & 0 & 4 & 0 & 1 & 1 & 545 & L \\
0 & -8 & 4 & 0 & 2 & 0 & 1 & 0 & -40 &
\end{array}
$$

$$
\begin{array}{rrrrrrrrrl}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & L \\
4 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 25 & L \\
8 & 0 & 4 & 0 & 1 & 1 & 0 & 0 & 125 & L \\
16 & 0 & 8 & 0 & 4 & 0 & 1 & 1 & 625 & L \\
8 & 0 & 4 & 0 & 2 & 0 & 1 & 0 & 0 &
\end{array}
$$

Figura 5.5: Conclusão das duas figuras anteriores. As figuras mostram a aplicação do algoritmo Simplex Lexicográfico a um sistema com 9 colunas. O algoritmo executa $2^{(9-1)/2}$ iterações. As figuras registram o valor de E no início de cada iteração. O rótulo "L" indica as linhas que estão em L. A enumeração de N está indicada pelas letras que encabeçam as colunas, tomadas em ordem alfabética.

5.6 Número de operações aritméticas

Mesmo sabendo que o número de iterações do algoritmo cresce exponencialmente com o número de componentes da matriz, é interessante estimar o número de operações aritméticas executadas durante uma iteração.

Ajuste a notação de modo que M e N sejam os conjuntos $\{1, 2, \ldots, m\}$ e $\{1, 2, \ldots, n\}$ respectivamente. Assim poderemos dizer que m é o número de linhas e n o número de colunas da matriz D.

Para decidir que caso se aplica e determinar a linha e coluna em torno da qual fará a próxima pivotação, cada iteração do algoritmo Simplex executa no máximo mn divisões e menos que mn subtrações.[2] (Na verdade, é mais justo trocar "mn" por "m^2", uma vez que o resultado das comparações lexicográficas só depende dos $|Q| + 1$ primeiras colunas na enumeração Δ.) Por outro lado, cada pivotação requer não mais que mn multiplicações e divisões e menos que mn adições e subtrações, como já observamos na seção 2.6. Portanto, o número total de operações aritméticas em cada iteração é menor que

$$4mn\,.$$

(A título de comparação, a multiplicação de G por D requer m^2n multiplicações e outras tantas adições.)

Qual o custo de uma operação aritmética? O universo natural do Simplex é o dos números racionais: se cada componente da matriz D é um número racional então todos os números gerados pelo algoritmo serão racionais, uma vez que o algoritmo só envolve as quatro operações aritméticas e portanto transforma

[2] As subtrações estão implícitas nas comparações lexicográficas.

números racionais em outros números racionais. O custo de uma operação aritmética sobre números racionais depende da magnitude dos numeradores e denominadores dos números envolvidos. Para estimar esse custo será necessário, portanto, obter uma delimitação superior para os numeradores e denominadores gerados pelo algoritmo. Faremos isto no capítulo 12. Podemos adiantar que esses números são, em geral, muito maiores que os numeradores e denominadores dos componentes da matriz dada D.

5.7 Conclusão

O algoritmo Simplex recebe uma matriz D, o índice n de uma coluna e o índice m de uma linha e devolve uma matriz inversível G tal que $G_{[\,,m]} = I_{[\,,m]}$ e a matriz GD é simples com relação a n, m.

O algoritmo é importante não só do ponto de vista computacional mas também do ponto de vista conceitual: ele prova a *existência* de um matriz G com as propriedades enunciadas.

O algoritmo executa cerca de 2^{n-1} iterações no pior caso, onde n é o número de colunas da matriz D, mas em geral converge bem antes.

É claro que o algoritmo Simplex pode ser executado de modo aproximado, com aritmética de ponto flutuante. Uma tal implementação aproximada pode cometer erros arbitrariamente grandes. Há um grande repertório de truques [Chv83] que procuram reduzir tais erros.

5.8 Apêndice: Segunda fase do Simplex

Pode ser instrutivo examinar a versão especializada do algoritmo Simplex Lexicográfico que se aplica a matrizes D que têm a seguinte forma: existe uma parte Q_0 de $N - n$ tal que

$$D_{[M-m,\,Q_0]} \text{ é de bijeção}\,,\ D_{[M-m,\,n]} \geq o \quad \text{e} \quad D_{[m,\,Q_0]} = o\,.$$

É claro que essa versão do algoritmo corresponde à segunda fase do Simplex Adote uma enumeração Δ de N na qual n seja o primeiro elemento e os elementos de Q_0 precedam os elementos de $N - n - Q_0$. Cada iteração começa com matrizes F, G e E. A primeira iteração começa com $F = G = I$ e $E = D$. Cada iteração consiste no seguinte:

CASO 1: $E_{[m,\,k]} < 0$ *para algum* k *em* $N - n$ *.*
Seja P^* o conjunto dos p em $M - m$ para os quais $E_{[p,\,k]} > 0$.

CASO 1A: P^* *é vazio .*
Devolva F e G e pare (E é simples ilimitada).

CASO 1B: P^* *não é vazio .*
Escolha qualquer p em P^* tal que o vetor $E_{[p,\,]}/E_{[p,\,k]}$
seja lexicograficamente mínimo em relação à enumeração Δ.

Seja F', G', E' o resultado da pivotação de F, G, E em torno de p, k.
Comece nova iteração com F', G' e E' nos papéis de F, G e E.

CASO 2: $E_{[m,\,N-n]} \geq o$.
Devolva F e G e pare (E é simples solúvel). □

No início de cada iteração, para cada i em $M - m$, o vetor $E_{[i,\,]}$ é lexicograficamente positivo com relação à enumeração Δ. Ademais, existe uma parte Q de N tal que $E_{[M-m,\,Q]}$ é uma matriz de bijeção e o vetor $E_{[m,\,Q]}$ é nulo.

O algoritmo converge porque duas iterações diferentes jamais começam com o mesmo valor de Q. Portanto, o número de iterações é limitado pelo número de subconjuntos de $N - n$ que têm $|M - m|$ elementos.

5.9 Apêndice: Regra de Bland

O algoritmo Simplex Lexicográfico que discutimos na seção 5.3 é baseado numa regra que escolhe a linha em torno da qual será executada a próxima pivotação. Uma outra regra, igualmente eficaz, foi demonstrada por Bland [Bla77]. A regra de Bland consiste no seguinte:

Adote uma enumeração arbitrária sobre o conjunto $N - n$;
essa enumeração permanece fixa ao longo da execução do algoritmo.
Escolha sempre *o menor k dentre os que podem entrar na base* e
o menor q dentre os que podem sair da base.

Em particular, escolha sempre o menor k que satisfaz a definição dos casos I.1 e II.1 da heurística Simplex. Nos casos I.1B e II.1B, escolha p em L^* de tal modo que, para todo i em $L^* - p$,

$$\begin{aligned} & f_{[p]}/E_{[p,\,k]} < f_{[i]}/E_{[i,\,k]} \\ \text{ou} \quad & f_{[p]}/E_{[p,\,k]} = f_{[i]}/E_{[i,\,k]} \ \text{ mas } \ q_p < q_i \,, \end{aligned}$$

onde q_i é o elemento de Q tal que $E_{[i,\,q_i]} = 1$. A prova de que a regra de Bland força a convergência do Simplex é baseada na seguinte observação:

durante uma seqüência de ocorrências dos casos I.1B ou II.1B, um elemento k de $N - n$ que tenha entrado na base Q só sai da base depois que um elemento maior que k tiver entrado na base.

A demonstração dessa observação é um tanto complexa e será omitida. O algoritmo Simplex com regra de Bland executa cerca de 2^{n-1} iterações no pior caso.

Exercícios

5.1 Verifique que a ordem lexicográfica entre vetores sobre N relativa a qualquer enumeração de N é reflexiva ($x \preceq x$), transitiva (se $x \preceq y$ e $y \preceq z$ então $x \preceq z$) e antisimétrica (se $x \preceq y$ e $y \preceq x$ então $x = y$).

5.2 Suponha que $y \succcurlyeq o$ em relação a uma dada enumeração do conjunto de índices. Mostre que $x - y \lessdot x$ para todo vetor x.

5.3 Seja $a, b, c, \ldots$ uma seqüência de vetores, todos indexados por $1, 2, 3$. Suponha que os componentes de cada vetor são inteiros entre 0 e 9. Adote a enumeração usual $\langle 1, 2, 3\rangle$ e suponha que a seqüência de vetores é estritamente decrescente no sentido lexicográfico (isto é, $a \succcurlyeq b \succcurlyeq c \succcurlyeq \ldots$). Que comprimento pode ter a seqüência?

5.4 Mostre que duas iterações diferentes do algoritmo podem começar com o mesmo valor de Q e valores diferentes de G.

5.5 Escreva o algoritmo Simplex Lexicográfico em uma linguagem mais formal, mais próxima de PASCAL ou C. Programe o algoritmo em um computador.

5.6 Escreva o algoritmo Simplex com regra de Bland. Programe o algoritmo em um computador.

5.7 Sejam n e m as cardinalidades de N e M respectivamente. Mostre que o número total de iterações do algoritmo Simplex Lexicográfico é limitado pela soma

$$\binom{n-1}{0} + \cdots + \binom{n-1}{m-1}.$$

Capítulo 6

Forma tradicional do Simplex

$A\,,\ b\,,\ c$
$GA\,,\ Gb\,,\ c + gA$

O algoritmo Simplex discutido nos capítulos anteriores opera sobre uma matriz em que uma determinada coluna e uma determinada linha foram destacadas para receber tratamento especial. Para aplicar o algoritmo ao problema de programação linear (de que trataremos no capítulo 7), é conveniente separar a coluna especial e a linha especial da matriz, e tratar esses dois objetos como vetores. Em outras palavras, é conveniente transformar ternos da forma D, n, m — em que D é uma matriz e n e m são índices — em ternos da forma A, b, c — em que A é uma matriz e b e c são vetores. A reformulação é meramente notacional, mas vale a pena discutí-la com algum vagar. É o que faremos neste capítulo.

6.1 Sistemas matriz-vetor-vetor

Um **sistema** sobre $M \times N$ é um terno A, b, c que consiste em uma matriz A sobre $M \times N$, um vetor b sobre M e um vetor c sobre N.

Se D é uma matriz sobre $\bar{M} \times \bar{N}$ com coluna especial $\bar{n}$ e linha especial $\bar{m}$ (veja definição na seção 3.1) então o correspondente sistema A, b, c é definido por

$$A = D_{[\bar{M}-\bar{m},\, \bar{N}-\bar{n}]}\,, \quad b = D_{[\bar{M}-\bar{m},\, \bar{n}]}\,, \quad c = D_{[\bar{m},\, \bar{N}-\bar{n}]}\,. \tag{6.a}$$

Ou seja, b é a última coluna de D, c é a última linha de D e A é o restante de D. É óbvio que A, b, c é um sistema sobre $(\bar{M}-\bar{m}) \times (\bar{N}-\bar{n})$.

Reciprocamente, se A, b, c é um sistema sobre $M \times N$ então a correspondente matriz D resulta da justaposição de A, b e c:

$$D_{[M,\, N]} = A\,, \quad D_{[M,\, \bar{n}]} = b\,, \quad D_{[\bar{m},\, N]} = c\,, \tag{6.b}$$

sendo $\bar{m}$ um objeto que não está em M e $\bar{n}$ um objeto que não está em N. O valor de $D_{[\bar{m},\, \bar{n}]}$ é arbitrário e irrelevante. É claro que D é uma matriz sobre $(M+\bar{m}) \times (N+\bar{n})$, sendo $\bar{m}$ sua linha especial e $\bar{n}$ sua coluna especial.

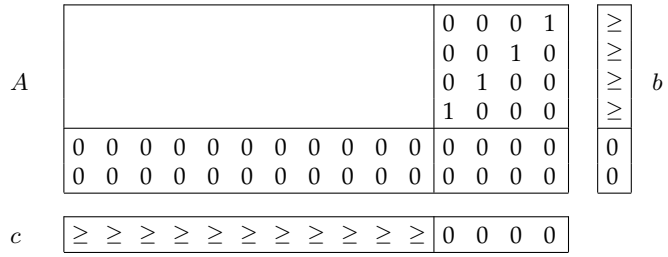

Figura 6.1: Um sistema simples solúvel. Compare com a figura 3.1.

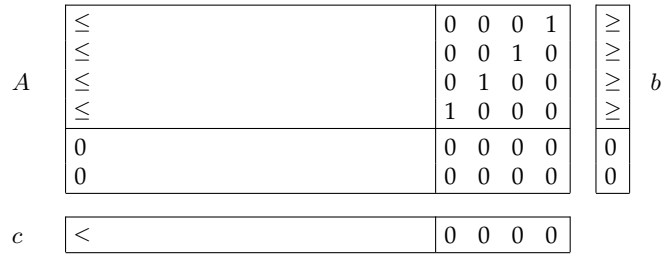

Figura 6.2: Um sistema simples ilimitado. Compare com a figura 3.3. Deduza da figura 3.2 a aparência de um sistema simples inviável.

6.2 Sistemas simples

Um sistema A, b, c é **simples** se a matriz D que resulta da justaposição de A, b e c for simples (veja seção 3.1) com relação ao par de índices que corresponde a b, c. Em particular, A, b, c é **simples solúvel**, **simples inviável** ou **simples ilimitado** se a matriz D for simples solúvel, simples inviável ou simples ilimitada, respectivamente. Os conceitos de **base**, **linha de inviabilidade** e **coluna de ilimitação** de um sistema simples A, b, c são as adaptações óbvias dos correspondentes conceitos para matriz simples (veja seção 3.1).

6.3 Algoritmo Simplex

O algoritmo Simplex pode ser apresentado agora da seguinte maneira:

Algoritmo Simplex (forma tradicional) *Recebe um sistema A, b, c sobre $M \times N$ e devolve matrizes F e G sobre $M \times M$ e um vetor g sobre M tais que $FG = I$ e o sistema* g

$$GA\,, Gb\,, c + gA$$

é simples (solúvel, inviável ou ilimitado).

$$\begin{array}{rrrrrr} -1 & 1 & 1 & 0 & 0 & 2 \\ 1 & 2 & 0 & 1 & 0 & 10 \\ 3 & -2 & 0 & 0 & 1 & 6 \\ -3 & 0 & 1 & 0 & 0 & \end{array}$$

$$\begin{array}{rrr} 1 & 1 & -1 \\ 0 & 2 & 1 \\ 0 & -2 & 3 \end{array} \qquad \begin{array}{rrr} 1 & -1/8 & 3/8 \\ 0 & 3/8 & -1/8 \\ 0 & 1/4 & 1/4 \\ -1 & 7/8 & 3/8 \end{array} \qquad \begin{array}{rrrrrr} 0 & 0 & 1 & -1/8 & 3/8 & 3 \\ 0 & 1 & 0 & 3/8 & -1/8 & 3 \\ 1 & 0 & 0 & 1/4 & 1/4 & 4 \\ 0 & 0 & 0 & 7/8 & 3/8 & \end{array}$$

Figura 6.3: Digamos que A, b, c é o sistema representado no topo da figura. Sejam F, G e g os objetos definidos na parte inferior esquerda da figura (g está representado sob G). Verifique que $FG = I$. Verifique que o sistema $GA, Gb, c + gA$ na parte inferior direita da figura é simples solúvel.

O algoritmo consiste no seguinte. Construa a justaposição D de A, b e c descrita em (6.b). Seja $\bar{n}$ a coluna especial e $\bar{m}$ a linha especial de D. Submeta $D, \bar{n}, \bar{m}$ ao algoritmo Simplex (veja seção 5.3), que devolverá matrizes $\bar{F}$ e $\bar{G}$ tais que $\bar{F}\bar{G} = I$, $\bar{G}_{[\,,\bar{m}]} = I_{[\,,\bar{m}]}$ e $\bar{G}D$ é simples. Defina F, G e g da seguinte maneira: *F G g*

$$F = \bar{F}_{[M,\,M]}\,, \quad G = \bar{G}_{[M,\,M]} \quad \text{e} \quad g = \bar{G}_{[\bar{m},\,M]}\,. \tag{6.c}$$

Devolva F, G e g e pare.

É claro que $FG = I$. É claro também que $\bar{G}D$ é o resultado da justaposição de GA, Gb e $c + gA$. Como $\bar{G}D$ é simples, o sistema $GA, Gb, c+gA$ é simples. Portanto, o algoritmo tem o comportamento que prometemos. □

Como já observamos na seção 5.5, o número de iterações do algoritmo é limitado por 2^n, sendo $n = |N|$. Como já observamos na seção 5.6, cada iteração executa menos que $4(m + 1)(n + 1)$ operações aritméticas, onde $m = |M|$.

Se os componentes do sistema A, b, c são números racionais então F, G e g também terão componentes racionais, uma vez que o algoritmo só envolve as quatro operações aritméticas.

6.4 Invariantes

A título de curiosidade, eis os invariantes do Simplex (seções 4.3 e 4.4) traduzidos para a nova notação: no início de cada iteração existem partes P e Q de M e N respectivamente, matrizes F e G, um vetor g, e um sistema A', b', c' tais que

(i0) $b'_{[P]} \geq o$,

(i1) $A'_{[P,\,Q]}$ é de bijeção, $A'_{[M-P,\,Q]} = O$ e $c'_{[Q]} = o$,

(i2) $FG = I$,

(i3) $A' = GA,\ b' = Gb$ e $c' = c + gA$,

(i4) $G_{[\,,\,M-P]} = I_{[\,,\,M-P]}$ e $g_{[M-P]} = o$,

(i5) $F_{[\,,\,M-P]} = I_{[\,,\,M-P]}$ e $F_{[\,,\,P]} = A_{[\,,\,Q]}\,\widetilde{J}$,

(i6) $A_{[M-P,\,Q]} = -G_{[M-P,\,P]}\,A_{[P,\,Q]}$ e $c_{[Q]} = -g_{[P]}\,A_{[P,\,Q]}$,

(i7) $A_{[P,\,Q]}\,\widetilde{J}\,G_{[P,\,P]} = I$,

onde $\widetilde{J}$ é a transposta da matriz de bijeção $A'_{[P,\,Q]}$. O vetor g que o algoritmo devolve é redundante: ele poderia ser calculado a partir de A, c, G e A'. De fato,

$$g_{[P]} = -c_{[Q]}\widetilde{J}G_{[P,\,P]} \tag{6.d}$$

de acordo com (i6) e (i7) e $g_{[M-P]} = o$ de acordo com (i4).

6.5 Conclusão

O algoritmo Simplex transforma qualquer sistema A, b, c em um sistema equivalente $GA, Gb, c + gA$ que é simples (solúvel, inviável ou ilimitado).

A seguinte questão conceitual certamente terá ocorrido ao leitor atento: um dado sistema A, b, c pode ser transformado em sistemas simples de tipos diferentes? Por exemplo, é possível que uma execução do Simplex produza um sistema simples solúvel e uma outra execução, com os mesmos dados, produza um sistema simples ilimitado? A resposta é *não*. Poderíamos demonstrar esta propriedade de consistência já, mas é mais confortável tratar disso no capítulo 7 (seção 7.4).

Exercícios

6.1 Seja A, b, c o sistema descrito abaixo. Exiba matrizes F e G e um vetor g tais que $FG = I$ e o sistema $GA, Gb, c + gA$ é simples.

$$\begin{array}{rrrrrrrr}
1 & 2 & 3 & -1 & 1 & 0 & 0 & 10 \\
2 & 3 & -1 & 1 & 0 & 1 & 0 & 7 \\
1 & 1 & 3 & 2 & 0 & 0 & 1 & 8 \\
\\
-3 & -1 & -5 & -2 & 0 & 0 & 0 &
\end{array}$$

Parte II

Programação Linear

Capítulo 7

Problema canônico primal

O algoritmo Simplex (veja capítulo 6) foi criado para resolver o problema canônico de programação linear, que discutiremos neste capítulo. O conceito de matriz simples e os termos *solúvel*, *inviável* e *ilimitado* que usamos ao discutir o Simplex (veja seção 3.1), encontrão aqui sua justificativa.

A heurística Simplex e a disciplina de programação linear foram criados por George Dantzig[Dan63] e Leonid Kantorovich[Kan39] durante a II Guerra Mundial (entre 1935 e 1945 aproximadamente). O Simplex foi se tornando cada vez mais popular desde então.[1]

7.1 Definição do problema

O **problema canônico primal** de programação linear consiste no seguinte:

> **Problema** $\mathsf{CP}(A, b, c)$: *Dada uma matriz A, um vetor b e um vetor c, encontrar um vetor $x \geq o$ tal que $Ax = b$ e cx é mínimo.*

A equação $Ax = b$ e a inequação $x \geq o$ são as **restrições** do problema. Poderíamos enunciar o problema dizendo: *minimize cx sob as restrições $Ax = b$ e $x \geq o$*. O conjunto de todos os vetores x que satisfazem as restrições é conhecido como **poliedro canônico primal** e será denotado aqui por

restrições

$$X(A, b)\,.$$

Se o número cx for interpretado como *custo* do vetor x, o problema canônico primal consiste em encontrar um vetor de custo mínimo em $X(A, b)$.

Problemas solúveis, inviáveis e ilimitados. Um vetor x em $X(A, b)$ é uma solução do problema $\mathsf{CP}(A, b, c)$ se $cx \leq cx'$ para todo x' em $X(A, b)$. O problema pode não ter solução alguma. Isso acontece, por exemplo, se $X(A, b)$ é

[1] Veja a introdução do livro de Chvátal [Chv83] e a resenha histórica [LKS91] organizada por Lenstra, Rinnooy Kan e Schrijver.

$$
\begin{array}{rrcrcrcrcr}
\text{minimizar} & 4x_1 & + & 1x_2 & + & 5x_3 & + & 3x_4 & & \\
\text{sujeito a} & 1x_1 & - & 1x_2 & - & 1x_3 & + & 3x_4 & = & 2 \\
 & 5x_1 & + & 1x_2 & + & 3x_3 & - & 8x_4 & = & 44 \\
 & 1x_1 & - & 1x_2 & - & 4x_3 & + & 5x_4 & = & -3 \\
\text{e} & x_1 & , & x_2 & , & x_3 & , & x_4 & \geq & 0
\end{array}
$$

Figura 7.1: Um problema canônico primal, escrito em notação não-matricial.

vazio. Dizemos que o problema é **inviável**[2] se $X(A, b) = \emptyset$ e **viável** em caso contrário. inviável viável

O problema também não tem solução se $X(A, b)$ contém vetores de custo arbitrariamente negativo, ou seja, se para todo número ξ existe x em $X(A, b)$ tal que $cx < \xi$. Nesse caso, dizemos que o problema é **ilimitado**.[3] ilimitado

O problema canônico primal estará resolvido quando tivermos um algoritmo que, ao receber um sistema A, b, c, responda com uma solução x do problema $\mathsf{CP}(A, b, c)$ ou com a informação de que o problema não tem solução. É desejável que o algoritmo produza informações adicionais que permitam conferir suas respostas. Assim, no primeiro caso, o algoritmo deveria fornecer informações que permitissem verificar a minimalidade de cx e no segundo caso deveria fornecer informações que comprovassem a inexistência de solução. Esses objetivos só serão plenamente alcançados no próximo capítulo.

$$
\begin{array}{rrrrr}
1 & -1 & -1 & 3 & 2 \\
5 & 1 & 3 & -8 & 44 \\
1 & -1 & -4 & 5 & -3 \\
4 & 1 & 5 & 3 & \\
 & & & & \\
64/9 & 31/9 & 5/3 & 0 &
\end{array}
$$

Figura 7.2: A parte superior da figura exibe um sistema A, b, c, com b à direita e c abaixo da matriz A. (Compare com a figura 7.1.) O vetor x representado abaixo de c é solução do problema $\mathsf{CP}(A, b, c)$. É fácil constatar que x está em $X(A, b)$, mas não é tão fácil verificar que x minimiza cx.

Observação sobre terminologia. Para muitos praticantes de programação linear, uma "solução" é qualquer vetor x tal que $Ax = b$; uma "solução viável" é qualquer vetor em $X(A, b)$; e uma "solução ótima" é qualquer vetor x em "solução viável" "solução ótima"

[2] Não confundir com o conceito de sistema *simples* inviável definido na seção 6.2.
[3] Não confundir com o conceito de sistema *simples* ilimitado definido na seção 6.2.

$X(A,b)$ que minimize cx. O presente texto não usa essa terminologia porque ela é inconsistente e desrespeita o sentido corriqueiro da palavra *solução*.

7.2 Problemas simples

Diremos que um problema $\mathsf{CP}(A,b,c)$ é **simples** se o sistema A,b,c for simples (veja seção 6.2). Um problema simples pode ser resolvido por mera inspeção, como mostraremos a seguir.

Problema simples inviável. Digamos que os conjuntos de índices de linhas e colunas do sistema A,b,c são M e N respectivamente. Suponha que o sistema é simples inviável, ou seja, suponha que existe h em M tal que

$$A_{[h,\,]} \le o \;\text{ e }\; b_{[h]} > 0 \quad\text{ ou }\quad A_{[h,\,]} \ge o \;\text{ e }\; b_{[h]} < 0\,.$$

Então o problema $\mathsf{CP}(A,b,c)$ não tem solução: para todo vetor $x \ge o$ tem-se $A_{[h,\,]}x \le 0 < b_{[h]}$ ou $A_{[h,\,]}x \ge 0 > b_{[h]}$, donde $Ax \ne b$, e portanto $X(A,b)$ é vazio.

Problema simples solúvel. Suponha que o sistema A,b,c é simples solúvel e tem base de linhas P e base de colunas Q, isto é, suponha que

$$\begin{array}{rll} & A_{[P,Q]} \text{ é de bijeção}\,, & b_{[P]} \ge o\,,\\ A_{[M-P,N-Q]} = O\,, & A_{[M-P,Q]} = O\,, & b_{[M-P]} = o\,,\\ c_{[N-Q]} \ge o\,, & c_{[Q]} = o\,. & \end{array}$$

Seja x o vetor definido pelas equações

$$x_{[N-Q]} = o \quad\text{ e }\quad Ax = b\,.$$

Existe um e um só vetor x que satisfaz estas condições. É claro que $x_{[Q]} = \widetilde{A}_{[Q,\,P]}\, b_{[P]}$, onde $\widetilde{A}$ é a transposta de A, e portanto $x \ge o$. Esse é um **vetor básico** do sistema; dizemos que x está **associado** à base Q. É óbvio que x está em $X(A,b)$. Como $x_{[N-Q]}$ e $c_{[Q]}$ são nulos, temos

vetor básico

$$cx = 0\,.$$

Por outro lado, $cx' \ge o$ para todo x' em $X(A,b)$, pois $c \ge o$. Logo, x é solução do problema $\mathsf{CP}(A,b,c)$.

Problema simples ilimitado. Suponha que o sistema A,b,c é simples ilimitado e tem bases P e Q e coluna de ilimitação k, ou seja, suponha que

$$\begin{array}{rll} A_{[P,k]} \le o\,, & A_{[P,Q]} \text{ é de bijeção}\,, & b_{[P]} \ge o\,,\\ A_{[M-P,k]} = o\,, & A_{[M-P,Q]} = O\,, & b_{[M-P]} = o\,,\\ c_{[k]} < 0\,, & c_{[Q]} = o\,. & \end{array}$$

$$
\begin{array}{rrrrrrrr}
1 & 9 & 9 & 9 & 9 & 9 & 9 & 0 \\
0 & -9 & -9 & -9 & -9 & -9 & -9 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 9 & 9 & 9 & -9 & -9 & -9 & 1 \\
\\
0 & 0 & 0 & 0 & 0 & 0 & 0 &
\end{array}
$$

Figura 7.3: A figura define um sistema simples inviável A, b, c (com b à direita e c abaixo de A). O sistema tem duas linhas de inviabilidade: 2 e 3. O problema $\mathsf{CP}(A, b, c)$ é simples inviável.

$$
\begin{array}{rrrrrrrrrr}
9 & 9 & 9 & 9 & 9 & 0 & 0 & 0 & 1 & 12 \\
9 & 9 & 9 & 9 & 9 & 0 & 1 & 0 & 0 & 13 \\
9 & 9 & 9 & 9 & 9 & 0 & 0 & 1 & 0 & 14 \\
9 & 9 & 9 & 9 & 9 & 1 & 0 & 0 & 0 & 15 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\\
8 & 8 & 8 & 8 & 0 & 0 & 0 & 0 & 0 & \\
\\
0 & 0 & 0 & 0 & 0 & 15 & 13 & 14 & 12 &
\end{array}
$$

Figura 7.4: A parte superior da figura descreve um sistema simples solúvel A, b, c. O problema $\mathsf{CP}(A, b, c)$ é simples solúvel. Na parte inferior da figura temos um vetor básico. Esse vetor é solução do problema $\mathsf{CP}(A, b, c)$.

$$
\begin{array}{rrrrrrrrrr}
9 & 9 & 9 & 9 & 9 & -2 & 0 & 0 & 1 & 12 \\
9 & 9 & 9 & 9 & 9 & -3 & 0 & 1 & 0 & 13 \\
9 & 9 & 9 & 9 & 9 & -4 & 1 & 0 & 0 & 14 \\
9 & 9 & 9 & 9 & 9 & 0 & 0 & 0 & 0 & 0 \\
9 & 9 & 9 & 9 & 9 & 0 & 0 & 0 & 0 & 0 \\
\\
8 & 8 & -8 & -8 & 0 & -8 & 0 & 0 & 0 & \\
\\
0 & 0 & 0 & 0 & 0 & 0 & 14 & 13 & 12 & \\
0 & 0 & 0 & 0 & 0 & 1 & 4 & 3 & 2 &
\end{array}
$$

Figura 7.5: A parte superior da figura define um sistema simples ilimitado A, b, c. O problema $\mathsf{CP}(A, b, c)$ é simples ilimitado. A parte inferior da figura define vetores x e x'. Para qualquer λ positivo, o vetor $x + \lambda x'$ está em $X(A, b)$ e tem custo -8λ, o que comprova o caráter ilimitado do problema.

Então o problema $\mathsf{CP}(A, b, c)$ é ilimitado, pelas razões que passamos a expor. Seja x o vetor definido pelas equações

$$x_{[N-Q]} = o \quad \text{e} \quad Ax = b\,.$$

Existe um e um só vetor que satisfaz estas equações; é claro que $x \geq o$. Seja x' o vetor definido pelas equações

$$x'_{[N-Q-k]} = o\,, \quad x'_{[k]} = 1 \quad \text{e} \quad Ax' = o\,.$$

Existe um e um só vetor que satisfaz estas equações. É claro que $x'_{[Q]} = -\widetilde{A}_{[Q,\,P]}\, A_{[P,\,k]}$ e portanto $x' \geq o$. Para qualquer λ positivo, é óbvio que o vetor $x + \lambda x'$ está em $X(A, b)$. Ademais,

$$c\,(x + \lambda x') \;=\; \lambda\, c_{[k]}\,,$$

donde $c\,(x + \lambda x')$ é tanto menor quanto maior for λ. Portanto, o problema $\mathsf{CP}(A, b, c)$ é ilimitado.

7.3 O Simplex resolve o problema

Para resolver um problema canônico primal arbitrário, basta reduzí-lo a um problema simples. Uma redução genérica tem o efeito descrito a seguir.

Observação 7.1 *Para qualquer sistema A, b, c, qualquer vetor g e quaisquer matrizes F e G tais que $FG = I$, os problemas $\mathsf{CP}(A, b, c)$ e $\mathsf{CP}(GA, Gb, c + gA)$ têm o mesmo conjunto de soluções.*

DEMONSTRAÇÃO: Se $Ax = b$ então é claro que $(GA)x = Gb$. Reciprocamente, se $(GA)x = Gb$ então

$$Ax = (FG)Ax = F(GA)x = F(Gb) = (FG)b = b\,.$$

Esse raciocínio mostra que $Ax = b$ se e só se $GAx = Gb$ e portanto que $X(A, b) = X(GA, Gb)$. Resta mostrar que minimizar cx é o mesmo que minimizar $(c + gA)x$. Isto é assim porque a diferença entre cx e $(c + gA)x$ não depende de x:

$$cx - (c + gA)x = cx - cx - (gA)x = -g(Ax) = -gb\,.$$

Logo, se cx é mínimo em $X(A, b)$ então $(c+gA)x$ também é mínimo em $X(A, b)$. Reciprocamente, se $(c + gA)x$ é mínimo então cx também é mínimo. □

Portanto, qualquer problema canônico primal $\mathsf{CP}(A, b, c)$ pode ser resolvido da seguinte maneira: Submeta o sistema A, b, c ao algoritmo Simplex (veja capítulo 6). O algoritmo devolverá matrizes F e G e um vetor g tais que $FG = I$ e o sistema A', b', c' é simples, sendo

$$A' = GA\,, \quad b' = Gb \quad \text{e} \quad c' = c + gA\,.$$

$$\begin{array}{rrrrrrrr} 1 & 2 & 1 & 0 & 1 & 0 & 0 & 12 \\ 2 & 5 & 0 & -1 & 0 & 1 & 0 & 10 \\ -1 & -3 & 1 & 1 & 0 & 0 & 1 & 2 \\ 0 & 1 & -2 & -1 & 0 & 1 & 0 & \end{array}$$

Figura 7.6: Ilustração da observação 7.1. Considere o sistema A, b, c definido pela figura. Seja a o vetor $A[2,]$. Para todo x em $X(A, b)$ temos $(c + a)x = cx + ax = cx + 10$. Portanto, minimizar cx é o mesmo que minimizar $(c + a)x$. Os problemas $\mathsf{CP}(A, b, c)$ e $\mathsf{CP}(A, b, c + a)$ são equivalentes.

Se A', b', c' é simples inviável, o problema $\mathsf{CP}(A, b, c)$ é inviável. Se A', b', c' é simples solúvel, então o vetor básico x do sistema A', b', c' é solução do problema $\mathsf{CP}(A, b, c)$. Se A', b', c' é simples ilimitado, o problema $\mathsf{CP}(A, b, c)$ não tem solução por ser ilimitado.

O algoritmo que acabamos de descrever mostra que, para qualquer sistema A, b, c, o problema $\mathsf{CP}(A, b, c)$ tem solução ou é inviável ou é ilimitado.

7.4 Conclusão

O problema canônico primal consiste em encontrar um vetor x que minimize cx sujeito às restrições $x \geq o$ e $Ax = b$. Todo problema canônico primal tem solução ou é inviável ou é ilimitado. O algoritmo Simplex reduz qualquer problema canônico primal a um problema equivalente que pode ser resolvido por mera inspeção.

terminologia

Convém acrescentar duas observações sobre terminologia. Quando o número cx é interpretado como custo do vetor x, diz-se que $(c + gA)\,x$ é o **custo reduzido** de x. Diz-se que uma solução x do problema $\mathsf{CP}(A, b, c)$ é **básica** se x é o vetor básico do problema simples solúvel $\mathsf{CP}(GA, Gb, c + gA)$.

Durante a discussão do Simplex, adiamos a análise da seguinte questão de consistência: É possível reduzir um dado sistema A, b, c a dois sistemas simples de tipos diferentes? A resposta a esta pergunta fica clara agora. Digamos que F, G, g, F', G', g' são tais que $FG = I$, $F'G' = I$ e os sistemas

$$GA, Gb, c + gA \quad \text{e} \quad G'A, G'b, c + g'A$$

são ambos simples. Então os dois sistemas simples são de um mesmo tipo. De fato, *(1)* se o primeiro sistema é simples inviável, então $X(A, b)$ é vazio e portanto o segundo sistema não pode ser simples solúvel nem simples ilimitado; *(2)* se o primeiro sistema é simples solúvel, então existe x em $X(A, b)$ tal que cx é mínimo e, portanto, o segundo sistema não pode ser simples inviável nem ilimitado; *(3)* se o primeiro sistema é simples ilimitado, então $X(A, b)$ não é vazio e cx não tem mínimo, donde o segundo sistema não pode ser simples inviável nem solúvel.

7.5 Exemplo

Uma empresa fabrica quatro modelos de um produto. Digamos que x_i é o número de unidades do modelo i produzidas. Cada produto passa por dois estágios de fabricação. O primeiro estágio dispõe de não mais que 600 homens-hora e o segundo de não mais que 400 homens-hora. Digamos que o número de homens-hora necessários, em cada estágio, para a fabricação de cada modelo do produto impõe as seguintes restrições:

$$\begin{aligned} 4x_1 + 9x_2 + 7x_3 + 10x_4 &\leq 600 \\ 1x_1 + 1x_2 + 3x_3 + 40x_4 &\leq 400\,. \end{aligned}$$

Digamos que o lucro total é $12x_1 + 20x_2 + 18x_3 + 40x_4$. Queremos planejar a produção de modo a maximizar o lucro total.

Maximizar o lucro é o mesmo que minimizar $-12x_1 - 20x_2 - 18x_3 - 40x_4$. Assim, nosso problema equivale ao seguinte problema canônico primal: encontrar números não-negativos $x_1, \ldots, x_6$ que minimizem $-12x_1 - 20x_2 - 18x_3 - 40x_4$ respeitadas as restrições

$$\begin{aligned} 4x_1 + 9x_2 + 7x_3 + 10x_4 + 1x_5 + 0x_6 &= 600 \\ 1x_1 + 1x_2 + 3x_3 + 40x_4 + 0x_5 + 1x_6 &= 400\,. \end{aligned}$$

As "variáveis de folga" x_5 e x_6 correspondem a modelos fictícios do produto; elas consomem o que sobra da disponibilidade da mão de obra. Em suma, estamos diante de um problema $\mathsf{CP}(A, b, c)$, onde A, b, c é o sistema descrito na figura 7.7.

$$\begin{array}{rrrrrrr} 4 & 9 & 7 & 10 & 1 & 0 & 600 \\ 1 & 1 & 3 & 40 & 0 & 1 & 400 \\ -12 & -20 & -18 & -40 & 0 & 0 & \end{array}$$

Figura 7.7: Sistema A, b, c.

Seja G a matriz e g o vetor descritos na figura 7.8 (verifique que G é inversível). O sistema $GA, Gb, c+gA$ (figura 7.9) é simples solúvel; sua base é composta pelas colunas 1 e 4. O plano de produção ótimo é

$$x_1 = \tfrac{1}{3}\,400\,, \quad x_2 = x_3 = 0\,, \quad x_4 = \tfrac{1}{3}\,20\,, \quad x_5 = x_6 = 0\,.$$

Com este plano, teremos $cx = -\frac{1}{3}\,5600$. Os números são fracionários porque nosso modelo não exige que x_1, x_2, x_3, x_4 sejam inteiros; tal exigência tiraria o problema do mundo da programação linear e o tornaria bem mais complexo. Talvez nossa empresa queira adotar $x_1 = 133$, $x_2 = x_3 = 0$ e $x_4 = 7$ e torcer para que o arredondamento não afete muito o lucro.

programação inteira

$$\begin{array}{rr} 4/15 & -1/15 \\ -1/150 & 4/150 \\ 44/15 & 4/15 \end{array}$$

Figura 7.8: Matriz G e vetor g.

$$\begin{array}{rrrrrrr} 1 & 35/15 & 25/15 & 0 & 4/15 & -1/15 & 400/3 \\ 0 & -5/150 & 5/150 & 1 & -1/150 & 4/150 & 20/3 \\ 0 & 20/3 & 10/3 & 0 & 44/15 & 4/15 & \end{array}$$

Figura 7.9: Sistema $GA, Gb, c + gA$.

Exercícios

7.1 Suponha que para todo x em $X(A, b)$ existe x' em $X(A, b)$ tal que $cx' < cx$. O problema $\mathsf{CP}(A, b, c)$ é ilimitado?

7.2 Suponha que $c \geq o$. Mostre que o vetor nulo é solução do problema canônico $\mathsf{CP}(A, o, c)$.

7.3 Seja x um vetor em $X(A, b)$ e g um vetor tal que $c+gA \geq o$ e $(c+gA)x = 0$. Prove que x é solução do problema $\mathsf{CP}(A, b, c)$.

7.4 Os problemas canônicos $\mathsf{CP}(A, b, c)$ e $\mathsf{CP}(A, b, -c)$ podem ser ambos ilimitados?

7.5 A **folga** de x é o conjunto, digamos $S(x)$, de todos os índices j para os quais $x_{[j]}$ não é nulo. Suponha que A, b, c é um sistema simples solúvel e que x é o correspondente vetor básico. Mostre que a folga de x é minimal, ou seja, que não existe x' em $X(A, b)$ tal que $S(x') \subset S(x)$, em que $\subset$ indica subconjunto próprio. (Veja apêndice C, página 178.) $\subset$

7.6 Seja A uma matriz sobre $M \times N$, b um vetor sobre M e i um elemento de M. Mostre que o problema de encontrar o menor valor de $b_{[i]}$ para o qual $X(A, b)$ não é vazio equivale ao problema $\mathsf{CP}(A_{[M-i,\,]}, b_{[M-i]}, A_{[i,\,]})$.

Capítulo 8

Problema canônico dual

Dual: **1.** Composto de duas partes. [. . .]
3. Divisão da categoria de número que existia, ao lado do singular e do plural, no indo-europeu e em certas línguas dele derivadas, indicando um par de seres.
Novo Dicionário Aurélio

O problema canônico primal, de que tratou o capítulo anterior, está intimamente relacionado com um segundo problema básico de programação linear. A relação entre os dois problema é o objeto de estudo da teoria da dualidade. O algoritmo Simplex (capítulo 6) pode ser usado para resolver os dois problemas simultaneamente.

8.1 Definição do problema

O **problema canônico dual** de programação linear consiste no seguinte:

> **Problema** $\mathsf{CD}(A, c, b)$: *Dada uma matriz A, um vetor b e um vetor c, encontrar um vetor y tal que $yA \leq c$ e yb é máximo.*

As inequações $yA \leq c$ são as **restrições** do problema.[1] Poderíamos enunciar o problema dizendo: *maximize yb sob as restrições $yA \leq c$.* O conjunto de todos os vetores y que satisfazem as restrições será denotado por restrições

$$Y(A, c)\,.$$

Diz-se que $Y(A, c)$ é o **poliedro canônico dual**. Se o número yb for interpretado como *valor* do vetor y, o problema consiste em encontrar um vetor de valor máximo em $Y(A, c)$.

[1] O leitor atento já terá reconhecido nessas restrições um reflexo da condição $c + gA \geq o$, presente na definição de sistema simples solúvel (seção 6.2).

$$
\begin{array}{lrcrcrcl}
\text{maximize} & 2y_1 & + & 44y_2 & - & 3y_3 & & \\
\text{sujeito a} & y_1 & + & 5y_2 & + & y_3 & \leq & 4 \\
& -y_1 & + & y_2 & - & y_3 & \leq & 1 \\
& -y_1 & + & 3y_2 & - & 4y_3 & \leq & 5 \\
& 3y_1 & - & 8y_2 & + & 5y_3 & \leq & 3
\end{array}
$$

Figura 8.1: Um problema canônico dual (compare com figura 7.1), escrito em notação não-matricial.

Problemas solúveis, inviáveis e ilimitados. Um vetor y em $Y(A, c)$ é solução do problema canônico dual $\mathsf{CD}(A, c, b)$ se $yb \geq y'b$ para todo y' em $Y(A, c)$. O problema pode não ter solução alguma. Isto acontece, por exemplo, se $Y(A, c)$ é vazio; acontece também se $Y(A, c)$ contém vetores de valor arbitrariamente grande, isto é, se para todo número ξ existe y em $Y(A, c)$ tal que $yb > \xi$. No primeiro caso, dizemos que o problema é **inviável**. No segundo caso, dizemos que o problema é **ilimitado**. Se $Y(A, c)$ não é vazio, dizemos que o problema é **viável**.

inviável
ilimitado
viável

8.2 Lema da dualidade

Há uma relação muito íntima entre os problemas canônico dual e canônico primal. Essa relação está resumida no seguinte lema.

Lema 8.1 (da dualidade canônica) *Para todo x em $X(A, b)$ e todo y em $Y(A, c)$ tem-se $cx \geq yb$.*

DEMONSTRAÇÃO (muito simples, mas fundamental): Basta examinar o produto $y \cdot A \cdot x$. Por um lado,

$$yAx \;=\; y\,(Ax) \;=\; yb\,,$$

onde a segunda igualdade vale porque $Ax = b$. Por outro lado,

$$yAx \;=\; (yA)\,x \;\leq\; cx\,,$$

onde a desigualdade vale porque $yA \leq c$ e $x \geq o$. □

O lema é às vezes chamado, um tanto pomposamente, de teorema fraco da dualidade. Vale o seguinte corolário, óbvio mas importante: *Se o problema primal $\mathsf{CP}(A, b, c)$ é ilimitado então o problema dual $\mathsf{CD}(A, c, b)$ é inviável.* É claro que também vale o corolário dual: *Se o problema primal é ilimitado então o problema dual é inviável.* Portanto, para mostrar que o problema primal não é ilimitado basta exibir um elemento de $Y(A, c)$; e para mostrar que o problema dual não é ilimitado basta exibir um vetor em $X(A, b)$.

teorema fraco da dualidade

$$\begin{array}{rrrrrr}
2 & 1 & -1 & -1 & 3 & 2 \\
0 & 5 & 1 & 3 & -8 & 44 \\
-1 & 1 & -1 & -4 & 5 & -3 \\
 & 4 & 1 & 5 & 3 &
\end{array}$$

Figura 8.2: Ilustração do lema 8.1. Seja A, b, c o sistema definido pela parte direita da figura (o mesmo da figura 7.2). Seja y o vetor representado à esquerda de A, na vertical. Verifique que $yA \leq c$ e $yb = 7$. Conclua que $cx \geq 7$ para todo $x \geq o$ tal que $Ax = b$.

O lema da dualidade tem mais um corolário importante: *Para qualquer x em $X(A,b)$ e qualquer y em $Y(A,c)$, se $cx = yb$ então x é solução de* $\mathsf{CP}(A,b,c)$ *e y é solução de* $\mathsf{CD}(A,c,b)$. Portanto, para tornar evidente que um certo vetor x em $X(A,b)$ de fato minimiza cx é suficiente exibir um vetor y em $Y(A,c)$ tal que $cx = yb$. Ao mesmo tempo, para tornar óbvio que um certo vetor y em $Y(A,c)$ maximiza yb basta exibir um vetor x em $X(A,b)$ tal que $cx = yb$.

$$\begin{array}{rrrrrr}
11/18 & 1 & -1 & -1 & 3 & 2 \\
5/6 & 5 & 1 & 3 & -8 & 44 \\
-7/9 & 1 & -1 & -4 & 5 & -3 \\
 & 4 & 1 & 5 & 3 & \\
 & & & & & \\
 & 64/9 & 31/9 & 5/3 & 0 &
\end{array}$$

Figura 8.3: Sistema A, b, c (o mesmo da figura 8.2) e vetores y (à esquerda de A) e x (abaixo de c). Verifique que y está em $Y(A,c)$, que x está em $X(A,b)$ e que $yb = 362/9 = cx$. Conclua que x é solução do correspondente problema canônico primal e y é solução do problema canônico dual.

Folgas complementares. A **folga** de um elemento x de $X(A,b)$ é o conjunto dos índices k para os quais $x_{[k]} > 0$; a **folga** de um elemento y de $Y(A,c)$ é o conjunto dos índices q para os quais $(yA)_{[q]} < c_{[q]}$. Dizemos que o par x, y **tem folgas complementares** se a folga de x é disjunta da folga de y. Se N denota o conjunto de índices de colunas de A, a definição também pode ser formulada assim: existe uma parte Q de N tal que $(yA)_{[Q]} = c_{[Q]}$ e $x_{[N-Q]} = o$.

Fato 8.2 *Para todo x em $X(A,b)$ e todo y em $Y(A,c)$, o par x, y tem folgas complementares se e só se $cx = yb$.*

DEMONSTRAÇÃO: Suponha que x, y tem folgas complementares. Então $cx - yb = cx - yAx = (c - yA)x = \sum_j (c - yA)_{[j]}\, x_{[j]} = 0$. Logo, $cx = yb$.

Suponha agora que $cx = yb$. Então $cx - yAx = 0$, donde $\sum_j (c - yA)_{[j]}\, x_{[j]} = 0$. Como $x \geq o$ e $yA \leq c$, cada um dos termos desta soma é menor ou igual a 0. Como a soma é nula, cada um de seus termos deve ser nulo. □

8.3 Vetores de inviabilidade

A inviabilidade do problema canônico dual está intimamente ligada à ilimitação do problema canônico primal. De modo análogo, há uma relação íntima entre a ilimitação do problema canônico dual e a inviabilidade do problema canônico primal. Os dois lemas abaixo resumem essas relações. Os lemas mostram que para tornar óbvia a inviabilidade e/ou ilimitação dos problemas basta exibir vetores apropriados em $X(A, o)$ e/ou em $Y(A, o)$.

Lema 8.3 (da inviabilidade primal) *Se $y'b > 0$ para algum vetor y' em $Y(A, o)$ então o problema primal $\mathsf{CP}(A, b, c)$ é inviável e o problema dual $\mathsf{CD}(A, c, b)$ é ilimitado ou inviável.*

DEMONSTRAÇÃO: Seja y' um vetor como o descrito no enunciado. Para todo x em $X(A, b)$ teríamos a contradição $0 < y'b = y'(Ax) = (y'A)x \leq 0$, donde se conclui que $X(A, b)$ é vazio.

Suponha que $\mathsf{CD}(A, c, b)$ não é inviável e seja y um vetor em $Y(A, c)$. Então, para todo λ positivo, $(y + \lambda y')A = yA + \lambda y'A \leq c + o = c$, e portanto o vetor $y + \lambda y'$ está em $Y(A, c)$. O valor desse vetor, igual a $yb + \lambda y'b$, é tanto maior quanto maior for λ. Portanto o problema $\mathsf{CD}(A, c, b)$ é ilimitado. □

Diante desse lema, é razoável que todo vetor y' em $Y(A, o)$ que satisfaça a condição $y'b > 0$ seja chamado **vetor de inviabilidade do problema primal**. $\mathsf{CP}(A, b, c)$.

vetor de inviabilidade primal

$$\begin{array}{cc|ccccc} 1 & 1/2 & 2 & 2 & 4 & 0 & 24 \\ 0 & 0 & 1 & 1 & 2 & 0 & 12 \\ -1 & -1 & 2 & 0 & 5 & -1 & -10 \\ -1 & -1 & -1 & 1 & -2 & 1 & 2 \\ & & 1 & 1 & 1 & 1 & \end{array}$$

Figura 8.4: Ilustração do lema 8.3. A figura mostra um sistema A, b, c, um vetor y' (à esquerda de A) e um vetor y (à esquerda de y'). Verifique que y está em $Y(A, c)$, que y' está em $Y(A, o)$, e que $y'b$ é positivo. Conclua que $\mathsf{CP}(A, b, c)$ é inviável e $\mathsf{CD}(A, c, b)$ é ilimitado.

Lema 8.4 (da inviabilidade dual) *Se $cx' < 0$ para algum vetor x' em $X(A, o)$ então o problema dual $\mathsf{CD}(A, c, b)$ é inviável e o problema primal $\mathsf{CP}(A, b, c)$ é ilimitado ou inviável.*

DEMONSTRAÇÃO: Seja x' um vetor como o descrito no enunciado. Para todo y em $Y(A, c)$ teríamos a contradição $0 > cx' \geq yAx' = yo = 0$. Logo, $Y(A, c)$ é vazio.

Suponha que $\mathsf{CP}(A, b, c)$ não é inviável e seja x um vetor qualquer em $X(A, b)$. Então, para todo λ positivo, $x + \lambda x'$ está em $X(A, b)$, pois $A(x + \lambda x') = Ax + \lambda Ax' = b + o = b$ e $x + \lambda x' \geq o$. Ademais, o custo de $x + \lambda x'$, igual a $cx + \lambda cx'$, é tanto menor quanto maior for λ. Portanto, o problema $\mathsf{CP}(A, b, c)$ é ilimitado. □

Esse lema justifica o uso do termo **vetor de inviabilidade do problema dual** $\mathsf{CD}(A, c, b)$ para designar qualquer vetor x' em $X(A, o)$ tal que $cx' < 0$.

vetor de inviabilidade dual

$$\begin{array}{rrrrr} 2 & 2 & 4 & -8 & 24 \\ 1 & 1 & 2 & -4 & 12 \\ 2 & 0 & 5 & -4 & 10 \\ -1 & 1 & -2 & 0 & 2 \end{array}$$

$$\begin{array}{rrrr} 1 & 1 & 1 & -8 \end{array}$$

$$\begin{array}{rrrr} 5 & 7 & 0 & 0 \\ 2 & 2 & 0 & 1 \end{array}$$

Figura 8.5: Ilustração do lema 8.5. Sistema sistema A, b, c, vetor x (abaixo de c) e vetor x' (abaixo de x). Verifique que x está em $X(A, b)$, que x' está em $X(A, o)$ e que cx' é negativo. Conclua que $\mathsf{CP}(A, b, c)$ é ilimitado e $\mathsf{CD}(A, c, b)$ é inviável.

8.4 Algoritmo baseado no Simplex

O algoritmo abaixo usa o Simplex para resolver simultaneamente os dois problemas canônicos. O algoritmo produz uma solução do primal e uma solução do dual, ou provas da inviabilidade do primal e da ilimitação do dual, ou provas da ilimitação do primal e da inviabilidade do dual, ou provas da inviabilidade de ambos os problemas. Se os dois problemas têm solução, cada uma constitui prova da otimalidade (minimalidade ou maximalidade) da outra. As provas de inviabilidade e de ilimitação consistem em vetores de inviabilidade apropriados.

Algoritmo para o par de problemas canônicos *Recebe um sistema A, b, c e devolve um dos seguintes pares de vetores:*

(1) *x' em $X(A, o)$ e y' em $Y(A, o)$ tais que $cx' < 0$ e $y'b > 0$, ou*
(2) *y em $Y(A, c)$ e y' em $Y(A, o)$ tal que $y'b > 0$, ou*
(3) *x em $X(A, b)$ e x' em $X(A, o)$ tal que $cx' < 0$, ou*
(4) *x em $X(A, b)$ e y em $Y(A, c)$ tais que $cx = yb$.*

Nos itens *(1)* e *(2)*, y' é um vetor de inviabilidade do problema primal $\mathsf{CP}(A, b, c)$. Nos itens *(1)* e *(3)*, x' é um vetor de inviabilidade do problema dual $\mathsf{CD}(A, c, b)$. No item *(4)*, x é solução do problema primal e y é solução do problema dual.

O algoritmo consiste no seguinte. Seja N o conjunto de índice de colunas de A. Submeta o sistema A, b, c ao algoritmo Simplex, que devolverá matrizes F e G e um vetor g tais que $FG = I$ e o sistema $GA, Gb, c + gA$ é simples.

CASO 1: *o sistema GA, Gb, $c + gA$ é simples inviável.*
Seja h uma linha de inviabilidade. Defina o vetor y' da seguinte maneira: se $G_{[h,\,]}b > 0$ então $y' = G_{[h,\,]}$ senão $y' = -G_{[h,\,]}$. Submeta o sistema A, o, c ao algoritmo Simplex, que devolverá matrizes F_0 e G_0 e um vetor g_0 tais que $F_0G_0 = I$ e o sistema G_0A, G_0o, $c + g_0A$ é simples solúvel ou simples ilimitado.

CASO 1.1: *o sistema G_0A, G_0o, $c + g_0A$ é simples ilimitado.*
Seja Q_0 uma base de colunas e k uma coluna de ilimitação do sistema. Seja x' o vetor definido pelas equações $x'_{[k]} = 1$, $x'_{[N-Q_0-k]} = o$ e $(G_0A)x' = o$. Devolva x' e y' e pare.

CASO 1.2: *o sistema G_0A, G_0o, $c + g_0A$ é simples solúvel.*
Devolva $-g_0$ e y' e pare.

CASO 2: *o sistema GA, Gb, $c + gA$ é simples ilimitado.*
Seja Q uma base de colunas e k uma coluna de ilimitação do sistema. Seja x o vetor básico associado a Q (isto é, $x_{[N-Q]} = o$ e $(GA)x = Gb$). Seja x' o vetor definido pelas equações $x'_{[k]} = 1$, $x'_{[N-Q-k]} = o$ e $(GA)x' = o$. Devolva x e x' e pare.

CASO 3: *o sistema GA, Gb, $c + gA$ é simples solúvel.*
Seja Q uma base de colunas do sistema. Seja x o vetor básico associado a Q (isto é, $x_{[N-Q]} = o$ e $(GA)x = Gb$). Devolva x e $-g$ e pare. □

Eis a análise do algoritmo. No caso 1, é claro que $y'A \leq o$ e $y'b$ é positivo. Em particular, y' está em $Y(A, o)$. Para decidir se $Y(A, c)$ é ou não vazio, o algoritmo resolve o problema $\mathsf{CP}(A, o, c)$. Como $X(A, o)$ contém o vetor nulo, o problema $\mathsf{CP}(A, o, c)$ não é inviável; isto explica por que o caso 1 tem apenas dois e não três subcasos.

No caso 1.1, o vetor x' está em $X(G_0A, o)$, que é idêntico a $X(A, o)$ pois G_0 é inversível. Ademais, cx' é negativo pois

$$\begin{aligned} cx' &= (c + g_0A)x' - g_0Ax' \\ &= (c + g_0A)_{[k]}\, x'_{[k]} - g_0o \\ &= (c + g_0A)_{[k]} \end{aligned}$$

e $(c + g_0A)_{[k]}$ é negativo. Portanto, o par de vetores x', y' satisfaz o item *(1)* do enunciado do algoritmo.

No caso 1.2 temos $c + g_0 A \geq o$ e portanto $-g_0$ está em $Y(A, c)$. Assim, o par $-g_0, y'$ satisfaz o item *(2)* do enunciado do algoritmo.

No caso 2, o algoritmo devolve um vetor x em $X(GA, Gb)$ e um vetor x' em $X(GA, o)$. Os vetores x e x' também estão em $X(A, b)$ e $X(A, o)$ respectivamente, pois G é inversível. Ademais,

$$\begin{aligned} cx' &= (c + gA)x' - gAx' \\ &= (c + gA)_{[k]}\, x'_{[k]} - go \\ &= (c + gA)_{[k]}\,, \end{aligned}$$

donde cx' é negativo. Portanto, ao devolver x e x' o algoritmo está se comportando como previsto no item *(3)*.

No caso 3 tem-se $x \geq o$, $Ax = b$ e $c + gA \geq o$, donde x está em $X(A, b)$ e $-g$ está em $Y(A, c)$. Ademais, o par $x, -g$ tem folgas complementares, uma vez que $(c + gA)_{[Q]}$ e $x_{[N-Q]}$ são nulos. Logo, $cx = -gb$. Assim, ao devolver x e $-g$ o algoritmo está se comportanto como previsto no item *(4)*.

8.5 Teorema da dualidade

O algoritmo da seção anterior demonstra o seguinte teorema da dualidade canônica, também conhecido como teorema forte da dualidade.

teorema forte da dualidade

Teorema 8.5 (da dualidade canônica) *Para qualquer matriz A, qualquer vetor b e qualquer vetor c, vale uma e apenas uma das seguintes afirmações:*

(1) existem x' em $X(A, o)$ e y' em $Y(A, o)$ tais que $cx' < 0$ e $y'b > 0$;
(2) existe y em $Y(A, c)$ e y' em $Y(A, o)$ tal que $y'b > 0$;
(3) existe x em $X(A, b)$ e x' em $X(A, o)$ tal que $cx' < 0$;
(4) existem x em $X(A, b)$ e y em $Y(A, c)$ tais que $cx = yb$.

(Fica subentendido que b é indexado pelo mesmo conjunto que as linhas de A e que c é indexado pelo mesmo conjunto que as colunas de A.) O teorema da dualidade pode ser resumido da seguinte maneira: a menos que os problemas $\mathsf{CP}(A, b, c)$ e $\mathsf{CD}(A, c, b)$ sejam ambos inviáveis, tem-se

$$\min_x cx \;=\; \max_y yb\,,$$

onde $\min$ é tomado sobre x em $X(A, b)$ e $\max$ é tomado sobre y em $Y(A, c)$. Essa identidade traduz não somente o caso *(4)* do enunciado do teorema, mas também os casos *(2)* e *(3)*, se estivermos dispostos a dizer que $\min cx = +\infty$ quando o problema primal é inviável, que $\min cx = -\infty$ quando o problema primal é ilimitado, que $\max yb = -\infty$ quando o problema dual é inviável e que $\max yb = +\infty$ quando o problema dual é ilimitado. A igualdade só não vale no caso *(1)*, quando os dois problemas são inviáveis. (Esse é o caso, por exemplo,

	X vazio	*X não-vazio*
Y vazio	primal inviável dual inviável	primal ilimitado dual inviável
Y não-vazio	primal inviável dual ilimitado	primal solúvel dual solúvel

Figura 8.6: Os quatro casos do teorema da dualidade 8.5 correspondem exatamente aos quatro possíveis valores do par $X(A, b), Y(A, c)$.

se todos os componentes de A são iguais a 0, todos os componentes de b são iguais a 1, e todos os componentes de c são iguais a -1.)

Todas as operações aritméticas do algoritmo Simplex transformam números racionais em outros números racionais. Portanto, vale o seguinte adendo ao teorema da dualidade: se os componentes de A, b, c são números racionais então os casos *(1), (2), (3)* ou *(4)* são satisfeitos por vetores com componentes racionais.

8.6 Conclusão

O problema canônico dual consiste em encontrar um vetor y que maximize yb sujeito às restrições $yA \leq c$. Todo problema canônico dual tem solução ou é inviável ou é ilimitado.

Há uma íntima relação entre o problema canônico primal $\mathsf{CP}(A, b, c)$ e o problema canônico dual $\mathsf{CD}(A, c, b)$: os dois problemas constituem as duas faces de uma mesma moeda. A menos que os dois problemas sejam inviáveis, tem-se $\min_x cx = \max_y yb$.

O algoritmo Simplex pode ser usado para resolver simultaneamente os dois problemas. Para esclarecer de forma cabal a natureza dos dois problemas (ambos inviáveis, um inviável e outro ilimitado, ou ambos solúveis), basta exibir dois vetores convenientes extraídos dos conjuntos $X(A, b)$, $Y(A, c)$, $X(A, o)$ e $Y(A, o)$.

8.7 Apêndice: Uma interpretação do Simplex

É interessante interpretar o funcionamento do Simplex à luz dos conceitos deste capítulo. Vamos nos restringir à segunda fase do algoritmo, em que as coisas ficam mais claras. Considere, pois, uma seqüência de iterações em que ocorre a alternativa II. Cada iteração começa com um sistema $GA, Gb, c - yA$ e uma base Q e portanto também, implicitamente, com o vetor básico x do sistema. Podemos dizer então que cada iteração começa com vetores x e y tais que

x está em $X(A,b)$ e
o par x, y tem folgas complementares (donde $cx = yb$).

Se y está em $Y(A,c)$, a execução do algoritmo termina. Senão, o algoritmo calcula

um novo par $\bar{x}, \bar{y}$ tal que $c\bar{x} \leq cx$ ou
um vetor x' em $X(A,o)$ tal que $cx' < 0$.

No primeiro caso, o algoritmo começa nova iteração. No segundo, a execução do algoritmo termina.

8.8 Apêndice: Problema do vetor viável

O **problema do vetor primal viável** consiste em encontrar um elemento de $X(A,b)$. Mais especificamente: dada uma matriz A e um vetor b, encontrar um vetor x tal que $x \geq o$ e $Ax = b$. Não é difícil verificar que esse problema equivale ao problema canônico $\mathsf{CP}(A,b,o)$.

O **problema do vetor dual viável** consiste em encontrar um elemento de $Y(A,c)$. Mais especificamente: dada uma matriz A e um vetor c, encontrar um vetor y tal que $yA \leq c$. Esse problema equivale ao problema $\mathsf{CD}(A,c,o)$.

Os seguintes corolários do teorema da dualidade (conhecidos como lemas de Farkas [Chv83, p.248]) descrevem as condições em que os problemas do vetor viável têm solução.

Corolário 8.6 (lema de Farkas) *Para qualquer matriz A e qualquer vetor b, vale uma e apenas uma das alternativas: (1) existe x em $X(A,b)$; (2) existe y' em $Y(A,o)$ tal que $y'b > 0$.*

Corolário 8.7 (lema de Farkas) *Para qualquer matriz A e qualquer vetor c, vale uma e apenas uma das alternativas: (1) existe y em $Y(A,c)$; (2) existe x' em $X(A,o)$ tal que $cx' < 0$.*

Do ponto de vista computacional, o corolário 8.6 precede o teorema da dualidade: antes de calcular os objetos de que trata o teorema, o algoritmo Simplex calcula (implicitamente) os objetos de que trata o corolário. É instrutivo demonstrar os corolários diretamente a partir do Simplex, ainda que as demonstrações sejam uma reciclagem de raciocínios já feitos acima.

DEMONSTRAÇÃO DE 8.6: Submeta A, b, o ao Simplex. O algoritmo devolverá objetos F, G e g tais que $FG = I$ e GA, Gb, gA é simples. É fácil verificar que g será necessariamente nulo e portanto GA, Gb, gA será simples solúvel ou simples inviável. No primeiro caso, qualquer vetor básico do sistema GA, Gb, o satisfaz *(1)*. No segundo caso, se h é uma linha de inviabilidade então $G_{[h,\,]}$ ou $-G_{[h,\,]}$ satisfaz *(2)*.

Resta mostrar que as duas alternativas não podem ser simultaneamente verdadeiras. Se *(1)* e *(2)* fossem ambas verdadeiras teríamos a contradição $0 < y'b = y'Ax \leq ox = 0$. □

DEMONSTRAÇÃO DE 8.7: Submeta A, o, c ao algoritmo Simplex, que devolverá objetos F, G e g tais que $FG = I$ e $GA, Go, c + gA$ é simples solúvel ou simples ilimitado (é claro que o sistema $GA, Go, c + gA$ não é simples inviável). No primeiro caso, $-g$ satisfaz *(1)*. No segundo, é fácil extrair de GA e $c + gA$ um vetor x' que satisfaz *(1)*.

As duas alternativas não podem ser simultaneamente verdadeiras: se assim fosse, teríamos a contradição $0 > cx' \geq yAx' = yo = 0$. □

Exercícios

8.1 Prove que o problema $\mathsf{CD}(A, c, b)$ é viável se e só se o problema $\mathsf{CP}(A, o, c)$ tem solução. Prove a afirmação dual: o problema $\mathsf{CP}(A, b, c)$ é viável se e só se o problema $\mathsf{CD}(A, o, b)$ tem solução.

8.2 Se existe um vetor y' em $Y(A, o)$ tal que $y'b > 0$ então existe um vetor y'' em $Y(A, o)$ tal que $y''b \geq 1$. Se existe um vetor x' em $X(A, o)$ tal que $cx' < 0$ então existe x'' em $X(A, o)$ tal que $cx'' \leq -1$.

8.3 Mostre que para todo x em $X(A, o)$, todo y em $Y(A, o)$ e todo índice j tem-se $(yA)_{[j]} = 0$ ou $x_{[j]} = 0$.

8.4 [Chv83] Dada uma matriz A, vetores b e c, e um número δ, dizemos que o par A, c **implica** o par b, δ se o sistema de inequações $yA \leq c$ tem pelo menos uma solução e se todo y que satisfaz o sistema $yA \leq c$ também satisfaz a inequação $yb \leq \delta$. Mostre que o teorema da dualidade é equivalente à seguinte proposição: se A, c implica b, δ então existe $x \geq o$ tal que $Ax = b$ e $cx \leq \delta$.

8.5 Encontre números não-negativos x_1, x_2, x_3, x_4 que satisfaçam as equações

$$\begin{array}{rcrcrcrcr}
2x_1 & + & 4x_2 & + & 2x_3 & & & = & 4 \\
x_1 & + & 2x_2 & + & x_3 & & & = & 2 \\
2x_1 & + & 5x_2 & & & - & x_4 & = & -10 \\
-x_1 & - & 3x_2 & + & x_3 & + & x_4 & = & 12 \quad .
\end{array}$$

Capítulo 9

Problema geral

Os problemas canônicos primal e dual discutidos nos capítulos 7 e 8 são casos particulares do problema geral de programação linear. O problema consiste na otimização (maximização ou minimização) de uma função linear sujeita a restrições lineares.

9.1 Definição do problema

O **problema** (geral) **de programação linear**, ou **ppl**, consiste no seguinte: Dadas matrizes A_{11}, A_{12}, A_{13}, A_{21}, A_{22}, A_{23}, A_{31}, A_{32}, A_{33} e vetores b_1, b_2, b_3, c_1, c_2, c_3, encontrar vetores x_1, x_2, x_3 que ppl

$$\begin{array}{rrcrcrcl}
\text{minimizem a expressão} & c_1x_1 & + & c_2x_2 & + & c_3x_3 & & \\
\text{sujeita às restrições} & x_1 & & & & & \geq & o \\
& A_{11}x_1 & + & A_{12}x_2 & + & A_{13}x_3 & \geq & b_1 \\
& A_{21}x_1 & + & A_{22}x_2 & + & A_{23}x_3 & = & b_2 \\
& A_{31}x_1 & + & A_{32}x_2 & + & A_{33}x_3 & \leq & b_3 \\
& & & & & x_3 & \leq & o\,.
\end{array}$$

Estamos supondo que, para $i = 1, 2, 3$, as matrizes A_{i1}, A_{i2} e A_{i3} têm um mesmo conjunto M_i de índices de linhas, e que M_i também é o conjunto de índices de b_i. Analogamente, estamos supondo que o vetor c_j e as colunas de A_{1j}, A_{2j} e A_{3j} têm um mesmo conjunto N_j de índices.

A **matriz do ppl** é a matriz D que resulta da justaposição de $A_{11}, .., A_{33}$, b_1, .., b_3 e c_1, .., c_3, à maneira da seção 6.1 (o i-ésimo bloco de linhas de D é formado por $A_{i1}, A_{i2}, A_{i3}, b_i$ e a única linha do quarto bloco é $c_1, c_2, c_3, 0$). A **função objetivo** do problema é a função que leva qualquer terno de vetores x_1, x_2, x_3 no número $c_1x_1 + c_2x_2 + c_3x_3$. Esse número é o **custo** do terno x_1, x_2, x_3.

Embora o ppl tenha sido formulado como um problema de minimização, nossa definição inclui, implicitamente, problemas de maximização, uma vez que minimizar $c_1x_1 + c_2x_2 + c_3x_3$ é o mesmo que maximizar $-c_1x_1 - c_2x_2 - c_3x_3$.

Se todas as matrizes exceto A_{21} são vazias e todos os vetores exceto c_1 e b_2 são vazios então o ppl é o problema canônico primal $\mathsf{CP}(A_{21}, b_2, c_1)$. Se todas as matrizes exceto A_{32} são vazias e todos os vetores exceto c_2 e b_3 são vazios, então o ppl é o problema canônico dual $\mathsf{CD}(\widetilde{A_{32}}, b_3, -c_2)$.

9.2 Dualidade

Há uma fundamental relação de dualidade entre problemas de programação linear. Por definição, o **dual** de um ppl cuja matriz é D é o ppl cuja matriz é $-\widetilde{D}$, ou seja, a transposta de D com sinal trocado. Assim, o dual do ppl descrito na seção anterior é o ppl

dual

$$
\begin{array}{rrcrcrcl}
\text{minimizar a expressão} & -b_1y_1 & - & b_2y_2 & - & b_3y_3 & & \\
\text{sujeita às restrições} & \widetilde{y_1} & & & & & \geq & o \\
& -\widetilde{A_{11}}y_1 & - & \widetilde{A_{21}}y_2 & - & \widetilde{A_{31}}y_3 & \geq & -c_1 \\
& -\widetilde{A_{12}}y_1 & - & \widetilde{A_{22}}y_2 & - & \widetilde{A_{32}}y_3 & = & -c_2 \\
& -\widetilde{A_{13}}y_1 & - & \widetilde{A_{23}}y_2 & - & \widetilde{A_{33}}y_3 & \leq & -c_3 \\
& & & & & y_3 & \leq & o\,.
\end{array}
$$

É evidente que *o dual do dual de qualquer ppl* P *é* P .

Digamos que P é o ppl descrito na seção anterior e que D é o seu dual. É evidente que D também pode ser escrito assim:

$$
\begin{array}{rrcrcrcl}
\text{maximizar a expressão} & y_1b_1 & + & y_2b_2 & + & y_3b_3 & & \\
\text{sujeita às restrições} & & & & & y_3 & \leq & o \\
& y_1A_{11} & + & y_2A_{21} & + & y_3A_{31} & \leq & c_1 \\
& y_1A_{12} & + & y_2A_{22} & + & y_3A_{32} & = & c_2 \\
& y_1A_{13} & + & y_2A_{23} & + & y_3A_{33} & \geq & c_3 \\
& y_1 & & & & & \geq & o\,.
\end{array}
$$

Observe que o ppl D tem uma incógnita para cada restrição de P (exceto as restrições "$x_1 \geq o$" e "$x_3 \leq o$") e uma restrição para cada incógnita de P. Por exemplo, a incógnita y_1 corresponde à restrição $A_{11}x_1 + A_{12}x_2 + A_{13}x_3 \geq b_1$ de P; e a restrição $y_1A_{11} + y_2A_{21} + y_3A_{31} \leq c_1$ corresponde à incógnita x_1 de P.

Se todas as matrizes exceto A_{21} são vazias e todos os vetores exceto c_1 e b_2 são vazios então P é o problema canônico primal $\mathsf{CP}(A_{21}, b_2, c_1)$ e D é o problema canônico dual $\mathsf{CD}(A_{21}, c_1, b_2)$.

9.3 Lema da dualidade

Digamos que P é o ppl definido na seção 9.1 e D é o seu dual. O conjunto dos

P
D

$$\begin{array}{ll} \text{minimizar} & cx + dy \\ \text{sujeita a} & Ax + By = f\,, \quad l \le y \le u \quad \text{e} \quad x \le o\,. \end{array}$$

Figura 9.1: Exemplo de um ppl. As matrizes A e B são dadas. Os vetores c, d, f, l e u são dados. As incógnitas são x e y. Identifique as matrizes $A_{11}, .., A_{33}$ e os vetores b_1, b_2, b_3 e c_1, c_2, c_3.

$$\begin{array}{lll} \text{minimizar} & cx & \\ \text{sujeito a} & x \ge o \text{ e } Ax \ge b & \end{array} \qquad \begin{array}{lll} \text{maximizar} & yb & \\ \text{sujeito a} & y \ge o \text{ e } yA \le c & \end{array}$$

Figura 9.2: Verifique que o ppl à direita é o dual do ppl à esquerda.

$$\begin{array}{rcccccccl} & x_1 & & & & & & \ge & o \\ & A_{11}x_1 & + & A_{12}x_2 & + & A_{13}x_3 & & \ge & b_1 \\ & A_{21}x_1 & + & A_{22}x_2 & + & A_{23}x_3 & & = & b_2 \\ & A_{31}x_1 & + & A_{32}x_2 & + & A_{33}x_3 & & \le & b_3 \\ & & & & & x_3 & & \le & o \\ \min & c_1x_1 & + & c_2x_2 & + & c_3x_3 & & & \end{array}$$

$$\begin{array}{ccccccc} & & & & & & \max \\ & y_1A_{11} & y_1A_{12} & y_1A_{13} & y_1 & y_1b_1 \\ & + & + & + & & + \\ & y_2A_{21} & y_2A_{22} & y_2A_{23} & & y_2b_2 \\ & + & + & + & & + \\ y_3 & y_3A_{31} & y_3A_{32} & y_3A_{33} & & y_3b_3 \\ \le & \le & = & \ge & \ge & \\ o & c_1 & c_2 & c_3 & o & \end{array}$$

Figura 9.3: A primeira parte da figura é uma representação taquigráfica de um ppl. A segunda (leia na vertical, de cima para baixo) é uma representação do seu dual. As posições relativas dos símbolos A_{ij}, b_i e c_j não se alteram.

ternos x_1, x_2, x_3 de vetores que satisfazem as restrições do problema P será denotado por $X(b_1, b_2, b_3)$ e o conjunto dos ternos y_1, y_2, y_3 de vetores que satisfazem as restrições do problema D será denotado por $Y(c_1, c_2, c_3)$. Em linguagem geométrica, diz-se que esses conjuntos são **poliedros**.

$X(b_1, b_2, b_3)$
$Y(c_1, c_2, c_3)$

Lema 9.1 (da dualidade) *Para todo terno x_1, x_2, x_3 em $X(b_1, b_2, b_3)$ e todo terno y_1, y_2, y_3 em $Y(c_1, c_2, c_3)$ vale a desigualdade $c_1x_1 + c_2x_2 + c_3x_3 \geq y_1b_1 + y_2b_2 + y_3b_3$.*

DEMONSTRAÇÃO: Para todo terno x_1, x_2, x_3 e todo terno y_1, y_2, y_3,

$$\begin{aligned} c_1x_1 + c_2x_2 + c_3x_3 &\geq (y_1A_{11} + y_2A_{21} + y_3A_{31})\, x_1 + \\ &\quad (y_1A_{12} + y_2A_{22} + y_3A_{32})\, x_2 + \\ &\quad (y_1A_{13} + y_2A_{23} + y_3A_{33})\, x_3 \\ &= y_1\,(A_{11}x_1 + A_{12}x_2 + A_{13}x_3) + \\ &\quad y_2\,(A_{21}x_1 + A_{22}x_2 + A_{23}x_3) + \\ &\quad y_3\,(A_{31}x_1 + A_{32}x_2 + A_{33}x_3) \\ &\geq y_1b_1 + \\ &\quad y_2b_2 + \\ &\quad y_3b_3\,, \end{aligned}$$

como queríamos demonstrar. □

O lema (também conhecido como teorema fraco da dualidade) tem o seguinte corolário: se $c_1x_1 + c_2x_2 + c_3x_3 = y_1b_1 + y_2b_2 + y_3b_3$ então x_1, x_2, x_3 é solução do problema P e y_1, y_2, y_3 é solução do problema D.

9.4 Construção do dual

A definição do dual de um ppl é pesada. Felizmente, a demonstração do lema da dualidade pode ser usada para inferir a forma correta do dual de qualquer ppl P: escreva uma incógnita do dual para cada restrição do P e uma restrição do dual para cada incógnita do P; em seguida, procure as desigualdades mais brandas que assegurem a validade do lema da dualidade. Os seguintes exemplos ilustram a construção.

P

Primeiro exemplo. Suponha que o problema P consiste em encontrar números x_1, x_2, x_3, x_4 que minimizem a expressão [1]

$$11x_1 + 12x_2 + 13x_3 + 14x_4$$

[1] Note que aqui x_1, x_2 e x_3 são números e não vetores.

sob as restrições

$$\begin{aligned} 15x_1 + 16x_2 + 17x_3 + 18x_4 &\leq 19 \\ 20x_1 + 21x_2 + 22x_3 + 23x_4 &= 24 \\ 25x_1 + 26x_2 + 27x_3 + 28x_4 &\geq 29 \\ x_1 &\leq 0 \\ x_3 &\geq 0\,. \end{aligned}$$

Qual o dual de P? O dual terá três incógnitas: uma incógnita y_1 associada à primeira restrição, uma incógnita y_2 associada à segunda, etc. A função objetivo do dual será, portanto, $y_1 19 + y_2 24 + y_3 29$. Para todos os vetores x que satisfazem as restrições de P e todos os vetores y que satisfazem as restrições do seu dual devemos ter $11x_1 + 12x_2 + 13x_3 + 14x_4 \geq y_1 19 + y_2 24 + y_3 29$. Essa desigualdade entre as funções objetivo deverá valer em virtude da seqüência de relações

$$\begin{aligned} 11x_1 + 12x_2 + 13x_3 + 14x_4 &\geq (y_1 15 + y_2 20 + y_3 25)\, x_1 + \\ &\quad (y_1 16 + y_2 21 + y_3 26)\, x_2 + \\ &\quad (y_1 17 + y_2 22 + y_3 27)\, x_3 + \\ &\quad (y_1 18 + y_2 23 + y_3 28)\, x_4 \\ &= y_1\,(15x_1 + 16x_2 + 17x_3 + 18x_4) + \\ &\quad y_2\,(20x_1 + 21x_2 + 22x_3 + 23x_4) + \\ &\quad y_3\,(25x_1 + 26x_2 + 27x_3 + 28x_4) \\ &\geq y_1 19 + y_2 24 + y_3 29\,. \end{aligned}$$

As restrições mais brandas que garantem a primeira desigualdade são

$$\begin{aligned} 11 &\leq y_1 15 + y_2 20 + y_3 25\,, && \text{pois } x_1 \leq 0\,, \\ 12 &= y_1 16 + y_2 21 + y_3 26\,, && \text{pois } x_2 \text{ não tem restrição de sinal}\,, \\ 13 &\geq y_1 17 + y_2 22 + y_3 27\,, && \text{pois } x_3 \geq 0\,, \\ 14 &= y_1 18 + y_2 23 + y_3 28\,, && \text{pois } x_4 \text{ não tem restrição de sinal}\,. \end{aligned}$$

As restrições mais brandas que garantem a segunda desigualdade são

$$\begin{aligned} y_1 &\leq 0\,, && \text{pois } 15x_1 + 16x_2 + 17x_3 + 18x_4 \leq 19, \text{ e} \\ y_3 &\geq 0\,, && \text{pois } 25x_1 + 26x_2 + 27x_3 + 28x_4 \geq 29\,. \end{aligned}$$

Portanto, o dual de P consiste em encontrar números y_1, y_2, y_3 que maximizem a expressão $19y_1 + 24y_2 + 29y_3$ sujeita às restrições

$$\begin{aligned} 15y_1 + 20y_2 + 25y_3 &\geq 11 \\ 16y_1 + 21y_2 + 26y_3 &= 12 \\ 17y_1 + 22y_2 + 27y_3 &\leq 13 \\ 18y_1 + 23y_2 + 28y_3 &= 14 \\ y_1 &\leq 0 \\ y_3 &\geq 0\,. \end{aligned}$$

Segundo exemplo. Seja P o problema de encontrar vetores x e v que minimizem a expressão $cx + fv$ sujeita às restrições

$$Ax \leq b\,, \quad Dv = e \quad \text{e} \quad x \geq o\,.$$

(Aqui, A e D são matrizes dadas e c, f, b e e são vetores dados.) Qual o dual de P? O dual terá uma incógnita, digamos y, associada à restrição $Ax \leq b$ e uma outra incógnita, digamos w, associada à restrição $Dv = e$. A função objetivo do dual será, portanto, $yb + we$. As funções objetivo deverão satisfazer a relação $cx + fv \geq yb + we$. Essa desigualdade deverá valer em virtude da seqüência de relações

$$\begin{aligned} cx + fv &\geq (yA)x + (wD)v \\ &= y(Ax) + w(Dv) \\ &\geq yb + we\,. \end{aligned}$$

As restrições mais brandas sobre y e w que garantem todas essas desigualdades são

$$yA \leq c\,, \quad wD = f \quad \text{e} \quad y \leq o\,.$$

Portanto, o dual de P consiste em encontrar vetores y e w que maximizem a expressão $yb + we$ sujeita a essas restrições.

Terceiro exemplo. Seja P o problema de encontrar vetores y e w que maximizem $yb + we$ sujeita às restrições

$$yA \leq c\,, \quad wD = f \quad \text{e} \quad y \leq o\,.$$

O problema dual terá uma incógnita, digamos x, associada à restrição $yA \leq c$ e uma incógnita, digamos v, associada à restrição $wD = f$. A função objetivo do dual será $cx + fv$. As restrições do problema dual devem garantir a validade das relações

$$\begin{aligned} yb + we &\leq y(Ax) + w(Dv) \\ &= (yA)x + (wD)v \\ &\leq cx + fv\,. \end{aligned}$$

As restrições mais brandas sobre x e v que garantem todas estas desigualdades são

$$Ax \leq b\,, \quad Dv = e \quad \text{e} \quad x \geq o\,.$$

Portanto, o dual de P consiste em encontrar x e v que minimizem a expressão $cx + fv$ sujeita a essas restrições.

9.5 Teorema da dualidade

Retornemos aos problema P e D e aos poliedros $X(b_1, b_2, b_3)$ e $Y(c_1, c_2, c_3)$ da seção 9.3. O problema P é **viável** se $X(b_1, b_2, b_3)$ não é vazio, **inviável** se

$X(b_1, b_2, b_3)$ é vazio e **ilimitado** se para todo número ξ existe um terno x_1, x_2, x_3 em $X(b_1, b_2, b_3)$ tal que $c_1x_1 + c_2x_2 + c_3x_3 < \xi$. Se P é inviável ou ilimitado, é evidente que P não tem solução.

Definições análogas valem para o problema D: o problema é **viável** se $Y(c_1, c_2, c_3)$ não é vazio, **inviável** se $Y(c_1, c_2, c_3)$ é vazio e **ilimitado** se para todo número ξ existe um terno y_1, y_2, y_3 em $Y(c_1, c_2, c_3)$ tal que $y_1b_1 + y_2b_2 + y_3b_3 > \xi$.

O lema da dualidade tem o seguinte corolário óbvio: se um dos problemas do par P, D é ilimitado então o outro é inviável.

Um **vetor de inviabilidade** para o problema P é qualquer terno y'_1, y'_2, y'_3 em $Y(o, o, o)$ tal que $y'_1b_1 + y'_2b_2 + y'_3b_3 > 0$. Um **vetor de inviabilidade** para o problema D é qualquer terno x'_1, x'_2, x'_3 em $X(o, o, o)$ tal $c_1x'_1 + c_2x'_2 + c_3x'_3 < 0$. É fácil demonstrar o seguinte lema:

> **Lema 9.2** (da inviabilidade) *Se existe um vetor de inviabilidade para o problema* P *então* P *é inviável e* D *é inviável ou ilimitado. Se existe um vetor de inviabilidade para o problema* D *então* D *é inviável e* P *é inviável ou ilimitado.*

O teorema abaixo (também conhecido como teorema forte da dualidade) estende ao par P, D o teorema da dualidade 8.5.

> **Teorema 9.3** (da dualidade) *Vale uma e apenas uma das seguintes afirmativas:*
>
> *(1) existe um vetor de inviabilidade para* P *e existe um vetor de inviabilidade para* D*;*
> *(2)* D *é viável e existe um vetor de inviabilidade para* P*;*
> *(3)* P *é viável e existe um vetor de inviabilidade para* D*;*
> *(4) existem* x_1, x_2, x_3 *em* $X(b_1, b_2, b_3)$ *e* y_1, y_2, y_3 *em* $Y(c_1, c_2, c_3)$ *tais que* $c_1x_1 + c_2x_2 + c_3x_3 = y_1b_1 + y_2b_2 + y_3b_3$.

É claro que nos casos *(1)*, *(2)* e *(3)* os problemas P e D não têm solução; no caso *(4)*, x_1, x_2, x_3 é solução de P e y_1, y_2, y_3 é solução de D. A demonstração de que duas das afirmativas não podem ser simultaneamente verdadeiras é elementar: ela decorre dos lemas da inviabilidade e do lema da dualidade. A demonstração de que pelo menos uma das afirmativas é verdadeira não é tão simples: ela consiste em uma redução ao caso, já demonstrado na seção 8.4, em que os problemas P e D são canônicos. Para não tornar o texto ainda mais indigesto, vamos nos limitar a ilustrar o processo de redução com alguns exemplos. Faremos isso na próxima seção.

redução

9.6 Redução ao canônico primal

Todo ppl equivale a algum problema canônico primal. Em virtude dessa equivalência, qualquer algoritmo (como o Simplex) que resolva problemas canônicos

$$\begin{array}{cccc} 11 & 15 & 19 & 23 \\ 12 & 16 & 20 & 24 \\ 13 & 17 & 21 & 25 \\ 14 & 18 & 22 & \end{array}$$

$$\begin{array}{ccccccccc} 11 & 15 & 19 & -11 & -15 & -19 & 1 & 0 & 0 & 23 \\ 12 & 16 & 20 & -12 & -16 & -20 & 0 & 1 & 0 & 24 \\ 13 & 17 & 21 & -13 & -17 & -21 & 0 & 0 & 1 & 25 \\ 14 & 18 & 22 & -14 & -18 & -22 & 0 & 0 & 0 & \end{array}$$

Figura 9.4: O topo da figura especifica um sistema A, b, c. Considere o problema de minimizar cx sujeito a $Ax \leq b$. Esse problema equivale ao problema canônico $\mathsf{CP}(A', b', c')$, onde A', b', c' é o sistema representado na parte inferior da figura.

pode ser usado para resolver um ppl arbitrário. Os dois exemplos abaixo ilustram o fenômeno.

Primeiro exemplo. Considere o problema de encontrar um vetor x que minimize a expressão cx sujeita às restrições

$$Ax \leq b \tag{9.a}$$

(veja exemplo numérico na figura 9.4). Esse problema equivale ao seguinte problema canônico primal: encontrar vetores u, v e r que minimizem a expressão $cu - cv$ sujeita às restrições

$$Au - Av + Ir = b\,, \quad u \geq o\,, \quad v \geq o \quad \text{e} \quad r \geq o\,. \tag{9.b}$$

Eis a verificação da equivalência. Suponha que x satisfaz as restrições (9.a). Sejam u, v, r vetores definidos pelas equações

$$\begin{aligned} u_{[i]} &= \text{se } x_{[i]} \geq 0 \text{ então } x_{[i]} \text{ senão } 0 \\ v_{[i]} &= \text{se } x_{[i]} \geq 0 \text{ então } 0 \text{ senão } -x_{[i]} \\ r &= b - Ax\,. \end{aligned}$$

É claro que o terno u, v, r satisfaz as restrições (9.b). Ademais, x e o terno u, v, r atribuem o mesmo valor às correspondentes funções objetivo: cx é igual a $cu - cv$. Reciprocamente, se o terno u, v, r satisfaz as restrições (9.b) então o vetor $x = u - v$ satisfaz as restrições (9.a). Ademais, $cx = cu - cv$.

Segue dessa discussão que há uma correspondência biunívoca entre as soluções do primeiro problema e as soluções do segundo, e que soluções correspondentes dão o mesmo valor às funções objetivo. Portanto, os dois problemas são equivalentes.

A propósito, o dual do problema canônico (9.b) consiste em maximizar zb sujeito a $zA \leq c$, $zA \geq c$ e $z \leq o$. Não é de surpreender que esse problema seja equivalente ao dual do primeiro problema, que consiste em maximizar yb sujeito a $yA = c$ e $y \leq o$.

Segundo exemplo. Considere o problema de encontrar vetores x e z que maximizem a expressão $ax + dz$ sujeita às restrições

$$\begin{aligned} Ax + Bz &= b\,, \\ Cx + Dz &\le c\,, \\ x &\le o \end{aligned}$$

(veja exemplo numérico na figura 9.5). Esse problema equivale ao seguinte problema canônico primal: encontrar vetores y, v, w, r que minimizem a expressão $ay - dv + dw$ sujeita às restrições

$$\begin{aligned} -Ay + Bv - Bw &= b\,, \\ -Cy + Dv - Dw + Ir &= c\,, \\ y \ge o\,, \quad v \ge o\,, \quad w \ge o\,, \quad r &\ge o\,. \end{aligned}$$

Eis a verificação da equivalência. Para qualquer par x, z de vetores, sejam y, v, w e r os vetores definidos pelas equações

$$\begin{aligned} y &= -x \\ v_{[i]} &= \text{se } z_{[i]} \ge 0 \text{ então } z_{[i]} \text{ senão } 0 \\ w_{[i]} &= \text{se } z_{[i]} \ge 0 \text{ então } 0 \text{ senão } -z_{[i]} \\ r &= c - Cx - Dz\,. \end{aligned}$$

Se x, z satisfaz as restrições do primeiro problema então os vetores y, v, w, r satisfazem as restrições do segundo. Ademais, os vetores em questão dão o mesmo valor às correspondentes funções objetivo, a menos de uma troca de sinal: $ax + dz = -(ay - dv + dw)$ (a troca de sinal se deve à substituição de "maximizar" por "minimizar").

Reciprocamente, se y, v, w, r satisfazem as restrições do segundo problema então o par de vetores $-y, v - w$ satisfaz as restrições do primeiro. Ademais, $ay - dv + dw = -(a(-y) + d(v - w))$.

Em suma, há uma correspondência biunívoca entre as soluções do primeiro problema e as soluções do segundo, e soluções correspondentes dão o mesmo valor (a menos de uma troca de sinal) às funções objetivo.

9.7 Conclusão

O problema geral de programação linear consiste em encontrar, no conjunto de todos os vetores que satisfazem dadas restrições lineares, um vetor que minimize uma dada função linear. Os problemas canônicos primal e dual são casos particulares desse problema.

Os problemas de programação linear estão casados em pares: a cada problema corresponde um problema dual (se D é o dual de P então P é o dual de D). Um problema e o seu dual estão intimamente relacionados: ou ambos

encontrar números x_i e z_i que maximizem

$$11x_1 + 16x_2 + 21z_1 + 26z_2 \text{ sujeito a}$$

$$\begin{aligned}
12x_1 + 17x_2 + 22z_1 + 27z_2 &= 32\\
13x_1 + 18x_2 + 23z_1 + 28z_2 &= 33\\
14x_1 + 19x_2 + 24z_1 + 29z_2 &\le 34\\
15x_1 + 20x_2 + 25z_1 + 30z_2 &\le 35\\
x_1 &\le 0\\
x_2 &\le 0
\end{aligned}$$

encontrar números não-negativos y_i, v_i, w_i e r_i que minimizem

$$11y_1 + 16y_2 - 21v_1 + 21w_1 - 26v_2 + 26w_2 \text{ sujeito a}$$

$$\begin{aligned}
-12y_1 - 17y_2 + 22v_1 - 22w_1 + 27v_2 - 27w_2 &= 32\\
-13y_1 - 18y_2 + 23v_1 - 23w_1 + 28v_2 - 28w_2 &= 33\\
-14y_1 - 19y_2 + 24v_1 - 24w_1 + 29v_2 - 29w_2 + 1r_1 + 0r_2 &= 34\\
-15y_1 - 20y_2 + 25v_1 - 25w_1 + 30v_2 - 30w_2 + 0r_1 + 1r_2 &= 35
\end{aligned}$$

Figura 9.5: O ppl no topo da figura e o problema canônico primal na parte inferior da figura são equivalentes.

são inviáveis, ou um é inviável e o outro é ilimitado, ou ambos têm solução. Esse fato é a essência do teorema da dualidade.

Qualquer problema de programação linear pode ser reduzido a um problema canônico primal equivalente. O algoritmo Simplex (ou qualquer outro algoritmo que resolva problemas canônicos primais) pode então ser usado para resolver o problema reduzido.

9.8 Apêndice: Uma interpretação do dual

O seguinte exemplo procura ilustrar o significado das incógnitas (ou variáveis) duais. Considere o problema de encontrar números não-negativos x_1, x_2, x_3, x_4 que minimizem a expressão $11x_1 + 12x_2 + 13x_3 + 14x_4$ sob as restrições

$$\begin{aligned}
15x_1 + 16x_2 + 17x_3 + 18x_4 &\ge 19\\
20x_1 + 21x_2 + 22x_3 + 23x_4 &\ge 24\\
25x_1 + 26x_2 + 27x_3 + 28x_4 &\ge 29\,.
\end{aligned}$$

Esse problema pode estar modelando a tarefa de um nutricionista: encontrar quantidades x_1, x_2, x_3, x_4 de pacotes de carne, batatas, leite e goiabada respectivamente que minimizem o custo $11x_1 + 12x_2 + 13x_3 + 14x_4$ de uma refeição que contenha pelo menos 19 gramas de cálcio, pelo menos 24 gramas de carboidratos, e pelo menos 29 gramas de proteínas. É claro que a primeira restrição dá o conteúdo de cálcio nos quatro alimentos (15 gramas por pacote de carne,

16 gramas por pacote de batatas, etc.), a segunda restrição dá o conteúdo de carboidratos, etc.

O dual do nosso problema consiste em encontrar números não-negativos y_1, y_2, y_3 que maximizem $19y_1 + 24y_2 + 29y_3$ sob as restrições

$$\begin{aligned} 15y_1 + 20y_2 + 25y_3 &\leq 11 \\ 16y_1 + 21y_2 + 26y_3 &\leq 12 \\ 17y_1 + 22y_2 + 27y_3 &\leq 13 \\ 18y_1 + 23y_2 + 28y_3 &\leq 14\,. \end{aligned}$$

O dual pode ser interpretado como segue. Um fabricante de cálcio em pó, carboidratos em pó e proteínas em pó quer lançar seus produtos no mercado a preços y_1 por grama de cálcio, y_2 por grama de carboidrato e y_3 por grama de proteína. Para competir com os produtos naturais, o preço de um pacote de carne artificial (composta por 15 gramas de cálcio, 20 gramas de proteína e 25 gramas de carboidratos) não pode ser superior ao preço de um pacote de carne natural. Analogamente, os preços das batatas, leite e goiabada artificiais não podem ultrapassar os preços dos correspondentes produtos naturais. Satisfeitas estas restrições, o fabricante quer maximizar o preço $y_1 19 + y_2 24 + y_3 29$ de uma refeição que contenha as quantidades mínimas dos três nutrientes.

É fácil verificar (veja o exercício 9.7, página 98) que ambos os problemas são viáveis. O teorema da dualidade garante então que ambos os problemas têm solução e que $11x_1+12x_2+13x_3+14x_4 = 19y_1+24y_2+29y_3$ para qualquer solução x_1, x_2, x_3, x_4 do primeiro problema e qualquer solução y_1, y_2, y_3 do segundo.

Exercícios

9.1 Considere o problema de encontrar vetores x e z tais que $x \geq o$, $Ax+Dz \geq b$ e $cx + fz$ é mínimo. Verifique que o dual deste problema consiste em encontrar um vetor y tal que $y \geq o$, $yA \leq c$, $yD = f$ e yb é máximo.

9.2 Seja X o conjunto dos vetores $x \geq o$ tais $Ex \geq f$ e $Ax = b$. Seja Y o conjunto dos pares y, w tais que $y \leq o$ e $yE + wA \geq c$. Prove que para todo x em X e todo par y, w em Y tem-se $cx \leq yf + wb$.

9.3 Demonstre o lema da inviabilidade 9.2.

9.4 Seja P o problema de encontrar vetores x e u que maximizem a expressão $cx + du$ sujeita às restrições $Ax + Bu = d$ e $o \leq u \leq e$. Mostre que P é inviável se existem vetores y' e z' tais que $y'A = o$, $y'B + z' \geq o$, $z' \geq o$ e $y'd + z'e < 0$. Mostre que um par y', z' como o que acabamos de descrever certamente existe se e tem algum componente negativo. Mostre que o dual de P é inviável se existe um vetor x' tal que $Ax' = o$ e $cx' > 0$.

9.5 Suponha dada uma matriz A, vetores b e c e um número β. Seja R o conjunto dos vetores y que satisfazem as restrições $yA \leq c$ e $yb > \beta$. Seja S o conjunto dos vetores x que satisfazem as restrições $Ax = b$, $x \geq o$

e $cx \leq \beta$. Seja T o conjunto dos vetores x que satisfazem as restrições $Ax = o$, $x \geq o$ e $cx < 0$. Mostre que R é vazio se e só se $S \cup T$ não é vazio.

9.6 Seja S o conjunto dos vetores x tais que $Ax > o$, isto é, $A_{[i,\,]}\, x > 0$ para todo i. Seja T o conjunto dos vetores não-nulos y tais que $y \geq o$ e $yA = o$. Mostre que S é vazio se e só se T não é vazio.

9.7 Seja P o problema que consiste em minimizar cx sujeito a $Ax \geq b$ e $x \geq o$. Mostre que o dual de P consiste em maximizar yb sujeito a $yA \leq c$ e $y \geq o$. Mostre que se $c \geq o$, $b \geq o$, $A \geq O$ e $A_{[i,\,]} \neq o$ para todo i então ambos os problemas têm solução.

Parte III

Algoritmos para Dados Inteiros

Capítulo 10

Determinantes

Os próximos capítulos pretendem examinar as "variantes inteiras" do algoritmo de Gauss-Jordan e do algoritmo Simplex. A análise dessas variantes depende do conceito de determinante.[1] Embora o conceito seja bem conhecido, convém fazer uma revisão.

O determinante de uma matriz quadrada A é um número da forma $\sigma - \sigma'$, onde σ e σ' são somas de produtos de elementos de A e cada produto contém exatamente um elemento de cada linha e de cada coluna de A. Por exemplo, o determinante da matriz

$$\begin{matrix} \alpha & \beta & \gamma \\ \delta & \varepsilon & \zeta \\ \eta & \theta & \iota \end{matrix}$$

é $\alpha\varepsilon\iota + \beta\zeta\eta + \gamma\delta\theta - \eta\varepsilon\gamma - \alpha\zeta\theta - \beta\delta\iota$. A propriedade mais importante dos determinantes é a "lei do produto":

$$\det(AB) = \det(A)\det(B)\,.$$

O conceito de determinante será definido em duas etapas. Primeiro, definiremos o determinante de matrizes de permutação e mostraremos que este satisfaz a "lei do produto". Depois, usaremos o determinante de matrizes de permutação para definir o determinante de uma matriz quadrada arbitrária.

10.1 Sinal de uma matriz de permutação

Uma matriz **de permutação** é uma matriz de bijeção cujos conjuntos de índices de linhas e colunas são idênticos. É evidente que o produto de duas matrizes de permutação é uma matriz de permutação.

Um **arco** de uma matriz de permutação J sobre $M \times M$ é qualquer par ordenado $\langle j, i\rangle$ de elementos de M tal que $J_{[i,j]} = 1$. Um **circuito** em J é uma

[1] Veja a nota histórica *Matrices and Determinants* na *MacTutor History of Mathematics Archive* (University of St Andrews, Escócia).

seqüência $\langle k_1, \ldots, k_n\rangle$ de elementos de M, distintos dois a dois, tal que

$$\langle k_1, k_2\rangle,\ \langle k_2, k_3\rangle,\ \ldots\ , \langle k_{n-1}, k_n\rangle \quad \text{e} \quad \langle k_n, k_1\rangle$$

são arcos de J. Por exemplo, se $J_{[i,i]} = 1$ para algum i então a seqüência $\langle i\rangle$ é um circuito. Circuitos que diferem apenas por uma rotação — como, por exemplo, $\langle k_1, k_2, k_3\rangle$, $\langle k_2, k_3, k_1\rangle$ e $\langle k_3, k_1, k_2\rangle$ — são considerados idênticos. Feitas essas convenções, fica claro que cada elemento de M pertence a um e um só circuito de J. O número de circuitos de J será denotado por

$$\mathrm{circ}(J)\,.$$

É óbvio que $1 \leq \mathrm{circ}(J) \leq |M|$. É óbvio também que $\mathrm{circ}(J) = |M|$ se e só se J é a matriz identidade.

Uma **matriz de transposição** é uma matriz de permutação T tal que $\mathrm{circ}(T) = |M| - 1$. Em outras palavras, T é de transposição se existem elementos distintos, i e j, de M tais que $\langle j, i\rangle$ e $\langle i, j\rangle$ são arcos e todos os demais arcos de T têm a forma $\langle k, k\rangle$. Diremos que uma tal matriz transpõe i e j. transposição

Lema 10.1 *Seja T uma matriz de transposição que transpõe i e j e J uma matriz de permutação arbitrária. Se i e j pertencem ao mesmo circuito de J então* $\mathrm{circ}(TJ) = \mathrm{circ}(J) + 1$*; caso contrário,* $\mathrm{circ}(TJ) = \mathrm{circ}(J) - 1$.

DEMONSTRAÇÃO: É fácil entender o efeito da multiplicação de J por T: como $(TJ)_{[j,\,]} = J_{[i,\,]}$, um arco de J que tem a forma $\langle k, i\rangle$ é transformado no arco $\langle k, j\rangle$ de TJ; analogamente, um arco de J que tem a forma $\langle k, j\rangle$ é transformado no arco $\langle k, i\rangle$ de TJ.

Suponha que i e j pertencem a um mesmo circuito, digamos $\langle k_1, \ldots, k_n\rangle$, de J. Ajuste a notação de modo que $k_1 = i$. Seja p um índice tal que $k_p = j$. Então p

$$\langle k_1, k_2, \ldots, k_{p-1}\rangle \quad \text{e} \quad \langle k_p, k_{p+1}, \ldots, k_n\rangle$$

são circuitos[2] de TJ. Todos os demais circuitos de J também são circuitos de TJ. Logo, $\mathrm{circ}(TJ) = \mathrm{circ}(J) + 1$. Suponha agora que i e j pertencem a circuitos distintos, digamos $\langle h_1, \ldots, h_m\rangle$ e $\langle k_1, \ldots, k_n\rangle$, de J. Ajuste a notação de modo que $h_1 = i$ e $k_1 = j$. Então

$$\langle h_1, h_2, \ldots, h_m, k_1, k_2, \ldots, k_n\rangle$$

é um circuito[3] de TJ. Todos os demais circuitos de J também são circuitos de TJ. Logo, $\mathrm{circ}(TJ) = \mathrm{circ}(J) - 1$. □

O valor de $\mathrm{circ}(J)$ é menos importante que a relação entre a paridade de $\mathrm{circ}(J)$ e a de $|M|$. Diremos que uma matriz de permutação J é **positiva** se matriz positiva

[2] Se $n = 2$, então $p = 2$ e, portanto, esses circuitos são $\langle k_1\rangle$ e $\langle k_2\rangle$.

[3] Se $m = n = 1$, então esse circuito é $\langle h_1, k_1\rangle$.

$$\begin{matrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

Figura 10.1: Se J é a matriz da figura então $\mathrm{circ}(J) = 2$ e portanto $\mathrm{sig}(J) = +1$.

$$|M| - \mathrm{circ}(J)$$

é par e **negativa** em caso contrário. A matriz identidade, por exemplo, é positiva; e toda matriz de transposição é negativa. O **sinal** de uma matriz de permutação J será denotado por negativa

$$\mathrm{sig}(J)\,.$$

Assim, $\mathrm{sig}(J) = +1$ se J é positiva e $\mathrm{sig}(J) = -1$ se J é negativa.

O lema 10.1 mostra que $\mathrm{sig}(TJ) = -\mathrm{sig}(J)$ para qualquer matriz de transposição T e qualquer matriz de permutação J. O teorema abaixo generaliza essa relação, mostrando que sig satisfaz a "lei do produto".

Teorema 10.2 *Para todo par J, K de matrizes de permutação,* $\mathrm{sig}(JK) = \mathrm{sig}(J)\,\mathrm{sig}(K)$.

DEMONSTRAÇÃO: Nossa demonstração é uma indução no número $\mathrm{circ}(J)$. Suponha inicialmente que $\mathrm{circ}(J) = |M|$. Então a proposição vale trivialmente pois $J = I$ e portanto $\mathrm{sig}(J) = +1$.

Suponha agora que $\mathrm{circ}(J) < |M|$. Então J possui um circuito $\langle k_1, .., k_n\rangle$ com $n \geq 2$. Seja T a matriz de transposição que transpõe k_1 e k_2. De acordo com o lema 10.1, $\mathrm{circ}(TJ) = \mathrm{circ}(J) + 1$. Podemos pois supor, a título de hipótese de indução, que T

$$\mathrm{sig}(TJK) = \mathrm{sig}(TJ)\,\mathrm{sig}(K)\,.$$

Por outro lado, de acordo com o lema 10.1, temos $\mathrm{sig}(TJ) = -\mathrm{sig}(J)$ e também $\mathrm{sig}(TJK) = -\mathrm{sig}(JK)$. Logo, $\mathrm{sig}(JK) = \mathrm{sig}(J)\mathrm{sig}(K)$, como queríamos demonstrar. □

A demonstração do teorema revela, em particular, que toda matriz de permutação tem a forma $T_1 \cdot \ldots \cdot T_p I$, onde $T_1, .., T_p$ são matrizes de transposição. É evidente que uma tal matriz é positiva se e só se p é par.

10.2 Determinante de matriz quadrada

Uma matriz é **quadrada** se seus conjuntos de índices de linhas e colunas são matriz quadrada

idênticos. Em outras palavras, uma matriz A sobre $M \times N$ é quadrada se $M = N$. As definições abaixo aplicam-se apenas a matrizes quadradas.

Para qualquer matriz A sobre $M \times M$, vamos denotar por $\mathrm{diag}(A)$ o produto dos elementos da diagonal de A:

$$\mathrm{diag}(A) \;=\; \textstyle\prod_{i \in M} A_{[i,i]}\,.$$

O **determinante** de uma matriz quadrada A sobre $M \times M$ é o número $\sum_J \mathrm{sig}(J)\,\mathrm{diag}(AJ)$, em que a soma se estende a todas as matrizes de permutação J sobre $M \times M$. O determinante de A é denotado por $\det(A)$. Portanto,

$$\det(A) = \textstyle\sum_J \mathrm{sig}(J)\,\mathrm{diag}(AJ)\,.$$

É claro que $\det(A) = \sum_J \mathrm{diag}(AJ) - \sum_K \mathrm{diag}(AK)$, em que a primeira soma se refere a todas as matrizes de permutação positivas e a segunda se refere a todas as matrizes de permutação negativas.

A definição de determinante não se altera se a expressão $\mathrm{diag}(AJ)$ for substituída pela expressão

$$\mathrm{diag}(JA)\,,$$

uma vez que essas duas expressões têm o mesmo valor. De fato, $\mathrm{diag}(AJ) = \prod_i A_{[i,\,]}J_{[\,,i]} = \prod_i A_{[i,\varphi i]}$, onde φ é a bijeção de M em M definida pela equação $J_{[\varphi i,\,i]} = 1$. Se denotarmos por ψ a inversa de φ teremos $\prod_i A_{[i,\varphi i]} = \prod_j A_{[\psi j,\,j]} = \prod_j J_{[j,\,]}A_{[\,,j]} = \mathrm{diag}(JA)$.

$$\begin{array}{ccc} \alpha & \beta & \gamma \\ \delta & \varepsilon & \zeta \\ \eta & \theta & \iota \end{array} \qquad\qquad \begin{array}{ccc} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{array}$$

Figura 10.2: Digamos que A é a primeira das matrizes da figura e J é a segunda. Observe que $\mathrm{diag}(A\widetilde{J})$ é o produto dos elementos de A que estão nas posições correspondentes aos elementos não-nulos de J.

É muito fácil verificar as seguintes propriedades do determinante de uma matriz quadrada A: se A é de permutação então $\det(A) = \mathrm{sig}(A)$; se A é diagonal então

$$\det(A) = \mathrm{diag}(A)\,;$$

se $A_{[m,\,]} = o$ ou $A_{[\,,m]} = o$ para algum m então

$$\det(A) = 0\,.$$

Também é fácil verificar que para qualquer matriz diagonal D

$$\det(DA) = \mathrm{diag}(D)\,\det(A)\,.$$

Em particular, para qualquer número δ, $\det(\delta A) = \delta^m \det(A)$, onde m é o número de linhas de A.

Como calcular o determinante de uma matriz? Embora a definição envolva uma soma de $m!$ parcelas (pois este é o número de matrizes de permutação com m linhas), o determinante pode ser calculado com esforço proporcional a m^3, como veremos num próximo capítulo.[4]

10.3 Três propriedades básicas

Esta seção reune três propriedades básicas dos determinantes. No enunciado de todas as propriedades, A é uma matriz sobre $M \times M$.

Propriedade 10.3 (desenvolvimento por uma linha) *Para qualquer m em M, se $A_{[m,\,M-m]} = o$ então o determinante de A é o produto de $A_{[m,\,m]}$ pelo determinante de $A_{[M-m,\,M-m]}$.*

DEMONSTRAÇÃO: Suponha que $A_{[m,\,M-m]}$ é nulo. Seja J uma matriz de permutação sobre $M \times M$. Se $J_{[m,\,m]} = 0$ então $A_{[m,\,]}J_{[\,,m]} = 0$ e portanto $\operatorname{diag}(AJ) = \prod_i A_{[i,\,]}J_{[\,,i]} = 0$. Caso contrário,

$$\operatorname{diag}(AJ) = A_{[m,\,m]} \operatorname{diag}(BK)\,,$$

onde $B = A_{[M-m,\,M-m]}$ e $K = J_{[M-m,\,M-m]}$. Como $\operatorname{sig}(K) = \operatorname{sig}(J)$, temos

$$\textstyle\sum_J \operatorname{sig}(J) \operatorname{diag}(AJ) \;=\; A_{[m,\,m]} \sum_K \operatorname{sig}(K) \operatorname{diag}(BK)\,,$$

onde K percorre o conjunto das matrizes de permutação sobre $M-m \times M-m$. Logo, $\det(A) = A_{[m,\,m]} \det(B)$. □

Vale também a propriedade "transposta": se $A_{[M-m,\,m]}$ é nulo então $\det(A) = A_{[m,\,m]} \det(A_{[M-m,\,M-m]})$. A demonstração é análoga, essencialmente porque $\operatorname{diag}(JA) = \operatorname{diag}(AJ)$.

A propriedade seguinte é um caso particular da "lei do produto" que demonstraremos na próxima seção.

Propriedade 10.4 (multiplicação por matriz de permutação) *Para qualquer matriz de permutação K sobre $M \times M$, $\det(KA) = \operatorname{sig}(K)\ \det(A)$.*

DEMONSTRAÇÃO: Por definição, $\det(KA) = \sum_J \operatorname{sig}(J) \det(JKA)$. Em virtude do teorema 10.2, $\operatorname{sig}(J) = \operatorname{sig}(K)\operatorname{sig}(JK)$. Logo,

$$\begin{aligned}
\det(KA) &= \textstyle\sum_J \operatorname{sig}(K)\operatorname{sig}(JK)\operatorname{diag}(JKA) \\
&= \operatorname{sig}(K) \textstyle\sum_J \operatorname{sig}(JK)\operatorname{diag}(JKA) \\
&= \operatorname{sig}(K) \det(A)\,,
\end{aligned}$$

[4] A título de curiosidade, o **permanente** de uma matriz quadrada A é o número $\sum_J \operatorname{diag}(AJ)$, onde a soma se estende a todas as matrizes de permutação J. Não se conhece um algoritmo eficiente — isto é, polinomial no número de componentes de A — para o cálculo do permanente.

uma vez que quando J percorre o conjunto de todas as matrizes de permutação a matriz JK também percorre o conjunto de todas as matrizes de permutação. □

Propriedade 10.5 (matriz com duas linhas iguais) *Para dois elementos distintos h e k de M, se $A_{[h,\,]} = A_{[k,\,]}$ então $\det(A) = 0$.*

DEMONSTRAÇÃO: Seja T a matriz de transposição que transpõe h e k. Como $TA = A$, temos

$$\det(TA) = \det(A)\,.$$

Por outro lado, $\det(TA) = \operatorname{sig}(T)\det(A)$, em virtude da propriedade 10.4. Como $\operatorname{sig}(T) = -1$, temos

$$\det(TA) = -\det(A)\,.$$

Essas equações só podem ser simultaneamente verdadeiras se $\det(A)$ for nulo. □

Vale também a propriedade "transposta": se $A_{[\,,h]} = A_{[\,,k]}$ para dois elementos distintos h e k de M então $\det(A) = 0$.

10.4 Determinante do produto de matrizes

Esta seção mostra que para quaisquer matrizes quadradas A e B vale a identidade $\det(AB) = \det(A)\det(B)$. A demonstração será feita por etapas: primeiro mostraremos que a identidade vale quando A é quase-identidade; daí deduziremos a identidade no caso em que A é elementar; finalmente, cuidaremos do caso geral. As duas etapas finais são análogas ao algoritmo de Gauss-Jordan (veja capítulo 2).

quase-identidade

Uma matriz A sobre $M \times M$ é **quase-identidade** se existem elementos h e k de M tais que $A_{[i,j]} = I_{[i,j]}$ para todo par i, j distinto do par h, k. Diremos que esta é uma matriz quase-identidade **do tipo** h, k. É claro que toda matriz quase-identidade do tipo h, k é elementar.

Lema 10.6 (multiplicação por matriz quase-identidade) *Se A é uma matriz quase-identidade do tipo h, k então*

$$\begin{aligned} \det(AB) &= A_{[h,h]}\det(B) \quad \textit{se } h = k \textit{ e} \\ \det(AB) &= \det(B) \quad \textit{em caso contrário,} \end{aligned}$$

qualquer que seja a matriz B.

DEMONSTRAÇÃO: Se $h = k$ então A é uma matriz diagonal e portanto $\det(AB) = \operatorname{diag}(A)\det(B)$ enquanto $\operatorname{diag}(A) = A_{[h,h]}$.

Suponha agora que $h \neq k$. Seja α o número $A_{[h,k]}$. É claro que $(AB)_{[i,\,]} = B_{[i,\,]}$ para todo i distinto de h e $(AB)_{[h,\,]} = B_{[h,\,]} + \alpha B_{[k,\,]}$. Para toda matriz

de permutação J,

$$\begin{aligned}
\operatorname{diag}(ABJ) &= \textstyle\prod_i (AB)_{[i,\,]} J_{[\,,i]} \\
&= \textstyle\prod_i B_{[i,\,]} J_{[\,,i]} \;+\; \alpha B_{[k,\,]} J_{[\,,h]} \prod_{i\neq h} B_{[i,\,]} J_{[\,,i]} \\
&= \operatorname{diag}(BJ) \;+\; \alpha \check{B}_{[h,\,]} J_{[\,,h]} \prod_{i\neq h} \check{B}_{[i,\,]} J_{[\,,i]} \\
&= \operatorname{diag}(BJ) \;+\; \alpha \operatorname{diag}(\check{B}J)\,,
\end{aligned}$$

onde $\check{B}$ é a matriz definida pelas equações $\check{B}_{[M-h,\,]} = B_{[M-h,\,]}$ e $\check{B}_{[h,\,]} = B_{[k,\,]}$. Portanto,

$$\begin{aligned}
\det(AB) &= \textstyle\sum_J \operatorname{sig}(J) \operatorname{diag}(ABJ) \\
&= \textstyle\sum_J \operatorname{sig}(J) \operatorname{diag}(BJ) \;+\; \alpha \sum_J \operatorname{sig}(J) \operatorname{diag}(\check{B}J) \\
&= \det(B) + \alpha \det(\check{B})\,.
\end{aligned}$$

Como $\check{B}$ tem duas linhas iguais, a propriedade 10.5 garante que $\det(\check{B}) = 0$. Portanto, $\det(AB) = \det(B)$. □

Lema 10.7 (multiplicação por matriz elementar) *Se A é uma matriz elementar com coluna saliente h então $\det(AB) = A_{[h,h]} \det(B)$ qualquer que seja a matriz B.*

DEMONSTRAÇÃO: Seja M o conjunto de índices de linhas e colunas de A. A proposição é trivialmente verdadeira se $A_{[h,h]} = 0$, pois nesse caso $A_{[h,\,]}$ é nulo, donde $(AB)_{[h,\,]}$ é nulo, e portanto $\det(AB) = 0$.

Suponha no que segue que $A_{[h,h]} \neq 0$. Diremos que a **complexidade** de A é o número de componentes não-nulos do vetor $A_{[M-h,h]}$. Nossa demonstração prossegue por indução na complexidade de A e depende do lema 10.6.

Suponha que a complexidade de A é nula, ou seja, que o vetor $A_{[M-h,h]}$ é nulo. Então A é uma matriz quase-identidade do tipo h,h. Pelo lema 10.6, $\det(AB) = A_{[h,h]} \det(B)$, como queríamos demonstrar.

Suponha agora que $A_{[i,h]} \neq 0$ para algum i em $M - h$. Seja $\check{G}$ a matriz quase-identidade do tipo i,h definida pela equação $\check{G}$

$$\check{G}_{[i,h]} \;=\; -A_{[i,h]}/A_{[h,h]}$$

(veja a análise de algoritmo de Gauss-Jordan, seção 2.4) e seja A' a matriz $\check{G}A$. É fácil verificar que a matriz A' é elementar com coluna saliente h e que $A'_{[i,h]} = 0$ e $A'_{[M-i,h]} = A_{[M-i,h]}$. Como a complexidade de A' é menor que a complexidade de A podemos pois supor, a título de hipótese de indução, que A'

$$\det(A'B) = A'_{[h,h]} \det(B)\,.$$

Como $\check{G}$ é quase-identidade, o lema 10.6 garante que

$$\det(A'B) \;=\; \det(\check{G}AB) \;=\; \det(AB)\,.$$

Finalmente, como $A'_{[h,h]} = A_{[h,h]}$, temos $\det(AB) = A_{[h,h]} \det(B)$. □

É claro que vale a propriedade "transposta": $\det(AB) = A_{[h,h]} \det(B)$ para qualquer matriz elementar A com linha saliente h e qualquer matriz B.

Teorema 10.8 (do produto de determinantes[5]) *Para quaisquer matrizes A e B sobre $M \times M$,*

$$\det(AB) \;=\; \det(A)\det(B)\,.$$

DEMONSTRAÇÃO: A demonstração deste teorema é em tudo análoga à análise do algoritmo de Gauss-Jordan.

Digamos que a **simplicidade** de uma matriz A sobre $M \times M$ é a cardinalidade do maior subconjunto P de M dotado da seguinte propriedade: existe uma parte Q de M tal que $A_{[P,Q]}$ é de bijeção e $A_{[M-P,Q]}$ é nula. Suponha inicialmente que a simplicidade de A é $|M|$, ou seja, que A é uma matriz de permutação. Então, pela propriedade 10.4, $\det(AB) = \mathrm{sig}(A)\det(B)$. Como $\mathrm{sig}(A) = \det(A)$, temos a identidade desejada.

Suponha agora que a simplicidade de A é menor que $|M|$. Sejam P e Q partes de M tais que $A_{[P,Q]}$ é de bijeção, $A_{[M-P,Q]}$ é nula e $|P|$ é igual à simplicidade de A. Seja h um elemento de $M - P$. Há dois casos a considerar: P Q h

CASO 1: $A_{[h,\,]}$ é nulo. É claro que $\det(A)$ e $\det(AB)$ são nulos, uma vez que $(AB)_{[h,\,]} = A_{[h,\,]}B = o$. Portanto, a identidade $\det(AB) = \det(A)\det(B)$ vale trivialmente neste caso.

CASO 2: $A_{[h,\,]}$ não é nulo. Seja k um elemento de $M - Q$ tal que $A_{[h,k]} \neq 0$. Seja $\check{F}$ a matriz elementar com coluna saliente h definida pela equação k $\check{F}$

$$\check{F}_{[\,,h]} = A_{[\,,k]}$$

(veja a análise de algoritmo de Gauss-Jordan). Seja $\check{G}$ a matriz elementar com coluna saliente h definida pelas equações $\check{G}$

$$\check{G}_{[i,h]} = \alpha_i\,,$$

onde $\alpha_h = 1/A_{[h,k]}$ e $\alpha_i = -A_{[i,k]}/A_{[h,k]}$ para todo i em $M - h$. É fácil verificar que

$$\check{F}\check{G} = \check{G}\check{F} = I\,.$$

Seja A' a matriz $\check{G}A$. Então $A'_{[h,\,]} = \alpha_h A_{[h,\,]}$ e $A'_{[i,\,]} = A_{[i,\,]} + \alpha_i A_{[h,\,]}$ para cada i em $M - h$. Como $A_{[h,Q]}$ é nulo, temos A'

$$A'_{[\,,Q]} = A_{[\,,Q]}\,.$$

Por outro lado, $A'_{[i,k]} = \check{G}_{[i,\,]}A_{[\,,k]} = \check{G}_{[i,\,]}\check{F}_{[\,,h]} = (\check{G}\check{F})_{[i,h]} = I_{[i,h]}$ para cada i em M, donde

$$A'_{[\,,k]} = I_{[\,,h]}\,.$$

[5] Publicado em 1812, simultaneamente por Jacques P. M. Binet (1786–1856) e Augustin-Louis Cauchy (1789–1857).

Segue daí que $A'_{[P+h,\,Q+k]}$ é uma matriz de bijeção e $A'_{[M-P-h,\,Q+k]}$ é nula. Portanto, a simplicidadade de A' é maior que a simplicidade de A. Podemos pois supor, a título de hipótese de indução, que

$$\det(A'B) = \det(A')\det(B)\,.$$

Como $\check{F}A'B = AB$ e $\check{F}$ é elementar, o lema 10.7 garante que

$$\det(AB) = \check{F}_{[h,\,h]}\det(A'B) = \check{F}_{[h,\,h]}\det(A')\det(B)\,.$$

Como $A' = \check{G}A$ e $\check{G}$ é elementar, o lema 10.7 garante que $\det(A') = \check{G}_{[h,\,h]}\det(A)$. Portanto,

$$\det(AB) = \check{F}_{[h,\,h]}\check{G}_{[h,\,h]}\det(A)\det(B)\,.$$

Como $\check{F}_{[h,\,h]} = 1/\alpha_h$ e $\check{G}_{[h,\,h]} = \alpha_h$, temos a identidade desejada. □

10.5 Delimitação do determinante

É importante estabelecer uma delimitação superior para o valor absoluto de $\det(A)$, que denotaremos por $\mathrm{absdet}(A)$. $\mathrm{absdet}(A)$

Delimitação 10.9 *Para qualquer matriz A sobre $M \times M$,*

$$\mathrm{absdet}(A) \;\le\; \textstyle\prod_{i\in M}\sum_{j\in M}\alpha_{ij}\,,$$

onde α_{ij} denota o valor absoluto de $A_{[i,\,j]}$. α_{ij}

DEMONSTRAÇÃO: Há uma correspondência biunívoca óbvia entre matrizes de permutação sobre $M \times M$ e bijeções de M em M: uma matriz de permutação J e uma bijeção φ se correspondem se

$$J_{[i,\,\varphi i]} = 1 \text{ para todo } i \text{ em } M\,.$$

Para toda bijeção φ de M em M, podemos definir o número $\mathrm{sig}(\varphi)$ com sendo $\mathrm{sig}(J)$, onde J é a matriz de permutação que corresponde a φ. Podemos dizer então que

$$\det(A) = \textstyle\sum_{\varphi}\mathrm{sig}(\varphi)\prod_i A_{[i,\,\varphi i]}\,,$$

onde a soma se estende a todas as bijeções φ de M em M. Portanto,

$$\mathrm{absdet}(A) \;\le\; \textstyle\sum_{\varphi}\prod_i \alpha_{i\,\varphi i}\,.$$

Ora, cada um dos produtos que comparecem do lado direito da desigualdade também está presente[6] no produto de somas $\prod_i\sum_j\alpha_{ij}$. Logo, $\mathrm{absdet}(A)$ não é maior que esse produto de somas. □

Acabamos de mostrar que $\mathrm{absdet}(A)$ é limitado pelo produto das somas de linhas. Demonstra-se analogamente que $\mathrm{absdet}(A)$ é limitado pelo produto $\prod_j\sum_i\alpha_{ij}$ das somas de colunas.

[6] Da mesma forma que $\alpha\delta$ e $\beta\gamma$ estão presentes no produto de somas $(\alpha+\beta)(\gamma+\delta)$.

$$\begin{array}{ccc} \alpha & \beta & \gamma \\ \delta & \varepsilon & \zeta \\ \eta & \theta & \iota \end{array}$$

Figura 10.3: Suponha que todos os componentes da matriz são positivos. Então seu determinante é limitado por $(\alpha+\beta+\gamma)(\delta+\varepsilon+\zeta)(\eta+\theta+\iota)$ e também por $(\alpha+\delta+\eta)(\beta+\varepsilon+\theta)(\gamma+\zeta+\iota)$.

10.6 Conclusão

O determinante de uma matriz quadrada A é um número da forma $\sigma - \sigma'$, em que σ e σ' são somas de produtos de elementos de A e cada produto tem exatamente um elemento de cada linha e de cada coluna de A. A propriedade mais importante do determinante é a "lei do produto": para qualquer par A, B de matrizes

$$\det(AB) \;=\; \det(A)\det(B)\,.$$

Mesmo sem conhecer os detalhes da definição de determinante, fica claro que o determinante de uma matriz inteira é um número inteiro. Fica claro também que o valor absoluto do determinante é limitado pelo produto das somas de linhas e pelo produto das somas de colunas:

$$\text{absdet}(A) \;\le\; \textstyle\prod_i \sum_j \alpha_{ij} \quad \text{e} \quad \text{absdet}(A) \;\le\; \prod_j \sum_i \alpha_{ij}\,,$$

em que α_{ij} é o valor absoluto de $A_{[i,\,j]}$. Estas delimitações permitem prever, ainda que grosseiramente, a ordem de grandeza do determinante da matriz.

Exercícios

10.1 Mostre que $\text{sig}(\widetilde{J}) = \text{sig}(J)$ para toda matriz de permutação J.

10.2 Mostre que $\det(\widetilde{A}) = \det(A)$ para qualquer matriz quadrada A.

10.3 Suponha que A é uma matriz elementar com coluna saliente k. Mostre que $\det(A) = A_{[k,\,k]}$.

10.4 Mostre que o determinante de uma matriz quadrada A não depende da indexação das linhas e colunas, ou seja, mostre que $\det(JA\widetilde{J}) = \det(A)$ para qualquer matriz de permutação J.

10.5 Dê exemplos de matrizes para as quais a delimitação 10.9 vale com igualdade. Dê exemplos de matrizes para as quais a delimitação de determinantes de submatrizes vale com igualdade.

10.6 Deduza da delimitação produto-de-somas 10.9 a seguinte delimitação produto-de-produtos: Para qualquer matriz A sobre $M \times M$,

$$\text{absdet}(A) \;\le\; \textstyle\prod_{i\in M} \prod_{j\in M} (1+\alpha_{ij})\,,$$

onde α_{ij} é o valor absoluto de $A_{[i,j]}$.

Capítulo 11

Algoritmo de Gauss-Jordan-Chio

Este capítulo (que só depende dos capítulos 1, 2 e 10) examina uma variante do algoritmo de Gauss-Jordan. Por falta de um nome melhor, diremos que se trata do *algoritmo de Gauss-Jordan-Chio*.[1] (Veja lema 11.1 adiante.) Uma das conseqüências do algoritmo é a conhecida regra de Cramer[2] para solução de sistemas de equações por meio de determinantes.

O algoritmo transforma qualquer matriz D numa matriz equivalente E que é escalonada a menos de um fator multiplicativo. Ademais, os componentes de E são determinantes de submatrizes de D (em particular, se D só tem componentes inteiros então todos os componentes de E serão inteiros). Graças a essa propriedade, é possível formular uma boa delimitação superior para o valor absoluto dos componentes de E. A delimitação permite mostrar que o consumo de tempo do algoritmo é polinomial.

11.1 Algoritmo

Antes de formular o algoritmo, é preciso generalizar o conceito de matriz escalonada. Para qualquer número não-nulo δ, diremos que uma matriz K é **δ-bijetora** se a matriz $\delta^{-1}K$ é de bijeção. Diremos que uma matriz E sobre $M \times N$ é **δ-escalonada** se a matriz $\delta^{-1}E$ é escalonada, ou seja, se existem partes P e Q de M e N respectivamente tais que

δ-bijetora

δ-escalonada

$$E_{[M-P,\,]} = O \quad \text{e} \quad E_{[P,Q]} \text{ é } \delta\text{-bijetora.}$$

(Veja figura 11.1.) Podemos agora descrever o algoritmo.

> **Algoritmo de Gauss-Jordan-Chio** *Recebe uma matriz D sobre $M \times N$ e devolve matrizes F e G sobre $M \times M$ e um número não-nulo δ tais que $FG = \delta I$ e GD é δ-escalonada.*

[1] Referência a F. Chio [Chi53], conforme H. Eves [Eve80, sec.3.6]. Veja também Schrijver [Sch86, p.34].

[2] Referência a Gabriel Cramer (1704–1752).

Cada iteração começa com uma parte P de M, uma parte Q de N, matrizes F e G, uma matriz E e um número δ. A primeira iteração começa com P e Q vazios, $F = G = I$, $E = D$ e $\delta = 1$. Cada iteração consiste no seguinte:

CASO *1:* $E_{[h,\,]} \neq o$ *para algum* h *em* $M - P$. h
Escolha k em N tal que $E_{[h,k]} \neq 0$. k
Para cada i em M, seja δ_i o número $E_{[i,k]}$. $\delta_i = E[i,k]$
Seja F', G', E' o resultado da pivotação de F, G, E em torno de h, k.
Comece nova iteração com $P + h$, $Q + k$, F', G', E' e δ_h
nos papéis de P, Q, F, G, E e δ.

CASO *2:* $E_{[M-P,\,]} = O$.
Devolva F, G, δ e pare. □

A operação de pivotação a que se refere o texto do algoritmo é ligeiramente diferente da dos capítulos 2 e 3: dados elementos h de M e k de N, o **resultado da pivotação de** F, G, E **em torno de** h, k é o terno F', G', E' de matrizes definido pelas equações pivotação

$$\begin{aligned}
F'_{[\,,h]} &= D_{[\,,k]}\,, & F'_{[\,,i]} &= F_{[\,,i]}\,,\\
G'_{[h,\,]} &= G_{[h,\,]}\,, & G'_{[i,\,]} &= \frac{\delta_h G_{[i,\,]} - \delta_i G_{[h,\,]}}{\delta}\,,\\
E'_{[h,\,]} &= E_{[h,\,]}\,, & E'_{[i,\,]} &= \frac{\delta_h E_{[i,\,]} - \delta_i E_{[h,\,]}}{\delta}\,,
\end{aligned}$$

para cada i em $M - h$. (Veja exemplo numérico na figura 11.2.)

11.2 Análise: preliminares

Para mostrar como e por que o algoritmo de Gauss-Jordan-Chio produz os resultados anunciados basta verificar as seguintes propriedades.

Invariantes *No início de cada iteração do algoritmo,*

(i0) $\delta \neq 0$,
(i1) $FG = \delta I$,
(i2) $E = GD$,
(i3) $E_{[P,Q]}$ *é uma matriz* δ*-bijetora e* $E_{[M-P,Q]} = O$,
(i4) $G_{[\,,M-P]} = \delta I_{[\,,M-P]}$.

É claro que estas propriedades valem no início da primeira iteração. Suponha agora que elas valem no início de uma iteração qualquer que não a última. Para mostrar que continuam válidas no início da próxima iteração basta mostrar

0	0	0	δ									
0	0	δ	0									
0	δ	0	0									
δ	0	0	0									
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

Figura 11.1: Matriz δ-escalonada.

1	0	0	0	0	1	0	0	1	0	1
0	1	0	0	0	0	2	−1	1	0	
0	0	1	0	0	5	5	0	0	5	
0	0	0	1	0	0	3	1	2	0	
0	0	0	0	1	0	0	−5	1	2	
1	0	0	0	0	1	0	0	1	0	1
0	1	0	0	0	0	2	−1	1	0	
−5	0	1	0	0	0	5	0	−5	5	
0	0	0	1	0	0	3	1	2	0	
0	0	0	0	1	0	0	−5	1	2	
2	0	0	0	0	2	0	0	2	0	2
0	1	0	0	0	0	2	−1	1	0	
−10	−5	2	0	0	0	0	5	−15	10	
0	−3	0	2	0	0	0	5	1	0	
0	0	0	0	2	0	0	−10	2	4	
5	0	0	0	0	5	0	0	5	0	5
−5	0	1	0	0	0	5	0	−5	5	
−10	−5	2	0	0	0	0	5	−15	10	
25	5	−5	5	0	0	0	0	40	−25	
−50	−25	10	0	5	0	0	0	−70	60	
15	−5	5	−5	0	40	0	0	0	25	40
−15	5	3	5	0	0	40	0	0	15	
−5	−25	1	15	0	0	0	40	0	5	
25	5	−5	5	0	0	0	0	40	−25	
−50	−130	10	70	40	0	0	0	0	130	
80	65	10	−60	−25	130	0	0	0	0	130
−30	65	6	−10	−15	0	130	0	0	0	
−10	−65	2	40	−5	0	0	130	0	0	
50	−65	−10	60	25	0	0	0	130	0	
−50	−130	10	70	40	0	0	0	0	130	

Figura 11.2: Exemplo de execução do algoritmo de Gauss-Jordan-Chio. A figura registra os valores de G, E e δ no início de cada iteração.

que no fim do caso 1

$$\begin{aligned}
\delta_h &\neq 0\,, && \text{(11.a)}\\
F'G' &= \delta_h I\,, && \text{(11.b)}\\
E' &= G'D\,, && \text{(11.c)}\\
E'_{[\,,Q]} &= \delta_h\delta^{-1}E_{[\,,Q]}\,, && \text{(11.d)}\\
E'_{[\,,k]} &= \delta_h I_{[\,,h]}\,, && \text{(11.e)}\\
G'_{[\,,M-P-h]} &= \delta_h\delta^{-1}G_{[\,,M-P-h]}\,. && \text{(11.f)}
\end{aligned}$$

A propriedade (11.a) segue imediatamente da maneira como h e k são escolhidos no caso 1.

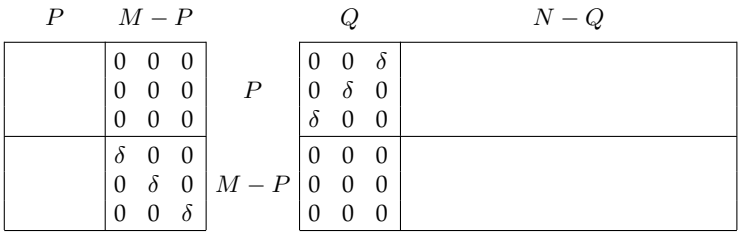

Figura 11.3: Matrizes G e E no início de uma iteração.

DEMONSTRAÇÃO DE (11.d) E (11.f): Por definição da operação de pivotação, $E'_{[h,\,]} = E_{[h,\,]}$ e

$$E'_{[i,\,]} = \delta_h\delta^{-1}E_{[i,\,]} - \delta_i\delta^{-1}E_{[h,\,]}$$

para cada i em $M-h$. Como $E_{[h,Q]}$ é nulo em virtude de (i3), temos a igualdade desejada. De modo análogo, $G'_{[h,\,]} = G_{[h,\,]}$ e

$$G'_{[i,\,]} = \delta_h\delta^{-1}G_{[i,\,]} - \delta_i\delta^{-1}G_{[h,\,]}$$

para cada i em $M-h$. Como $G_{[h,\,M-P-h]}$ é nulo em virtude de (i4), temos a igualdade desejada. □

Antes de empreender as demonstrações das demais propriedades, convém dar uma representação matricial à operação de pivotação. Seja $\check{F}$ a matriz elementar com coluna especial h definida pelas equações $\check{F}$

$$\check{F}_{[\,,h]} = \delta_h^{-1}E_{[\,,k]} \quad\text{e}\quad \check{F}_{[\,,M-h]} = \delta\delta_h^{-1}I_{[\,,M-h]}\,,$$

Seja $\check{G}$ a matriz elementar com coluna especial h definida pelas equações $\check{G}$

$$\begin{aligned}
\check{G}_{[M-h,\,h]} &= -\delta^{-1}E_{[M-h,\,k]}\,,\\
\check{G}_{[h,\,h]} &= 1\,,\\
\check{G}_{[\,,M-h]} &= \delta_h\delta^{-1}I_{[\,,M-h]}\,.
\end{aligned}$$

(Veja figuras 11.4 e 11.5.) É fácil verificar que

$$\check{F}\check{G} = \check{G}\check{F} = I\,, \quad E' = \check{G}E \quad \text{e} \quad G' = \check{G}G\,. \tag{11.g}$$

A propriedade (11.c) segue daí imediatamente. As duas propriedades restantes exigem um pouco mais de esforço.

$$\begin{array}{cccccc}
\delta/\delta_6 & 0 & 0 & 0 & 0 & \delta_1/\delta_6 \\
0 & \delta/\delta_6 & 0 & 0 & 0 & \delta_2/\delta_6 \\
0 & 0 & \delta/\delta_6 & 0 & 0 & \delta_3/\delta_6 \\
0 & 0 & 0 & \delta/\delta_6 & 0 & \delta_4/\delta_6 \\
0 & 0 & 0 & 0 & \delta/\delta_6 & \delta_5/\delta_6 \\
0 & 0 & 0 & 0 & 0 & \delta_6/\delta_6
\end{array}
\qquad
\begin{array}{cccccc}
\delta_6/\delta & 0 & 0 & 0 & 0 & -\delta_1/\delta \\
0 & \delta_6/\delta & 0 & 0 & 0 & -\delta_2/\delta \\
0 & 0 & \delta_6/\delta & 0 & 0 & -\delta_3/\delta \\
0 & 0 & 0 & \delta_6/\delta & 0 & -\delta_4/\delta \\
0 & 0 & 0 & 0 & \delta_6/\delta & -\delta_5/\delta \\
0 & 0 & 0 & 0 & 0 & 1
\end{array}$$

Figura 11.4: Exemplo com $h = 6$. A matriz à esquerda é $\check{F}$. A matriz à direita é $\check{G}$.

$$\begin{array}{cccccc}
\delta & 0 & 0 & 0 & 0 & \delta_1 \\
0 & \delta & 0 & 0 & 0 & \delta_2 \\
0 & 0 & \delta & 0 & 0 & \delta_3 \\
0 & 0 & 0 & \delta & 0 & \delta_4 \\
0 & 0 & 0 & 0 & \delta & \delta_5 \\
0 & 0 & 0 & 0 & 0 & \delta_6
\end{array}
\qquad
\begin{array}{cccccc}
\delta_6 & 0 & 0 & 0 & 0 & -\delta_1 \\
0 & \delta_6 & 0 & 0 & 0 & -\delta_2 \\
0 & 0 & \delta_6 & 0 & 0 & -\delta_3 \\
0 & 0 & 0 & \delta_6 & 0 & -\delta_4 \\
0 & 0 & 0 & 0 & \delta_6 & -\delta_5 \\
0 & 0 & 0 & 0 & 0 & \delta
\end{array}$$

Figura 11.5: Exemplo com $h = 6$. A matriz à esquerda é $\delta_6\check{F}$. A matriz à direita é $\delta\check{G}$.

DEMONSTRAÇÃO DE (11.b): Observe que a matriz $F'_{[\,,\,M-h]}$ é igual a $\delta_h\delta^{-1}F\check{F}_{[\,,\,M-h]}$ (ambas as matrizes são iguais a $F_{[\,,\,M-h]}$). Ademais,

$$\begin{aligned}
F'_{[\,,\,h]} &= D_{[\,,\,k]} \\
&= (\delta^{-1}FG)D_{[\,,\,k]} \\
&= \delta^{-1}F(E_{[\,,\,k]}) \\
&= \delta^{-1}F\left(\delta_h\check{F}_{[\,,\,h]}\right).
\end{aligned}$$

Em suma, $F' = \delta_h\delta^{-1}F\check{F}$. Logo,

$$\begin{aligned}
F'G' &= \delta_h\delta^{-1}(F\check{F})(\check{G}G) \\
&= \delta_h\delta^{-1}F(\check{F}\check{G})G \\
&= \delta_h\delta^{-1}FG \\
&= \delta_h I\,,
\end{aligned}$$

em virtude de (i1). □

DEMONSTRAÇÃO DE (11.e): Para cada i em M,

$$E'_{[i,\,k]} \;=\; \check{G}_{[i,\,]}E_{[\,,\,k]} \;=\; \check{G}_{[i,\,]}\delta_h\check{F}_{[\,,\,h]} \;=\; \delta_h(\check{G}\check{F})_{[i,\,h]} \;=\; \delta_h I_{[i,\,h]}\,.$$

Logo, $E'_{[\,,\,k]} = \delta_h I_{[\,,\,h]}$. □

Concluímos assim a demonstração de que os invariantes (i0) a (i4) valem no início de cada iteração do algoritmo. Os invariantes valem, em particular, no início da última iteração, quando $E_{[M-P,\,]}$ é nula. Nesta ocasião, em virtude de (i3),

$$E \text{ é } \delta\text{-escalonada}$$

(e tem bases P e Q). Além disso, em virtude de (i1), $FG = \delta I$. Portanto, o número δ e as matrizes F e G que o algoritmo devolve têm as propriedades anunciadas.

Tal como acontece no algoritmo de Gauss-Jordan (seção 2.3), a matriz F pode ser facilmente calculada a partir de D e E: no início de cada iteração, e portanto também ao final da execução do algoritmo,

$$F_{[\,,M-P]} = I_{[\,,M-P]} \quad \text{e} \quad F_{[\,,P]} = D_{[\,,Q]}\widetilde{J}\,,$$

onde $\widetilde{J}$ é a transposta da matriz $\delta^{-1}E_{[P,Q]}$.

11.3 Análise: invariante principal

A seção anterior deixou de lado o invariante mais característico do algoritmo: os componentes de G e E são determinantes de submatrizes de D. Para demonstrar esta propriedade, convém reformular algumas das variáveis do algoritmo.

Ajuste a notação de modo que M e N sejam disjuntos. Seja $\mathbb{D}$ a matriz que se obtém pela justaposição de D com a matriz identidade, isto é, $\mathbb{D}$ é a matriz sobre $M \times (M \cup N)$ definida pelas equações $\mathbb{D}$

$$\mathbb{D}_{[M,\,M]} = I \quad \text{e} \quad \mathbb{D}_{[M,\,N]} = D\,.$$

No início de cada iteração, seja $\mathbb{E}$ a matriz definida pela justaposição de G com E: $\mathbb{E}$

$$\mathbb{E}_{[M,\,M]} = G \quad \text{e} \quad \mathbb{E}_{[M,\,N]} = E$$

(veja figura 11.6). A relação $\mathbb{E} = G\,\mathbb{D}$, garantida pelo invariante (i2), leva a uma relação simples entre determinantes de certas submatrizes de $\mathbb{E}$ e determinantes das correspondentes submatrizes de $\mathbb{D}$. O enunciado preciso dessa relação depende da seguinte convenção de notação.

Uma função ψ de M em $M \cup N$ é uma **injeção** se ψi e ψj são distintos quando i e j são distintos.[3] Dada uma tal injeção ψ, denotaremos por $\mathbb{D}_\psi$ a matriz sobre $M \times M$ definida pelas equações $\mathbb{D}_\psi$

$$(\mathbb{D}_\psi)_{[\,,i]} \;=\; \mathbb{D}_{[\,,\psi i]}\,.$$

Portanto, $\mathbb{D}_\psi$ tem os mesmos componentes que $\mathbb{D}_{[\,,\psi M]}$ mas suas colunas são indexadas por M. Diremos que $\mathbb{D}_\psi$ é a **submatriz** de $\mathbb{D}$ **definida por** ψ. Definições análogas valem para a matriz $\mathbb{E}$. $\mathbb{E}_\psi$

[3] Aqui, ψi denota o valor de ψ em i.

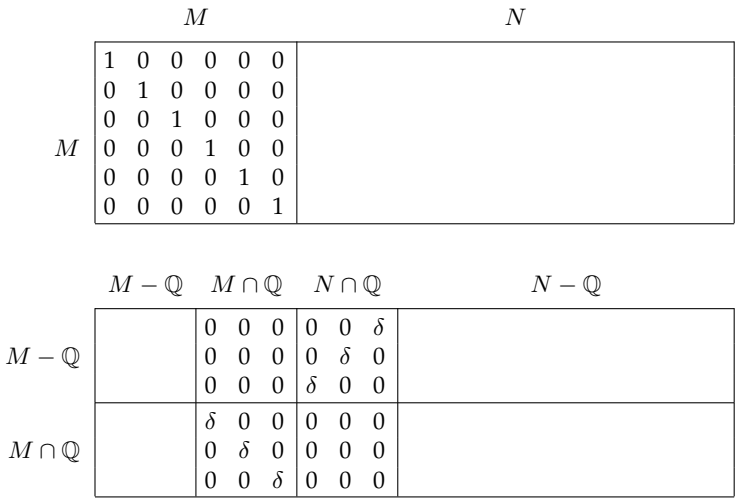

Figura 11.6: A primeira parte da figura é uma representação da matriz $\mathbb{D}$. A segunda parte representa a matriz $\mathbb{E}$.

Estamos prontos para formular a relação entre determinantes de submatrizes de $\mathbb{E}$ e $\mathbb{D}$. Essa relação corresponde ao "método de condensação de Chio" [Chi53] para cálculo de determinantes [Eve80, sec.3.6]. Poderíamos dizer que a relação é o invariante (i5) do algoritmo.

Lema 11.1 (lema de Chio) *No início de cada iteração do algoritmo, para qualquer injeção ψ de M em $M \cup N$,*

$$\det(\mathbb{E}_\psi) = \delta^{m-1} \det(\mathbb{D}_\psi)\,,$$

onde $\mathbb{E}_\psi$ e $\mathbb{D}_\psi$ são as submatrizes de $\mathbb{E}$ e $\mathbb{D}$ respectivamente definidas por ψ. Como de hábito, m é a cardinalidade de M.

DEMONSTRAÇÃO: É claro que a propriedade vale no início da primeira iteração, pois nessa ocasião $\mathbb{E} = \mathbb{D}$ e $\delta = 1$. Suponha agora que a propriedade vale no início de uma iteração qualquer que não a última; para mostrar que a propriedade vale no início da iteração seguinte, basta verificar que no fim do caso 1

$$\det(\mathbb{E}'_\psi) = \delta_h^{m-1} \det(\mathbb{D}_\psi)\,,$$

onde $\mathbb{E}'$ é a matriz definida pela justaposição de G' e E' e $\mathbb{E}'_\psi$ é a submatriz de $\mathbb{E}'$ definida por ψ. Em virtude de (11.g) temos $\mathbb{E}' = \check{G}\mathbb{E}$, donde $\mathbb{E}'_\psi = \check{G}\mathbb{E}_\psi$. O teorema 26 (do produto de determinantes) garante que

$$\det(\mathbb{E}'_\psi) \;=\; \det(\check{G}) \det(\mathbb{E}_\psi)\,.$$

Como $\check{G}$ é uma matriz diagonal, $\det(\check{G}) = (\delta_h/\delta)^{m-1}$. Isso prova a igualdade desejada. □

Podemos mostrar agora que cada componente de $\mathbb{E}$ é igual ao determinante de uma submatriz de $\mathbb{D}$. Seja $\mathbb{Q}$ o conjunto $Q \cup (M - P)$, donde $P = M - \mathbb{Q}$ e $M - P = M \cap \mathbb{Q}$. Os invariantes (i3) e (i4) garantem que $\mathbb{E}_{[M,\,\mathbb{Q}]}$ é uma matriz δ-bijetora (veja figura 11.6). Para cada i em M, seja

$\mathbb{Q}$

φ

$$\varphi i$$

o único elemento de $\mathbb{Q}$ para o qual $\mathbb{E}_{[i,\,\varphi i]} = \delta$. Assim, φ é uma injeção de M em $M \cup N$ e $\mathbb{E}_{\varphi} = \delta I$. A propriedade 11.2 abaixo — que poderíamos chamar de invariante (i6) do algoritmo — mostra que os componentes não-nulos de $\mathbb{E}_{[\,,\,\mathbb{Q}]}$ são determinantes de $\mathbb{D}_{\varphi}$; a propriedade 11.3 — que poderíamos denominar invariante (i7) — trata dos demais componentes de $\mathbb{E}$.

Propriedade 11.2 *No início de cada iteração do algoritmo,* $\delta = \det(\mathbb{D}_{\varphi})$.

DEMONSTRAÇÃO: De acordo com o lema 11.1, $\det(\mathbb{E}_{\varphi}) = \delta^{m-1} \det(\mathbb{D}_{\varphi})$. Mas $\det(\mathbb{E}_{\varphi}) = \delta^{m}$, uma vez que $\mathbb{E}_{\varphi} = \delta I$. Como $\delta \neq 0$ em virtude de (i0), temos $\delta = \det(\mathbb{D}_{\varphi})$. □

Propriedade 11.3 *No início de cada iteração do algoritmo, para cada* i *em* M *e cada* j *em* $(M \cup N) - \mathbb{Q}$,

$$\mathbb{E}_{[i,\,j]} \;=\; \det(\mathbb{D}_{\psi})\,,$$

onde ψ *é a injeção de* M *em* $M \cup N$ *definida pelas equações* $\psi i = j$ *e* $\psi h = \varphi h$ *para cada* h *em* $M - i$.

DEMONSTRAÇÃO: De acordo com o lema 11.1, $\det(\mathbb{E}_{\psi}) = \delta^{m-1} \det(\mathbb{D}_{\psi})$. Como $\delta \neq 0$, resta apenas provar que

$$\det(\mathbb{E}_{\psi}) = \delta^{m-1}\, \mathbb{E}_{[i,\,j]}\,.$$

Para isso, é preciso estudar a estrutura de $\mathbb{E}_{\psi}$. A definição de ψ garante que $\mathbb{E}_{\psi\,[\,,\,M-i]} = \mathbb{E}_{\varphi\,[\,,\,M-i]}$. Como $\mathbb{E}_{\varphi} = \delta I$, temos

$$\mathbb{E}_{\psi\,[\,,\,M-i]} = \delta I_{[\,,\,M-i]}\,,$$

donde $\mathbb{E}_{\psi\,[i,\,M-i]}$ é nulo. Portanto, o desenvolvimento do determinante de $\mathbb{E}_{\psi}$ pela linha i (propriedade 10.3) garante que

$$\begin{aligned} \det(\mathbb{E}_{\psi}) \;&=\; \mathbb{E}_{\psi[i,\,i]}\, \det(\mathbb{E}_{\psi[M-i,\,M-i]}) \\ &=\; \mathbb{E}_{\psi[i,\,i]}\, \delta^{m-1}\,. \end{aligned}$$

Para concluir a demonstração, resta observar que $\mathbb{E}_{\psi\,[\,,\,i]} = \mathbb{E}_{[\,,\,j]}$. □

Esta seção demonstrou que, no início de cada iteração do algoritmo de Gauss-Jordan-Chio, cada componente de $\mathbb{E}$ é o determinante de uma submatriz de $\mathbb{D}$. Se o conceito de submatriz for generalizado da maneira óbvia, a propriedade pode ser reformulada (com auxílio da propriedade 10.3 do desenvolvimento do determinante por uma linha) diretamente em termos de D, G e E: *No início de cada iteração do algoritmo, cada componente de* G *e de* E *é o determinante de alguma submatriz de* D .

11.4 Delimitação dos números gerados

As propriedades que discutimos acima permitem delimitar o valor absoluto dos componentes da matriz $\mathbb{E}$ ao longo da execução do algoritmo. Como $\mathbb{E}$ é mera justaposição de G e E, a delimitação se aplica aos componentes das matrizes G e E, e portanto também ao número δ.

Delimitação 11.4 *No início de cada iteração do algoritmo, para cada i em M e cada j em $M \cup N$,*

$$|\mathbb{E}[i,j]| \;\leq\; \textstyle\prod_{p\in M}(1+\sum_{q\in N}\alpha_{pq})\,,$$

onde α_{pq} é o valor absoluto de $D[p,q]$.

DEMONSTRAÇÃO: De acordo com as propriedades 11.2 e 11.3, $\mathbb{E}[i,j]$ é da forma $\det(\mathbb{D}_\psi)$, onde ψ é uma injeção de M em $M \cup N$. De acordo com a delimitação 10.9 do determinante,

$$\operatorname{absdet}(\mathbb{D}_\psi) \;\leq\; \textstyle\prod_{p\in M}(\sum_{q\in\psi M}|\mathbb{D}[p,q]|)\,.$$

A desigualdade não é violada se "$q \in \psi M$" for trocado por "$q \in M \cup N$" (sempre supondo M disjunto de N). Finalmente, como $\mathbb{D}$ é a justaposição de I e D,

$$\textstyle\sum_{q\in M\cup N}|\mathbb{D}[p,q]| \;=\; 1+\sum_{q\in N}\alpha_{pq}\,,$$

$$\begin{array}{rrrrrrr}
1 & 0 & 0 & 11 & 21 & 31 & 41 \\
0 & 1 & 0 & 12 & 22 & 32 & 42 \\
0 & 0 & 1 & 13 & 23 & 33 & 43
\end{array}$$

$$\begin{array}{rrrrrrr}
12 & -11 & 0 & 0 & 10 & 20 & 30 \\
-22 & 21 & 0 & 10 & 0 & -10 & -20 \\
10 & -20 & 10 & 0 & 0 & 0 & 0
\end{array}$$

$$\begin{array}{rrr}
21 & 11 & 0 \\
22 & 12 & 0 \\
23 & 13 & 1
\end{array}$$

$$\begin{array}{rrr}
21 & 0 & 0 \\
22 & 1 & 0 \\
23 & 0 & 1
\end{array}$$

$$\begin{array}{rrr}
21 & 11 & 41 \\
22 & 12 & 42 \\
23 & 13 & 43
\end{array}$$

Figura 11.7: Ilustração das propriedades 11.2 e 11.3. A primeira matriz da figura é $\mathbb{D}$ e a segunda é $\mathbb{E}$. O valor de δ é 10. A injeção φ é definida por $\varphi 1 = 5$, $\varphi 2 = 4$, $\varphi 3 = 3$. A terceira matriz da figura é $\mathbb{D}_\varphi$. O seu determinante vale 10. A quarta matriz é $\mathbb{D}_\psi$ com $i = 2$ e $j = 2$. O seu determinante, 21, é o valor de $\mathbb{E}[2,2]$. A quinta matriz é $\mathbb{D}_\psi$ com $i = 3$ e $j = 7$. O seu determinante, 0, é o valor de $\mathbb{E}[3,7]$. Todas as matrizes estão indexadas por $1, 2, 3, \ldots$ de cima para baixo e da esquerda para a direita.

como queríamos demonstrar. □

Uma demonstração semelhante prova que $\mathbb{E}_{[i,\,j]}$ é limitado pelo produto das somas de colunas $\prod_q \sum_p \alpha_{pq}$ (desta vez sem o "$1+$"). Mostraremos a seguir que $\mathbb{E}_{[i,\,j]}$ também é limitado por um produto de produtos. A delimitação é mais grosseira que as anteriores, mas muito útil, por ser mais fácil de manipular.

Delimitação 11.5 *No início de cada iteração do algoritmo, para cada i em M e cada j em $M \cup N$,*

$$|\mathbb{E}_{[i,\,j]}| \;\leq\; \textstyle\prod_{p\in M} \prod_{q\in N}(1+\alpha_{pq})\,,$$

em que α_{pq} é o valor absoluto de $D_{[p,\,q]}$.

DEMONSTRAÇÃO: Em virtude da delimitação 11.4, basta mostrar que

$$1 + \textstyle\sum_{q\in N} \alpha_{pq} \;\leq\; \prod_{q\in N}(1+\alpha_{pq})$$

para cada p em M. Ora, a soma à esquerda é parte do desenvolvimento do produto à direita (da mesma forma que $1+\alpha+\beta+\gamma$ é parte do desenvolvimento de $(1+\alpha)(1+\beta)(1+\gamma)$). Como as demais parcelas do desenvolvimento são não-negativas, temos a desigualdade desejada. □

11.5 Aplicação a matrizes inteiras

Tudo que dissemos até aqui não depende da natureza dos componentes de D. Suponha agora que todos os componentes da matriz dada D são inteiros. Submeta D ao algoritmo de Gauss-Jordan-Chio. No início de cada iteração, todos os componentes de $\mathbb{E}$ serão determinantes de submatrizes da matriz $\mathbb{D}$; portanto, todos os componentes de G e E serão inteiros (veja figura 11.8). Ademais, de acordo com a delimitação 11.5, todos os componentes de G e E estarão entre

$$-\omega \quad \text{e} \quad +\omega\,,$$

onde $\omega = \prod_p \prod_q (1+\alpha_{pq})$ e α_{pq} é o valor absoluto de $D_{[p,\,q]}$. A delimitação permanece válida se ω for definido pelo produto de somas $\prod_p (1+\sum_q \alpha_{pq})$ ou pelo produto de somas $\prod_q \sum_p \alpha_{pq}$.

É oportuno perguntar se os componentes da matriz E que o algoritmo gera não são muito maiores que o estritamente necessário. A resposta é negativa, ainda que num sentido fraco: os componentes de E admitem, não raro, um divisor comum não-trivial; mas há matrizes D para as quais pelo menos um dos componentes de E — digamos δ — resulta primo.

Quando a matriz dada D é inteira, o algoritmo de Gauss-Jordan-Chio pode ser executado usando somente aritmética inteira. Para isso, basta calcular as expressões que definem a operação de pivotação exatamente na ordem indicada: primeiro a expressão no numerador e depois a divisão por δ, que será exata. (Veja a figura 11.9.)

$$\frac{\delta_3\, E[4,4] - \delta_4\, E[3,4]}{\delta} \;=\; \frac{5\cdot 1 + 5\cdot 15}{2}$$

Figura 11.8: Examine o exemplo da figura 11.2. Apesar das divisões por δ, todos os números permanecem inteiros ao longo das sucessivas iterações. Por exemplo, durante a terceira iteração ocorre uma pivotação em torno de $3\,,3$ e o valor de $E'[4,4]$ é dado pela expressão da figura. A expressão tem valor inteiro, ainda que as sub-expressões $5\cdot 1/2$ e $5\cdot 15/2$ tenham valor fracionário.

$$\begin{array}{rrrrrrrrrr}
3 & 2 & 1 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
4 & 3 & 2 & 5 & 6 & 7 & 8 & 9 & 73 & 1 \\
5 & 6 & 3 & 4 & 7 & 9 & 8 & 10 & 1 & 2 \\
1 & 5 & 4 & 7 & 5 & 10 & 9 & 6 & 2 & 3 \\
32 & 6 & 5 & 8 & 9 & 10 & 9 & 2 & 7 & 4 \\
8 & 7 & 6 & 9 & 10 & 5 & 2 & 3 & 4 & 5 \\
9 & 8 & 7 & 11 & 3 & 2 & 1 & 4 & 5 & 8
\end{array}$$

$$\begin{array}{rrrrrrrr}
-129574 & 118845 & -13654 & 6748 & 1192 & 955 & -7880 \\
35638 & -35661 & 20158 & -10816 & -2992 & -5203 & 9236 \\
2472 & -3186 & 36 & -864 & 1860 & -678 & 288 \\
53560 & -45906 & -5780 & 3584 & -208 & 1874 & 4436 \\
-9064 & 15372 & -460 & -5852 & -1244 & 5848 & -3680 \\
41406 & -60315 & -750 & 18000 & 3480 & 5679 & -6000 \\
-45114 & 61527 & 450 & -10800 & -2088 & -8475 & 3600
\end{array}$$

$$\begin{array}{rrrrrrrrrr}
0 & 0 & 50676 & 0 & 0 & 0 & 0 & -89310 & 7482125 & -1237456 \\
0 & 50676 & 0 & 0 & 0 & 0 & 0 & 116190 & -2279561 & 364492 \\
50676 & 0 & 0 & 0 & 0 & 0 & 0 & -10884 & -200274 & 25368 \\
0 & 0 & 0 & 50676 & 0 & 0 & 0 & 1980 & -2839490 & 532912 \\
0 & 0 & 0 & 0 & 50676 & 0 & 0 & 26460 & 1024700 & -98920 \\
0 & 0 & 0 & 0 & 0 & 50676 & 0 & -111090 & -3978015 & 400560 \\
0 & 0 & 0 & 0 & 0 & 0 & 50676 & 117330 & 4033779 & -443040
\end{array}$$

Figura 11.9: Resultado da aplicação do algoritmo de Gauss-Jordan-Chio à matriz que aparece no topo da figura. O algoritmo devolve o número 50676 e as matrizes G (meio da figura) e E (parte inferior da figura). O valor da expressão $\prod_p (1 + \sum_q \alpha_{pq})$ é, aproximadamente, $4\,411\,057\,000\,000$.

Eficiência. O número de dígitos na representação de ω vale, aproximadamente, $\log \omega$. Se adotarmos o produto-de-produtos como definição de ω, teremos

$$\log \omega = \textstyle\sum_p \sum_q \log(1 + \alpha_{pq})$$

e, portanto, o número de dígitos de ω será igual, aproximadamente, ao número total de dígitos de D. Pode-se dizer que ω é uma medida do tamanho da matriz dada D.

Cada operação de pivotação envolve menos que $3(m-1)n$ multiplicações e divisões e menos que $(m-1)n$ subtrações, onde m é o número de linhas e n o número de colunas de D. Como o número de iterações não passa de m, o número total de operações aritméticas é menor que

$$4m^2n\,.$$

Suponhamos que o tempo necessário para executar uma operação aritmética entre dois números inteiros que estejam entre $-\omega$ e ω é limitado por $\log^2 \omega$. Então o consumo total de tempo do algoritmo será da ordem de

$$m^2 n \log^2 \omega$$

no pior caso. Em suma, o tempo que o algoritmo consome é limitado por uma função polinomial do espaço necessário para descrever a matriz D. Esta delimitação polinomial foi originalmente demonstrada por Jack Edmonds [Edm67].

Matrizes totalmente unimodulares. Uma matriz é **totalmente unimodular** se o determinante de cada uma de suas submatrizes quadradas vale -1, 0 ou $+1$ (em particular, o valor de cada componente é -1, 0 ou $+1$). Matrizes totalmente unimodulares aparecem, naturalmente, em vários problemas de otimização combinatória [CCPS98, Sch86, AMO93].

Se o algoritmo de Gauss-Jordan-Chio for aplicado a uma matriz totalmente unimodular D então, de acordo com a propriedade 11.2, o número δ será igual a -1 ou $+1$ em todas as iterações. Portanto, o algoritmo devolverá uma matriz inversível inteira G tal que GD ou $-GD$ é 1-escalonada.

11.6 Conclusão

Ao receber uma matriz inteira D, o algoritmo de Gauss-Jordan-Chio devolve uma matriz inversível inteira G e um número inteiro não-nulo δ tais que

$$\text{a matriz } GD \text{ é } \delta\text{-escalonada.}$$

(É como se o algoritmo produzisse uma matriz escalonada cujos componentes são números racionais, todos com o mesmo denominador δ.) O valor absoluto de cada componente de G e GD (e, em particular, o número δ) será limitado

pelo produto ω de todos os números da forma $1 + \alpha$ onde α é o valor absoluto de um componente de D.

A quantidade de tempo que o algoritmo consome no pior caso é da ordem de $m^2 n \log^2 \omega$, onde m é o número de linhas e n o número de colunas de D.

Exercícios

11.1 Use o algoritmo de Gauss-Jordan-Chio para calcular o determinante da matriz abaixo.

$$\begin{array}{rrrr} 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \\ 1 & 4 & 16 & 64 \\ 1 & 5 & 25 & 125 \end{array}$$

11.2 Suponha que ω é o produto das somas de colunas de D. Mostre que $\log \omega \leq n \log(n\alpha)$, onde $\alpha = \max_q \max_p \alpha_{pq}$ e n é o número de colunas de D.

11.3 Implemente o algoritmo de Gauss-Jordan-Chio em um computador. Suponha que a matriz dada, D, é inteira. Seu programa deve detectar *overflow* de operações aritméticas ou fazer aritmética inteira com precisão ilimitada.

Capítulo 12

Algoritmo Simplex-Chio

A operação de pivotação característica do algoritmo de Gauss-Jordan-Chio (veja capítulo 11) pode também ser usada no algoritmo Simplex. O resultado dessa combinação será chamado *algoritmo Simplex-Chio*, por falta de um nome melhor.

O algoritmo transforma qualquer matriz D numa matriz equivalente E que é simples a menos de um fator multiplicativo. Cada componente de E é igual ao determinante de alguma submatriz de D; portanto, se cada componente de D é inteiro então os componentes de E também serão inteiros. É possível dar uma boa delimitação superior para o valor absoluto dos componentes de E e assim mostrar que o consumo de tempo de cada iteração é polinomialmente limitado (ainda que o número de iterações possa ser exponencial).

12.1 Algoritmo

Para qualquer número não-nulo δ, diremos que uma matriz E é δ-**simples** com relação a um par n, m de índices se a matriz

$$\delta^{-1} E$$

é simples (solúvel, inviável ou ilimitada) com relação a n, m. O algoritmo Simplex-Chio recebe uma matriz arbitrária D e um par n, m de índices e devolve um número não-nulo δ e uma matriz E que é δ-simples com relação a n, m.

Para simplificar a exposição, vamos descrever apenas a heurística Simplex-Chio; essa heurística pode ser convertida num algoritmo pela introdução de um mecanismo de convergência como o que usamos no Simplex Lexicográfico (seção 5.3).

Heurística Simplex-Chio *Recebe uma matriz D sobre $M \times N$ e elementos n e m de N e M respectivamente; se convergir, devolve matrizes F e G e um número não-nulo δ tais que*

$$FG = \delta I\,, \quad G_{[\,,m]} = \delta I_{[\,,m]} \quad \textit{e} \quad GD \textit{ é } \delta\textit{-simples}$$

											n			
											0	0	δ	$\geq$
											0	δ	0	$\geq$
											δ	0	0	$\geq$
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
m	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	$\geq$	0	0	0	

Figura 12.1: Matriz δ-simples solúvel.

(solúvel, inviável ou ilimitada) com relação a n, m*.*

Cada iteração começa com matrizes F, G e E, um número δ, uma parte L de $M - m$ e um elemento h de $M - L$. A primeira iteração começa com $F = G = I$, $E = D$, $\delta = 1$, $L = \emptyset$ e com h em M, se possível distinto de m. Cada iteração consiste no seguinte, com $f = E_{[\,,n]}$:

h

ALTERNATIVA *I:* h *é diferente de* m *.*

CASO *I.1:* $E_{[h,k]} > 0$ *e* $f_{[h]} \geq 0$ *ou* $E_{[h,k]} < 0$ *e* $f_{[h]} \leq 0$ *para algum* k *em* $N - n$ *.*

k

Para cada p em $M - m$, seja δ_p o número $E_{[p,k]}$.

$\delta_p = E[p,k]$

Seja L^* o conjunto dos índices p em L para os quais $\delta_p/\delta > 0$.

CASO *I.1*A*:* $f_{[h]}/\delta_h \leq f_{[p]}/\delta_p$ *para todo* p *em* L^* *.*
Seja F', G', E' o resultado da pivotação de F, G, E em torno de h, k.
Seja L' o conjunto $L + h$.
Escolha h' em $M - L'$, se possível distinto de m.
Comece nova iteração com F', G', E', δ_h, L', h'
nos papéis de F, G, E, δ, L, h.

CASO *I.1*B*:* $f_{[h]}/\delta_h > f_{[p]}/\delta_p$ *para algum* p *em* L^* *.*
Escolha qualquer p em L^* tal que $f_{[p]}/\delta_p$ é mínimo.
Seja F', G', E' o resultado da pivotação de F, G, E em torno de p, k.
Comece nova iteração com F', G', E' e δ_p
nos papéis de F, G, E e δ.

CASO *I.2:* $E_{[h,\,N-n]} \leq o$ *e* $f_{[h]} > 0$ *ou* $E_{[h,\,N-n]} \geq o$ *e* $f_{[h]} < 0$ *.*
Devolva F, G, δ e pare.

CASO *I.3:* $E_{[h,\,N-n]} = o$ *e* $f_{[h]} = 0$ *.*
Seja L' o conjunto $L + h$.
Escolha h' em $M - L'$, se possível distinto de m.
Comece nova iteração com L' e h' nos papéis de L e h.

ALTERNATIVA *II:* h *é igual a* m *.*

CASO *II.1:* $E[h,k]/\delta < 0$ *para algum* k *em* $N - n$. k
Para cada p em $M - m$, seja δ_p o número $E[p,k]$. $\delta_p = E[p,k]$
Seja L^* o conjunto dos índices p em L para os quais $\delta_p/\delta > 0$.

CASO *II.1*A*:* L^* *é vazio* .
Devolva F, G, δ e pare.

CASO *II.1*B*:* L^* *não é vazio* .
Escolha qualquer p em L^* tal que $f_{[p]}/\delta_p$ é mínimo.
Seja F', G', E' o resultado da pivotação de F, G, E em torno de p, k.
Comece nova iteração com F', G', E' e δ_p
nos papéis de F, G, E e δ.

CASO *II.2:* $E_{[h,\,N-n]}/\delta \geq o$.
Devolva F, G, δ e pare. □

A operação de pivotação é definida exatamente como no algoritmo de Gauss-Jordan-Chio.

12.2 Análise

A análise da heurística é análoga à do algoritmo de Gauss-Jordan-Chio.

Invariantes *No início de cada iteração da heurística,*

(i0) $\delta \neq 0$,
(i1) $FG = \delta I$,
(i2) $E = GD$,

e existem partes P *de* L *e* Q *de* $N - n$ *tais que*

(i3) $E_{[P,Q]}$ *é* δ*-bijetora* , $E_{[M-P,Q]} = O$ *e* $E_{[L-P,\,]} = O$,
(i4) $G_{[\,,M-P]} = \delta I_{[\,,M-P]}$,
(i5) $f_{[P]} \geq o$,

onde f *é o vetor* $E_{[\,,n]}$. *Além disso, cada componente de* G *e de* E *é o determinante de uma submatriz de* D .

A demonstração dos invariantes é inteiramente análoga à que discutimos para o algoritmo de Gauss-Jordan-Chio. Os invariantes valem, em particular, no fim da última iteração. Portanto, as objetos F, G e δ que a heurística devolve nos casos I.2, II.1A e II.2 têm as propriedades anunciadas.

12.3 Aplicação a matrizes inteiras

Suponha agora que todos os componentes da matriz dada D são inteiros. Então, no início de cada iteração, em virtude de (i6), o número δ e todos os componen-

tes de G e de E também serão inteiros. Ademais, pelas mesmas razões que na delimitação 11.5, todos estarão entre

$$-\omega \quad \text{e} \quad +\omega\,,$$

onde $\omega = \prod_p \prod_q (1 + \alpha_{pq})$ e α_{pq} é o valor absoluto de $D[p,q]$. A delimitação permanece válida se ω for definido pelo produto de somas $\prod_p (1 + \sum_q \alpha_{pq})$ ou pelo produto de somas $\prod_q \sum_p \alpha_{pq}$.

Quando D é inteira, o Simplex-Chio pode ser executado usando somente aritmética inteira. De fato, a divisão por δ implícita na operação de pivotação terá sempre resultado inteiro. Além disso, a operação de divisão que aparece na definição de L^* não precisa ser executada explicitamente pois δ_p/δ é positivo se e só se δ_p e δ têm o mesmo sinal. Analogamente, as operações de divisão que aparecem na definição dos casos I.1A e I.1B e na escolha de p podem ser feitas implicitamente.

$$\begin{array}{rrrrrrrrr}
3 & 2 & 1 & 4 & 5 & 6 & 7 & 8 & 10 \\
4 & 3 & 2 & 5 & 6 & 7 & -8 & -9 & 11 \\
5 & 6 & 3 & 4 & 7 & -9 & 8 & -10 & 9 \\
1 & -5 & 4 & 7 & 5 & -10 & 9 & 6 & 8 \\
9 & 8 & 7 & 11 & 3 & 2 & 1 & 4 & 0
\end{array}$$

$$\begin{array}{rrrrr}
-157 & -334 & 277 & 170 & 0 \\
-466 & 389 & 13 & -19 & 0 \\
-1635 & 996 & -1542 & 1104 & 0 \\
1343 & -1522 & 1486 & -1597 & 0 \\
-7690 & 12866 & -12326 & 13991 & -7473
\end{array}$$

$$\begin{array}{rrrrrrrrr}
-252 & -504 & 686 & 0 & 0 & -7473 & 5319 & 0 & -1391 \\
204 & 408 & 275 & 0 & 0 & 0 & -6441 & -7473 & -416 \\
-7527 & -15054 & 147 & 0 & -7473 & 0 & -21813 & 0 & -10440 \\
3774 & 15021 & -3631 & -7473 & 0 & 0 & 19092 & 0 & -2714 \\
-86502 & -180477 & -15283 & 0 & 0 & 0 & -136920 & 0 & 65620
\end{array}$$

Figura 12.2: Submeta à heurística Simplex-Chio a matriz D descrita no topo da figura. Depois de 10 iterações, a heurística devolverá o número -7473, a matriz G (meio da figura) e a matriz E (parte inferior da figura). A matriz $-E/7473$ é simples solúvel com relação à coluna 9 e linha 5. O valor da expressão $\prod_p(1 + \sum_q \alpha_{pq})$ é 381 966 750.

Eficiência de uma iteração. Qualquer que seja o mecanismo de convergência adotado, o número de iterações do algoritmo Simplex-Chio será da ordem de 2^n no pior caso, onde n é o número de colunas de D.

Se adotarmos o mecanismo lexicográfico, o número de operações aritméticas em uma iteração será inferior a $m(4n+3m)$, onde m é o número de linhas de D, pelas razões que passamos a detalhar. Cada operação de pivotação envolve menos que $3(m-1)n$ multiplicações e divisões e menos que $(m-1)n$ subtrações.

A escolha da linha e coluna em torno da qual se fará a próxima pivotação envolve menos que $2(m-1)n$ multiplicações e menos que $(m-1)n$ subtrações. (Na verdade, é mais justo trocar "n" por "m", uma vez que o resultado das comparações lexicográficas só depende dos $|Q|+1$ primeiras colunas de E.)

Em cada operação aritmética, o número de dígitos de cada operando é limitado por $\log \omega$ e portanto o esforço necessário para executar a operação é limitado por $\log^2 \omega$. Logo, o consumo de tempo de uma iteração é da ordem de

$$mn \log^2 \omega \,.$$

12.4 Conclusão

Ao receber uma matriz inteira D e índices n e m de coluna e linha respectivamente, o algoritmo Simplex-Chio devolve uma matriz inversível inteira G e um número inteiro não-nulo δ tais que a matriz GD é δ-simples com relação a n, m.

O valor absoluto de cada componente de G e GD (e, em particular, o número δ) será limitado pelo produto de todos os números da forma $1+\alpha$, onde α é o valor absoluto de um componente de D.

Exercícios

12.1 Implemente o algoritmo Simplex-Chio (com regra lexicográfica ou regra de Bland) em um computador. Suponha que a matriz dada é inteira. Seu programa deve detectar *overflow* de operações aritméticas ou fazer aritmética inteira com precisão ilimitada.

Capítulo 13

Problemas com dados inteiros

Suponha dado um sistema de equações lineares com coeficientes inteiros ou um par dual de problemas canônicos de programação linear com coeficientes inteiros. Este capítulo resume o resultado da aplicação dos algoritmos Gauss-Jordan-Chio e Simplex-Chio a esses problemas. Mostra também como é possível prever a ordem de grandeza das soluções dos problemas.

13.1 Sistemas de equações

Suponha que A é uma matriz inteira e b é um vetor inteiro. Nosso problema é encontrar um vetor racional x^* tal que

$$Ax^* \;=\; b\,.$$

Submeta A ao algoritmo de Gauss-Jordan-Chio (seção 11.1) O algoritmo produzirá um inteiro não-nulo δ e uma matriz inversível G tal que GA é δ-escalonada. É fácil extrair de GA e Gb, como já fizemos na seção 2.8, uma solução x^* de nosso problema ou a evidência de que tal solução não existe. Segue daí o seguinte

Teorema 13.1 *Para qualquer matriz inteira A sobre $M \times N$ e qualquer vetor inteiro b sobre M, vale uma e apenas uma das alternativas:*

(1) existe um inteiro δ entre 1 e ω e um vetor inteiro x com componentes entre $-\omega$ e ω tais que $Ax = \delta b$,

(2) existe um vetor inteiro g com componentes entre $-\omega$ e ω tal que $gA = o$, $gb \neq 0$ e gb está entre $-\omega$ e ω,

onde ω é o tamanho do sistema A, b.

O **tamanho** do sistema A, b é o número $\prod_i(1 + \beta_i) \cdot \prod_i \prod_j (1 + \alpha_{ij})$, em que β_i e α_{ij} são os valores absolutos de $b_{[i]}$ e $A_{[i,j]}$ respectivamente. As conclusões continuam válidas se o tamanho do sistema for definido pelas expressões $\prod_i(1+ \beta_i + \sum_j \alpha_{ij})$ ou $(\prod_j \sum_i \alpha_{ij}) \cdot (\sum_i \beta_i)$.

A alternativa *(1)* é, essencialmente, o resultado da solução das equações $Ax^* = b$ pela "regra de Cramer". [1] A alternativa *(2)* é um certificado de que as equações não têm solução.

Matrizes totalmente unimodulares. Se A é totalmente unimodular (veja seção 11.5) então $\delta = 1$ no teorema acima e portanto existe um vetor inteiro x tal que $Ax = b$.

13.2 Problemas canônicos

Suponha que A, b, c é um sistema inteiro. Considere os problemas canônicos CP(A, b, c) e CD(A, c, b). O primeiro consiste em encontrar um vetor racional x^* em $X(A, b)$ que minimize a expressão cx^*; o segundo procura um vetor racional y^* em $Y(A, c)$ que maximize y^*b.

Submeta ao algoritmo Simplex-Chio (seção 12.1) o sistema D, m, n definido, como em (6.b), pela justaposição de A, b e c. O algoritmo produzirá uma matriz δ-simples E a partir da qual é fácil deduzir, como já fizemos nos capítulos 7 e 8, soluções x^* e y^* dos dois problemas canônicos ou a evidência de que tais soluções não existem.

Segue daí a versão do teorema da dualidade 8.5 enunciada abaixo. Nosso enunciado envolve os conjuntos

$$X(A, \delta b) \quad \text{e} \quad Y(A, \delta c)\,,$$

com δ positivo. É óbvio um vetor x está no primeiro conjunto se e só se x/δ está em $X(A, b)$; analogamente, um vetor y está no segundo se e só se y/δ está em $Y(A, c)$.

Teorema 13.2 (dualidade para dados inteiros) *Para qualquer matriz inteira A sobre $M \times N$, vetor inteiro b sobre M e vetor inteiro c sobre N, existe um inteiro positivo δ para o qual vale uma e apenas uma das seguintes afirmações:*

(1) existe um vetor inteiro x em $X(A, \delta b)$ e um vetor inteiro y em $Y(A, \delta c)$ tais que $cx = yb$,
(2) existe um vetor inteiro x em $X(A, \delta b)$ e um vetor inteiro x' em $X(A, o)$ tal que $cx' \leq -1$,
(3) existe um vetor inteiro y em $Y(A, \delta c)$ e um vetor inteiro y em $Y(A, o)$ tal que $y'b \geq +1$,
(4) existe um vetor inteiro x' em $X(A, o)$ e um vetor inteiro y' em $Y(A, o)$ tais que $cx' \leq -1$ e $y'b \geq +1$.

Ademais, cada componente de x, x', y e y', bem como os números δ, cx, yb, cx' e $y'b$, estão entre $-\omega$ e ω, onde ω é o tamanho do sistema A, b, c.

[1] Referência a Gabriel Cramer (1704–1752).

O **tamanho** do sistema A, b, c é o número $\prod_i \prod_j (1+\alpha_{ij}) \cdot \prod_i (1+\beta_i) \cdot \prod_j (1+\gamma_j)$, em que α_{ij}, β_i e γ_j são os valores absolutos de $A_{[i,j]}$, $b_{[i]}$ e $c_{[j]}$ respectivamente (é claro que i percorre o conjunto de índices de linhas de A e j percorre o conjunto de índices de colunas de A). O teorema continua válido se o tamanho do sistema A, b, c for definido pelas expressões $\prod_i (1 + \beta_i + \sum_j \alpha_{ij}) \cdot (1 + \sum_j \gamma_j)$ ou $\prod_j (\gamma_j + \sum_i \alpha_{ij}) \cdot (\sum_i \beta_i)$.

Matrizes totalmente unimodulares. Se A é totalmente unimodular então o teorema da dualidade 13.2 vale com $\delta = 1$. Nesse caso, se os problemas $\mathsf{CP}(A, b, c)$ e $\mathsf{CD}(A, c, b)$ têm soluções então existe um vetor inteiro x em $X(A, b)$ e um vetor inteiro y em $Y(A, c)$ tais que $cx = yb$.

Essas observações trazem à baila a seguinte questão, de fundamental importância para problemas de otimização combinatória: em que condições os problemas $\mathsf{CP}(A, b, c)$ e $\mathsf{CD}(A, c, b)$ têm soluções inteiras? Edmonds e Giles [EG84] investigaram essa questão e introduziram o conceito de sistema TDI (totally dual integral) para caracterizar algumas respostas. O leitor interessado poderá encontrar mais informações no livro de Schrijver [Sch86, Sch03].

13.3 Conclusão

O algoritmo Simplex-Chio pode ser usado para resolver o par dual de problemas canônicos $\mathsf{CP}(A, b, c)$ e $\mathsf{CD}(A, c, b)$ com dados A, b e c inteiros. A eventual solução do problema primal e a correspondente solução do problema dual terão a forma

$$x/\delta \quad \text{e} \quad y/\delta$$

respectivamente, onde x e y são vetores inteiros enquanto δ é um número inteiro positivo. Tanto δ quanto os componentes de x e y são limitados, em valor absoluto, pelo produto de todos os monômios da forma $(1+\alpha)$, onde α é o valor absoluto de um componente do sistema A, b, c.

As conclusões se aplicam também, num certo sentido, a dados racionais: se os dados são racionais, basta multiplicá-los todos por um inteiro positivo tão grande que todos se tornem inteiros.

Parte IV

Algoritmos Polinomiais

Capítulo 14

Introdução aos algoritmos polinomiais

O Simplex é um algoritmo exponencial: o número de operações aritméticas que o algoritmo executa, no pior caso, ao receber o sistema A, b, c que define um problema canônico primal é da ordem de

$$m\,(m{+}n)\,2^n\,,$$

onde m é o número de linhas e n o número de colunas de A. A mesma observação vale para o Simplex-Chio. Talvez não seja razoável exigir que o número de operações aritméticas executadas por um algoritmo de programação linear seja limitado por um polinômio em m e n. Mas existem algoritmos de programação linear em que o número de operações aritméticas é limitado por um polinômio em

$$m\,, \quad n \quad \text{e} \quad \log \omega\,,$$

onde $\log \omega$ representa o número total de dígitos necessário para escrever todos os componentes de A, b, c. O presente capítulo descreve as camadas externas de um tal algoritmo polinomial. O núcleo, conhecido como algoritmo de Yamnitsky-Levin, será discutido no próximo capítulo.

A existência de um algoritmo polinomial para programação linear foi objeto de especulação por algum tempo. O primeiro algoritmo desse tipo foi descrito por Khachiyan [Kha79, GL81, PS82] e ficou conhecido como **algoritmo do elipsóide**. Outros algoritmos polinomiais, como o que pretendemos descrever, foram publicados mais tarde.

O algoritmo que discutiremos não é uma alternativa prática para o Simplex, mas tem profundas implicações teóricas (veja Schrijver [Sch86, Sch03] e Grötschel-Lovász-Schrijver [GLS88, GLS93]). Além disso, algumas de suas idéias são úteis em outros algoritmos polinomiais, como os "algoritmo de pontos interiores" [Kar84, Gon89, Gon92], que estão ganhado aceitação na prática.

14.1 Problemas CD, PV, V e PI

O algoritmo polinomial que queremos descrever resolve o problema canônico dual (mas é preciso lembrar que qualquer problema de programação linear pode ser transformado num problema canônico dual equivalente). Convém repetir a definição do

> **Problema** CD(A, c, b): *Dado um sistema A, c, b, encontrar um vetor y em $Y(A, c)$ que maximize yb.*

Como de hábito, $Y(A, c)$ é o poliedro canônico dual, ou seja, o conjunto de todos os vetores y que satisfazem a restrição $yA \leq c$.

Nosso algoritmo consiste em três "camadas" que envolvem um "núcleo". A camada externa reduz o problema CD ao seguinte

> **Problema do Ponto Viável** PV(A, c): *Dado um sistema A, c, encontrar um vetor y em $Y(A, c)$.*

A segunda camada reduz o problema PV ao

> **Problema da Viabilidade** V(A, c): *Dado um sistema A, c, decidir se $Y(A, c)$ é ou não é vazio.*

A camada interna reduz o problema V ao

> **Problema do Ponto Interior** PI (A, c): *Dado um sistema A, c, encontrar um vetor z tal que $z \cdot A_{[\,,j]} < c_{[j]}$ para todo j.*

Finalmente, o núcleo, conhecido como algoritmo de Yamnitsky-Levin, resolve o problema PI . algoritmo de Yamnitsky-Levin

Trataremos do núcleo no próximo capítulo; o presente capítulo cuidará das reduções. A **redução** de um problema a outro é um método que resolve o primeiro recorrendo a um hipotético algoritmo para o segundo.

Tamanho e complexidade. Antes de tratar das reduções é necessário adotar algumas hipóteses e convenções de notação. Suporemos que os componentes das matrizes e vetores que definem nossos problemas são números inteiros.[1] Essa hipótese não é muito restritiva: qualquer sistema com dados racionais pode ser transformado num sistema inteiro equivalente mediante multiplicação por um número inteiro positivo suficientemente grande.

O número de linhas de qualquer matriz A será denotado por $m(A)$. O número de colunas será denotado por $n(A)$. O tamanho de A (veja capítulo 13) será medido pelo produto $\omega(A)$ de todos os números da forma $1 + \alpha$, onde α é o valor absoluto de um componente de A. Analogamente, o tamanho ω de $m(A)$ $n(A)$ $\omega(A)$

[1] Mas as eventuais soluções do problema podem ter componentes fracionários.

qualquer vetor b será o produto de todos os números da forma $1+\beta$, onde β é o valor absoluto de um componente de b. Usaremos também as abreviaturas

$$\omega(A,b) = \omega(A)\,\omega(b) \qquad \text{e} \qquad \omega(A,b,c) = \omega(A)\,\omega(b)\,\omega(c)\,,$$

qualquer que seja o vetor c. O número total de dígitos binários de um sistema inteiro A, b, c é essencialmente igual a $\lg \omega(A,b,c)$, onde $\lg$ denota o logaritmo na base 2. $\lg$

Diremos que a **complexidade** de um algoritmo é o número de operações aritméticas que o algoritmo executa no pior caso.[2] Um algoritmo que age sobre um sistema A, b, c é **polinomial** se sua complexidade é limitada por um polinômio em $m(A)$, $n(A)$ e $\lg \omega(A,b,c)$.

14.2 Redução do CD ao PV

Suponha que dispomos de um algoritmo que resolve o problema PV. Como esse algoritmo pode ser usado para resolver o problema CD?

A resposta é relativamente simples desde que se conheça o teorema da dualidade. Para resolver o problema CD(A,c,b), basta encontrar um par $y\,, x$ de vetores tal que

$$yA \le c\,, \quad Ax = b\,, \quad x \ge o \quad \text{e} \quad cx \le yb\,. \tag{14.a}$$

Esta é uma instância do problema PV; para tornar isso mais claro, basta colocar as restrições na forma

$$\begin{aligned} yA \le c\,, \quad x\widetilde{A} \le b\,, \quad -x\widetilde{A} \le -b\,, \\ -xI \le o\,, \quad -yb + xc \le 0\,. \end{aligned} \tag{14.b}$$

(Veja figura 14.1.) O lema da dualidade 8.1 garante que para todo par $y\,, x$ que satisfaz (14.a) o vetor y é uma solução do problema CD(A,c,b). Reciprocamente, para toda solução y do problema CD, o teorema da dualidade 8.5 garante a existência de um vetor x tal que o par $y\,, x$ é solução de (14.a).

Complexidade. Digamos que A', c' é o sistema descrito implicitamente em (14.b). Suponha que a complexidade de nosso hipotético algoritmo para o problema PV(A', c') é limitada por um polinômio em $m(A')$, $n(A')$ e $\lg \omega(A', c')$. É fácil verificar que $m(A') = m(A) + n(A)$ e $n(A') = 2n(A) + 2m(A) + 1$. Ademais,

$$\omega(A') = \omega(A)^3\, 2^{n(A)}\, \omega(b)\,\omega(c) \quad \text{e} \quad \omega(c') = \omega(c)\,\omega(b)^2\,,$$

donde $\lg \omega(A', c') \le n(A) + 3 \lg \omega(A,c,b)$.

Portanto, a complexidade de nosso algoritmo para o problema CD(A,c,b) será limitada por um polinômio em $m(A)$, $n(A)$ e $\lg \omega(A,c,b)$.

[2] Uma definição mais realista deveria levar em conta o tempo total dispendido na execução de cada operação aritmética; os algoritmos mencionados na introdução podem ser formulados de modo que seu consumo total de tempo seja polinomial.

y	A	O	O	O	$-b$
x	O	$\tilde{A}$	$-\tilde{A}$	I	c
	c	b	$-b$	o	0

Figura 14.1: Construção do problema PV que equivale ao problema $\mathsf{CD}(A, c, b)$.

14.3 Redução do PV ao V

Suponha que dispomos de um algoritmo $\mathcal{A}$ que resolve o problema V. Eis como $\mathcal{A}$ pode ser usado para resolver o problema $\mathsf{PV}(A, c)$, ou seja, para encontar um vetor y em $Y(A, c)$:

> Use $\mathcal{A}$ para decidir se $Y(A, c)$ é vazio. Se $Y(A, c)$ não é vazio, use $\mathcal{A}$ repetidas vezes para determinar uma parte K de N que seja maximal com relação à seguinte propriedade:[3] existe um vetor y tal que
>
> $$yA \leq c \quad \text{e} \quad -yA_{[\,,K]} \leq -c_{[K]}\,.^{4}$$
>
> (Veja a figura 14.2.) Seja A' a matriz $A_{[\,,K]}$. Use o algoritmo de Gauss-Jordan (capítulo 2) para determinar matrizes F e G tais que $FG = I$ e GA' é escalonada. Seja P' a base de linhas e Q' uma base de colunas de GA'. Seja w^* o único vetor que satisfaz as equações A' P' Q' w^*
>
> $$w^*(GA')_{[\,,Q']} = c_{[Q']} \quad \text{e} \quad w^*{}_{[M-P']} = o\,.$$
>
> O vetor $y^* = w^*G$ está em $Y(A, c)$. □

Análise do algoritmo. O conjunto K foi construído de modo que $yA \leq c$ e $yA' = c'$ para algum vetor y, onde $c' = c_{[K]}$. Mostraremos a seguir que a maximalidade de K garante que

$$(GA)_{[M-P',\,]} = O\,. \tag{14.c}$$

(Na terminologia do apêndice D, página 185, o vetor y é básico em $Y(A, c)$. A identidade (14.c) afirma que K é um gerador de A. A demonstração da identidade é uma combinação das demonstrações de D.1 e D.3.)

Suponha por um instante que $(GA)_{[i,\,]} \neq o$ para algum i em $M - P'$. Seja v v

[3] Ou seja, nenhum superconjunto próprio de K tem a propriedade.
[4] Mas o algoritmo $\mathcal{A}$ não fornece o vetor y.

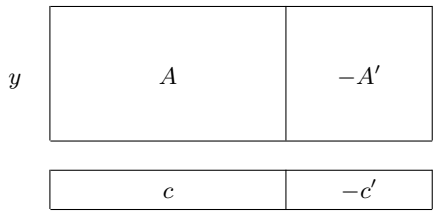

Figura 14.2: Construção do problema V que equivale ao problema PV(A, c). Legenda: $A' = A_{[\,,\,K]}$ e $c' = c_{[K]}$.

o vetor $G_{[i,\,]}$. Ajuste a notação, trocando v por $-v$ se necessário, de modo que $(vA)_{[j]}$ seja positivo para algum j em $N - K$. Seja λ o maior número tal que λ

$$\lambda \;\leq\; \frac{(c - yA)_{[j]}}{(vA)_{[j]}}$$

para todo j em $N-K$ tal que $(vA)_{[j]}$ é positivo. É fácil verificar que $yA+\lambda vA \leq c$. Como $(vA)_{[K]} = (GA)_{[i,\,K]} = o$, temos

$$(yA + \lambda vA)_{[K]} \;=\; c_{[K]}\,.$$

Ademais, em virtude da maximalidade de λ,

$$(yA + \lambda vA)_{[j]} \;=\; c_{[j]}$$

para algum j em $N - K$. Mas isso é incompatível com a maximalidade de K. A contradição prova (14.c).

Podemos mostrar agora que y^* está em $Y(A, c)$. Como $yA \leq c$, basta verificar que $y^*A = yA$. (Na terminologia do apêndice D, página 185, diríamos que y e y^* pertencem à mesma face minimal do poliedro. Se $P' = M$ então $y = y^*$ é um vértice do poliedro.) É claro que $y^*A = w^*GA$ e $yA = yFGA = wGA$, onde w é o vetor yF. Como $(GA)_{[M-P',\,]}$ é nula, temos

$$y^*A = w^*{}_{[P']}(GA)_{[P',\,]} \quad \text{e} \quad yA = w_{[P']}(GA)_{[P',\,]}\,.$$

Resta apenas verificar que $w^*{}_{[P']} = w_{[P']}$. Por definição, $w^*{}_{[P']}(GA')_{[P',\,Q']} = c_{[Q']}$. Por outro lado,

$$w_{[P']}(GA')_{[P',\,Q']} = w_{[P']}(GA)_{[P',\,Q']} = yA_{[\,,\,Q']} = c_{[Q']}\,,$$

uma vez que Q' é parte de K. Finalmente, como $(GA')_{[P',\,Q']}$ é uma matriz de bijeção, temos

$$w^*{}_{[P']} = w_{[P']}\,.$$

Segue daí que $y^*A = yA$, como queríamos demonstrar.

Complexidade. Para resolver o problema $\mathsf{PV}(A, c)$, nossa redução apela ao algoritmo $\mathcal{A}$ não mais que $1 + n(A)$ vezes. Em cada apelo, resolvemos um problema $\mathsf{V}(A', c')$ tal que

$$n(A') \leq 2n(A)\,, \quad m(A') = m(A) \quad \text{e} \quad \omega(A', c') \leq \omega(A, c)^2\,.$$

Se a complexidade do algoritmo $\mathcal{A}$ é limitada por um polinômio em $n(A')$, $m(A')$ e $\lg \omega(A', c')$, então a soma das complexidades em todos os apelos ao algoritmo será limitada por um polinômio em $n(A)$, $m(A)$ e $\lg \omega(A, c)$. A complexidade do algoritmo de Gauss-Jordan usado em nossa redução também é limitada por um polinômio. Em suma, a complexidade de nosso algoritmo para o problema $\mathsf{PV}(A, c)$ é limitada um polinômio em $m(A)$, $n(A)$ e $\lg \omega(A, c)$.

14.4 Redução do V ao PI

O problema $\mathsf{PI}\,(A', c')$ consiste em encontrar um *ponto interior* do poliedro $Y(A', c')$, ou seja, um vetor z tal que $zA'[\,, j] < c'[j]$ para cada j. Suponha que dispomos de um algoritmo para o problema PI e queremos resolver o problema da viabilidade para um sistema inteiro A, c. Adote as abreviaturas $n = n(A)$ e $\omega = \omega(A, c)$ e faça

$$\xi = n\omega\,, \quad A' = \xi A \quad \text{e} \quad c' = \xi c + u\,,$$

onde u é o vetor de 1s (todos os componentes de u são iguais a 1). Resolva o problema do ponto interior para o sistema A', c'. Se $Y(A', c')$ não tem ponto interior então o teorema abaixo garante que $Y(A, c)$ é vazio; caso contrário, o teorema garante que $Y(A, c)$ não é vazio (embora sem exibir, explicitamente, um elemento do poliedro).

Nossa redução depende do seguinte teorema, que é uma conseqüência simples do teorema da dualidade 13.2 para dados inteiros. O teorema afirma que o poliedro $Y(A, c)$ é vazio se e só se um poliedro auxiliar, ligeiramente maior, não tem pontos interiores.

Teorema 14.1 (da redução) *Para qualquer sistema inteiro A, c, o poliedro $Y(A, c)$ é vazio se e só se*

$$Y(\xi A\,,\ \xi c + u)$$

tem um ponto interior, onde $\xi = n\omega$.

DEMONSTRAÇÃO: Suponha que existe um vetor y tal que $yA \leq c$. É claro que $yA < c + \xi^{-1}u$ e portanto y é um ponto interior de $Y(\xi A, \xi c + u)$. Agora considere a recíproca. Suponha que $Y(\xi A, \xi c + u)$ tem um ponto interior z. É claro que

$$zA < c + \xi^{-1}u\,.$$

Considere o problema de encontrar um vetor y que satisfaça as restrições $yA \le c$. O teorema da dualidade 13.2 para dados inteiros garante que vale uma das seguintes alternativas:

> existem um inteiro δ entre 1 e ω e um vetor inteiro y com componentes entre $-\omega$ e ω tais que $yA \le \delta c$;
>
> existe um vetor inteiro x' com componentes entre 0 e ω tal que $Ax' = o$ e $cx' \le -1$.

Suponha por um momento que vale a segunda alternativa. Então, por um lado,

$$(c + \xi^{-1}u)x' \;>\; (zA)x' \;=\; z(Ax') \;=\; 0\,,$$

onde a primeira desigualdade vale porque x' não é nulo, uma vez que cx' não é nulo. Por outro lado,

$$(c + \xi^{-1}u)x' \;=\; cx' + \xi^{-1}ux' \;\le\; -1 + \xi^{-1}n\omega \;\le\; -1 + (n\omega)^{-1}n\omega \;=\; 0\,.$$

Esta contradição mostra que a segunda alternativa é impossível, e portanto vale a primeira. Como $yA \le \delta c$, o vetor $\delta^{-1}y$ está em $Y(A, c)$. □

A ênfase do teorema está menos nos pontos interiores que na possibilidade de aumentar ligeiramente o lado direito das restrições que definem um poliedro vazio sem que ele deixe de ser vazio. De fato, algumas alterações óbvias na demonstração acima provam que $Y(A, c)$ é vazio se e só se $Y(2\xi A, 2\xi c + u)$ é vazio.

Complexidade. Suponha que a complexidade de nosso algoritmo hipotético para o problema $\mathsf{PI}(A', c')$ é limitada por um polinômio em $n(A')$, $m(A')$ e $\lg \omega(A', c')$. Para resolver o problema $\mathsf{V}(A, c)$ como esboçamos acima é necessário tomar $A' = \xi A$ e $c' = \xi c + u$. Se observarmos que $m(A') = m(A)$ e $n(A') = n(A)$ e adotarmos as abreviaturas m e n respectivamente para esses dois números, teremos

$$\begin{aligned}
\omega' &= \omega(A')\,\omega(c') \\
&\le \xi^{\,mn}\,\omega(A)\,\xi^{\,n}\,\omega(c) \\
&= \xi^{\,mn+n}\,\omega \\
&= n^{\,mn+n}\,\omega^{\,mn+n+1}\,,
\end{aligned}$$

onde ω e ω' são abreviaturas para $\omega(A, c)$ e $\omega(A', c')$ respectivamente. Portanto

$$\lg \omega' = (mn + n)\lg n + (mn + n + 1)\lg \omega\,.$$

Assim, a complexidade do algoritmo que propusemos para o problema $\mathsf{V}(A, c)$ é limitada por um polinômio em $m(A)$, $n(A)$ e $\lg \omega(A, c)$.

14.5 Conclusão

Este capítulo mostrou como um algoritmo para o problema do ponto interior, PI, pode ser usado para resolver o problema canônico CD. O método consiste em uma redução ao problema PV, seguida de uma redução ao problema V, seguida de uma redução ao problema PI. Se o algoritmo para o problema $\mathsf{PI}(A', c')$ tiver complexidade limitada por um polinômio em $m(A')$, $n(A')$ e $\lg \omega(A', c')$ então a complexidade do algoritmo resultante para o problema $\mathsf{CD}(A, c, b)$ será limitada por um polinômio em $m(A)$, $n(A)$ e $\lg \omega(A, c)$.

A menos de casos excepcionais,[5] a complexidade do algoritmo de Yamnitsky-Levin (veja próximo capítulo) para o problema $\mathsf{PI}(A', c')$ é limitada pelo polinômio

$$70\,(m'+1)^4\,(m'+n'+1)\,\lg \omega'\,,$$

onde $m' = m(A')$, $n' = n(A')$ e $\omega' = \omega(A', c')$. Os termos mais significativos desse polinômio são

$$70\,m'^5 \lg \omega' \quad \text{e} \quad 70\,m'^4 n' \lg \omega'\,.$$

Se um problema $\mathsf{V}(A, c)$ for reduzido a um problema PI e esse último resolvido pelo algoritmo de Yamnitsky-Levin, o número total de operações aritméticas será limitado por um polinômio cujos termos mais significativos são

$$70\,m^6 n \lg \omega \quad \text{e} \quad 70\,m^5 n^2 \lg \omega\,,$$

onde $m = m(A)$, $n = n(A)$ e $\omega = \omega(A, c)$. Se um problema $\mathsf{PV}(A, c)$ for reduzido a um problema V e este for resolvido como sugerido acima, o número total de operações aritméticas será limitado por um polinômio cujos termos mais significativos são

$$280\,m^6 n^2 \lg \omega \quad \text{e} \quad 560\,m^5 n^3 \lg \omega\,,$$

onde $m = m(A)$, $n = n(A)$ e $\omega = \omega(A, c)$. Finalmente, no polinômio correspondente para o problema $\mathsf{CD}(A, c, b)$, o termo mais significativo será (*miserere nobis!*)

$$13440\,(m+n)^8 \lg \omega\,,$$

onde $m = m(A)$, $n = n(A)$ e $\omega = \omega(A, c, b)$. Esses cálculos, ainda que grosseiros, sugerem que nosso algoritmo dificilmente poderá competir com o Simplex na prática.

[5] Sistemas com menos que $7 + \lg m(A')$ componentes não-nulos.

Capítulo 15

Algoritmo de Yamnitsky-Levin

Este capítulo descreve um algoritmo polinomial para o problema do ponto interior, já enunciado no capítulo 14:

> **Problema** PI (A, c): *Dada uma matriz inteira A e um vetor inteiro c, encontrar um vetor z tal que $zA < c$.*

O algoritmo, publicado por Yamnitsky e Levin [YL82][Chv83, Sch86, Sch03], é semelhante ao célebre algoritmo do elipsóide [Kha79, GL81], mas, ao contrário daquele, opera apenas com números racionais. O número de operações aritméticas que o algoritmo de Yamnitsky-Levin executa é da ordem de

$$m^4 \, (m + n) \, \log \omega \, ,$$

onde m e n são os números de linhas e colunas de A respectivamente e $\log \omega$ representa o número total de dígitos necessário para escrever todos os componentes de A e c.

A concepção básica do algoritmo é simples: o poliedro $Y(A, c)$ é envolvido por um tetraedro suficientemente grande que é então progressivamente "encolhido" até que o centro do tetraedro seja um ponto interior do poliedro ou até que fique evidente que o poliedro não tem ponto interior. Embora a idéia pareça simples, sua realização técnica é bastante complexa.

15.1 Definições básicas

Os protagonistas principais do capítulo são uma matriz A sobre $M \times N$ e um vetor c sobre N. Suporemos que o sistema $A \, , c$ é inteiro, ou seja, que todos os componentes de A e c são números inteiros.

O **tamanho** de A (veja capítulo 13) é o produto de todos os números da forma $1 + \alpha$, onde α é o valor absoluto de um componente de A. O tamanho de c é definido de maneira análoga. O tamanho do sistema $A \, , c$ é o produto $\omega(A) \, \omega(c)$, onde $\omega(A)$ e $\omega(c)$ são os tamanhos de A e c respectivamente.

Como de hábito, $Y(A, c)$ denota o conjunto de todos os vetores y para os quais $yA \leq c$. Diremos que esse conjunto é o poliedro determinado por $A \, , c$.

Nosso problema é encontrar um **ponto interior** do poliedro, isto é, um vetor racional z tal que

ponto interior

$$zA \;<\; c\,.$$

A desigualdade deve ser entendida componente-a-componente: $zA_{[\,,j]} < c_{[j]}$ para cada índice j.

15.2 Tetraedros e seus volumes

A existência de um ponto interior em $Y(A,c)$ está intimamente relacionada com o "volume" de $Y(A,c)$: o poliedro tem um ponto interior se e só se seu "volume" não é nulo. Felizmente, não será necessário definir o "volume" de um poliedro arbitrário; basta definir o conceito para certos poliedros muito simples, que chamaremos tetraedros.

Um **tetraedro**[1] sobre M é o conjunto de todas as combinações convexas de $m+1$ vetores sobre M, onde $m = |M|$. Uma **combinação convexa** dos vetores $t_0, \ldots, t_m$ é qualquer vetor da forma

combinação convexa

$$\lambda_0\, t_0 + \cdots + \lambda_m t_m\,,$$

onde $\lambda_0, \ldots, \lambda_m$ são números não-negativos tais que $\lambda_0 + \cdots + \lambda_m = 1$. Os $m+1$ vetores que definem um tetraedro são os **vértices** ou **geradores** do tetraedro. O tetraedro gerado pelos vetores $t_0, \ldots, t_m$ será denotado por $[t_0, \ldots, t_m]$.

$[t_0, \ldots, t_m]$

Antes de definir o volume do tetraedro, convém ajustar nossa notação de modo que M seja o conjunto $\{1, \ldots, m\}$ e adotar a abreviatura $M{+}0 = \{0, 1, \ldots, m\}$. Feito esse ajuste, seja T a matriz definida pelas equações

$M{+}0$
T

$$T_{[i,\,0]} = 1 \quad \text{e} \quad T_{[i,\,M]} = t_i$$

para $i = 0, \ldots, m$. É claro que T é definida sobre $M{+}0 \times M{+}0$. Diremos que T é a **matriz do tetraedro** gerado pelos vetores $t_0, \ldots, t_m$. Graças ao ajuste de notação, a matriz é quadrada.

O **volume** do tetraedro gerado por $t_0, \ldots, t_m$ é, por definição, o valor absoluto do determinante da matriz do tetraedro. Se o valor absoluto de $\det(T)$ for denotado por $\mathrm{absdet}(T)$, podemos dizer que

volume

$$\mathrm{vol}\,[t_0, \ldots, t_m] \;=\; \mathrm{absdet}(T)\,.$$

O volume do tetraedro não depende da ordem em que tomamos seus vértices. De fato, se T' é a matriz do tetraedro gerado por uma permutação $t'_0, \ldots, t'_m$ de $t_0, \ldots, t_m$ então existe uma matriz de permutação J tal que $T' = JT$; como $\det(T') = \mathrm{sig}(J)\det(T) = \pm\det(T)$, temos $\mathrm{absdet}(T') = \mathrm{absdet}(T)$.

O número $\mathrm{vol}\,[t_0, \ldots, t_m]\,/m!$ é o volume do conjunto $[t_0, \ldots, t_m]$ no sentido geométrico. Nossa definição de volume deixa de lado o fator $1/m!$ porque o algoritmo de Yamnitsky-Levin só depende dos valores *relativos* dos volumes de certos tetraedros.

[1] A rigor, o termo *tetraedro* só é apropriado quando $m = 3$. Quando $m = 2$, o tetraedro é um triângulo. Quando $m = 1$, o tetraedro se reduz a um intervalo.

$$\begin{array}{rrr} -4 & -4 & -4 \\ 20 & -4 & -4 \\ -4 & 20 & -4 \\ -4 & -4 & 20 \end{array} \qquad \begin{array}{rrrr} 1 & -4 & -4 & -4 \\ 1 & 20 & -4 & -4 \\ 1 & -4 & 20 & -4 \\ 1 & -4 & -4 & 20 \end{array}$$

Figura 15.1: O volume do tetraedro gerado pelas linhas da primeira matriz é o valor absoluto do determinante da segunda matriz, ou seja, 24^3.

Propriedades do volume. O volume de um tetraedro pode ser calculado a partir das "arestas" $t_i - t_0$. Esse cálculo depende da **matriz reduzida** do tetraedro: trata-se da matriz $\bar{T}$ sobre $M \times M$ definida pelas equações $\bar{T}$

$$\bar{T}_{[i,\,]} = t_i - t_0\,.$$

Propriedade 15.1 (fórmula do volume) *Se $\bar{T}$ é a matriz reduzida do tetraedro gerado por $t_0, .., t_m$ então* $\operatorname{vol}[t_0, .., t_m] = \operatorname{absdet}(\bar{T})$.

DEMONSTRAÇÃO: Basta mostrar que o determinante de $\bar{T}$ é igual ao determinante da matriz T do tetraedro. A relação entre as duas matrizes é simples:

$$\bar{T} \;=\; (GT)_{[M,\,M]}\,,$$

onde G é a matriz elementar sobre $M{+}0 \times M{+}0$ com coluna saliente 0 definida pelas equações $G_{[0,\,0]} = 1$ e $G_{[M,\,0]} = -u$, onde u é o vetor de 1s ($u_{[j]} = 1$ para todo j). De acordo com propriedade 10.3, $\det((GT)_{[M,\,M]}) = \det(GT)$, uma vez que $(GT)_{[\,,0]} = Gu = I_{[\,,0]}$. De acordo com o teorema 26,

$$\det(GT) \;=\; \det(G)\det(T) \;=\; G_{[0,\,0]}\det(T) \;=\; \det(T)\,.$$

Logo, $\det(\bar{T}) = \det(T)$, como queríamos demonstrar. □

Essa fórmula do volume ajuda a calcular o volume do tetraedro que servirá de ponto de partida do algoritmo de Yamnitsky-Levin. Considere o tetraedro gerado pelos vetores $t_0 = \alpha u$ e $t_i = \alpha u + \beta I_{[i,\,]}$, onde u é um vetor de 1s (veja figura 15.1). Então

$$\operatorname{vol}[t_0, .., t_m] = \beta^m\,, \tag{15.a}$$

uma vez que a matriz reduzida do tetraedro é βI.

A análise do algoritmo de Yamnitsky-Levin depende de mais duas propriedades do volume de um tetraedro. A primeira dá condições suficientes para que o volume de um tetraedro seja nulo; a segunda mostra que se $[t'_0, .., t'_m] \subseteq [t_0, .., t_m]$ então $\operatorname{vol}[t'_0, .., t'_m] \leq \operatorname{vol}[t_0, .., t_m]$.

Propriedade 15.2 (do volume nulo) *Se $t_0 \cdot a = t_1 \cdot a = \cdots = t_m \cdot a$ para algum vetor não-nulo a sobre M então* $\operatorname{vol}[t_0, .., t_m] = 0$.

DEMONSTRAÇÃO: Seja $\bar{T}$ a matriz reduzida do tetraedro gerado por $t_0, \ldots, t_m$. De acordo com a propriedade 15.1, basta mostrar que $\det(\bar{T}) = 0$. Submeta $\bar{T}$ ao algoritmo de Gauss-Jordan (seção 2.3). O algoritmo produzirá matrizes F e G tais que $FG = I$ e a matriz $G\bar{T}$ é escalonada. Seja P a base de linhas da matriz $G\bar{T}$.

Suponha, tentativamente, que $P = M$ e portanto que $G\bar{T}$ é uma matriz de bijeção. De acodo com nossas hipóteses, $\bar{T}_{[i,\,]}a = t_i a - t_0 a = 0$ para cada i. Como $\bar{T}a$ é nulo, também $G\bar{T}a$ é nulo. Como a matriz $G\bar{T}$ é de bijeção, o vetor a é necessariamente nulo. o que é inconsistente com nossas hipóteses. Concluímos assim que $M - P$ não é vazio.

Seja i um elemento de $M - P$. Como o vetor $(G\bar{T})_{[i,\,]}$ é nulo, o determinante de $G\bar{T}$ é nulo. Como o determinante de $G\bar{T}$ é o produto dos determinantes de G e $\bar{T}$ (teorema 26), um desses dois últimos é nulo. Mas $\det(F)\det(G) = \det(I) = 1$, donde $\det(\bar{T}) = 0$. □

Propriedade 15.3 (monotonicidade) *Se os vetores $t'_0, \ldots, t'_m$ estão todos em $[t_0, \ldots, t_m]$ então* $\mathrm{vol}\,[t'_0, \ldots, t'_m] \le \mathrm{vol}\,[t_0, \ldots, t_m]$.

DEMONSTRAÇÃO: Sejam T e T' as matrizes dos tetraedros gerados, respectivamente, por $t_0, \ldots, t_m$ e $t'_0, \ldots, t'_m$. Basta mostrar que $\mathrm{absdet}(T') \le \mathrm{absdet}(T)$.

Por hipótese, cada vetor t'_i é uma combinação convexa de $t_0, \ldots, t_m$. Portanto, cada linha da matriz T' é uma combinação convexa das linhas de T. Em outras palavras, existe uma matriz $\check{G}$ sobre $M{+}0 \times M{+}0$ tal que

$$T' = \check{G}T\,, \quad \check{G}_{[i,\,]} \ge o \quad \text{e} \quad \textstyle\sum_j \check{G}_{[i,j]} = 1$$

para cada i em $M{+}0$. Portanto, o determinante de T' é o produto dos determinantes de $\check{G}$ e T (teorema 26). Mas

$$\mathrm{absdet}(\check{G}) \le \textstyle\prod_i (\sum_j \check{G}_{[i,j]}) = 1$$

de acordo com a delimitação 10.9. Logo, $\mathrm{absdet}(T') \le \mathrm{absdet}(T)$. □

15.3 Teorema do tetraedro interior

A seção anterior começou com a seguinte observação informal: um poliedro possui um ponto interior se e só se o seu "volume" não é nulo. Agora estamos em condições de fazer uma afirmação mais precisa: um poliedro possui um ponto interior se e só se inclui um tetraedro de volume não-nulo. O teorema abaixo prova esta afirmação é dá estimativas do volume do tetraedro e da norma de seus vértices. A **norma** de um vetor z é o maior componente, em valor absoluto, de z:

$$|z| \;=\; \max_j |z_{[j]}|\,.$$

O teorema adota, tacitamente, a convenção $M = \{1, \ldots, m\}$ que já usamos na seção anterior.

Teorema 15.4 (do tetraedro interior) *Seja A, c um sistema* inteiro *sobre $M \times N$. Se $Y(A,c)$ possui um ponto interior então existem um* inteiro δ *entre*[2] 1 *e $\ddot{\omega}$ e vetores* inteiros $z_0, \ldots, z_m$ *sobre M dotados das seguintes propriedades:*

$$\text{vol}\,[z_0/\delta, \ldots, z_m/\delta] \;\geq\; 1/\ddot{\omega}^m$$

e, para cada i, $z_i/\delta \;\in\; Y(A,c)$ e $|z_i| \;\leq\; 2\ddot{\omega} + 1$, onde $\ddot{\omega}$ é o número $4\,\omega(A)^2\omega(c)^2$.

DEMONSTRAÇÃO: Para qualquer inteiro positivo δ, é evidente que $Y(A, \delta c)$ é o conjunto de todos os vetores da forma δy com y em $Y(A,c)$. Se δ for suficientemente grande, alguns dos vetores em $Y(A, \delta c)$ terão componentes inteiros. São desse tipo os vetores $z_0, \ldots, z_m$ que estamos procurando. O vetor z_0, em particular, deverá estar suficientemente afastado de algumas das fronteiras do poliedro $Y(A, \delta c)$. A prova da existência de um tal z_0 será delegada, em parte, ao lema 15.5 abaixo.

Seja a o vetor definido da seguinte maneira para cada j em N: se $A_{[\,,j]} \leq o$ então $a_{[j]} = 0$ senão $a_{[j]} = \max_i A_{[i,j]}$. É óbvio que

$$a \geq o \quad \text{e} \quad a \geq A_{[i,\,]}$$

para cada i em M. Ademais, a é inteiro uma vez que A é inteira. A definição de a garante que para cada j existe i tal que $a_{[j]} \leq |A_{[i,j]}|$. Segue daí imediatamente que

$$\omega(a) \;\leq\; \omega(A)\,.$$

Nossa primeira tarefa é a determinação de um vetor inteiro z_0 e um número δ tais que a diferença entre δc e $z_0 A$ seja pelo menos a. Digamos que z é um ponto interior de $Y(A,c)$ e defina o número ζ da seguinte maneira: se a é nulo então $\zeta = 1$ senão ζ é o maior número racional que satisfaz a desigualdade $zA + \zeta a \leq c$. Como z é um ponto interior, temos

$$\zeta > 0\,.$$

Nestas condições, o lema 15.5 abaixo garante a existência de um inteiro δ, um inteiro ζ_0 e um vetor inteiro z_0 tais que

$$\zeta_0 \geq 1\,, \quad z_0 A + \zeta_0 a \leq \delta c\,, \quad 1 \leq \delta \leq 4\dot{\omega} \quad \text{e} \quad |z_0| \leq 8\dot{\omega}\,,$$

onde $\dot{\omega} = \omega(A)\omega(a)\omega(c)$. Como $a \geq o$ e $\omega(a) \leq \omega(A)$, temos

$$z_0 A + a \;\leq\; \delta c\,, \quad 1 \leq \delta \leq \ddot{\omega} \quad \text{e} \quad |z_0| \leq 2\ddot{\omega}\,.$$

(Note que z_0 pertence a $Y(A, \delta c)$ mas não é necessariamente um ponto interior.)

[2] A expressão "δ está entre α e β" deve ser entendida como $\alpha \leq \delta \leq \beta$.

Agora que temos um dos vértices do tetraedro, podemos construir os demais: para $i = 1, \ldots, m$,

$$z_i \;=\; z_0 + I_{[i,\,]}\,.$$

É evidente que cada z_i é inteiro e $|z_i| \le 2\ddot{\omega} + 1$. Como $a \ge A_{[i,\,]}$, temos

$$z_i A \;=\; z_0 A + A_{[i,\,]} \;\le\; z_0 A + a \;\le\; \delta c$$

para cada i em M. Como $a \ge o$, podemos dizer que $z_0 A \le z_0 A + a \le \delta c$. Portanto, cada um dos vetores z_i está em $Y(A, \delta c)$.

Finalmente, é preciso calcular o volume do tetraedro gerado pelos vetores $z_0/\delta, \ldots, z_m/\delta$. Seja $\bar{Z}$ a matriz reduzida do tetraedro. De acordo com a propriedade 15.1,

$$\mathrm{vol}\,[z_0/\delta, \ldots, z_m/\delta] \;=\; \mathrm{absdet}(\delta^{-1}\bar{Z})\,.$$

Como $\bar{Z}_{[i,\,]} = z_i - z_0 = I_{[i,\,]}$ para cada i, temos

$$\mathrm{absdet}(\delta^{-1}\bar{Z}) \;=\; \mathrm{absdet}(\delta^{-1} I) \;=\; \delta^{-m} \;\ge\; \ddot{\omega}^{-m}\,,$$

uma vez que $\delta \le \ddot{\omega}$. □

O teorema que acabamos de demonstrar tem a seguinte conseqüência imediata: se $Y(A, c)$ é parte de um tetraedro gerado por $t_0, \ldots, t_m$ e $\mathrm{vol}\,[t_0, \ldots, t_m] < 1/\ddot{\omega}^m$ então $Y(A, c)$ não tem ponto interior. Esta é, essencialmente, a "condição de parada" do algoritmo de Yamnitsky-Levin.

Lema do ponto profundo. Para completar a demonstração do teorema, é preciso estabelecer o seguinte lema, que é uma mera aplicação da versão inteira do teorema da dualidade 13.2. Grosso modo, o lema mostra que todo poliedro dotado de um ponto interior possui um ponto "profundo", ou seja, um ponto suficientemente afastado das fronteiras.

Lema 15.5 *Seja A uma matriz* inteira *sobre $M \times N$ e sejam a e c vetores* inteiros *sobre N. Se existe um número racional ζ e um vetor racional z tais que*

$$\zeta > 0 \quad \textit{e} \quad zA + \zeta a \le c$$

então existem um inteiro δ, um inteiro ζ^* e um vetor inteiro z^* tais que

$$\zeta^* \ge 1\,, \quad z^*A + \zeta^* a \le \delta c\,, \quad 1 \le \delta \le 4\dot{\omega} \quad \textit{e} \quad |z^*| \le 8\dot{\omega}\,,$$

onde $\dot{\omega} = \omega(A)\omega(a)\omega(c)$.

DEMONSTRAÇÃO: Considere o problema de encontrar um vetor z^* e um número ζ^* que

$$\text{maximizem } \zeta^* \text{ sujeito a } z^*A + \zeta^* a \le c \text{ e } \zeta^* \ge 0\,. \tag{15.b}$$

Convém mostrar explicitamente que este problema é canônico dual. Seja A' a matriz definida por

$$A'[M,N] = A\,, \quad A'[0,N] = a\,, \quad A'[M,0] = o \quad \text{e} \quad A'[0,0] = -1\,.$$

É claro que A' é definida sobre $M{+}0 \times N{+}0$; a notação está sendo ajustada de modo que 0 não esteja em N. Sejam b' e c' os vetores definidos por

$$b'[M] = o\,, \quad b'[0] = 1\,, \quad c'[N] = c \quad \text{e} \quad c'[0] = 0\,.$$

Nosso problema (15.b) é essencialmente igual ao problema canônico dual $\mathsf{CD}(A', c', b')$, que consiste em maximizar $y'b'$ sujeito a $y'A' \le c'$. Para qualquer solução y' do segundo problema, o par $y'[M], y'[0]$ é uma solução do primeiro.

Observe que $\omega(A')\omega(c')\omega(b') = 4\dot\omega$, uma vez que $\omega(A') = 2\omega(a)\omega(A)$, $\omega(b') = 2$ e $\omega(c') = \omega(c)$. De acordo com a versão inteira 13.2 do teorema da dualidade, existe um inteiro δ entre 1 e $4\dot\omega$ para o qual vale uma das seguintes alternativas: existem vetores inteiros y' e x' tais que

$$\begin{gathered} y'A' \le \delta c'\,, \quad A'x' = \delta b'\,, \quad x' \ge o\,, \quad y'b' = c'x'\,, \\ |y'| \le 4\dot\omega\,, \quad |x'| \le 4\dot\omega\,, \end{gathered}$$

ou existem vetores inteiros y' e y'' tais que

$$\begin{gathered} y'A' \le \delta c'\,, \quad y''A'' \le o\,, \quad y''b' \ge 1\,, \\ |y'| \le 4\dot\omega\,, \quad |y''| \le 4\dot\omega\,. \end{gathered}$$

Suponha inicialmente que vale a primeira das alternativas. Defina $z^* = y'[M]$ e $\zeta^* = y'[0]$. Observe que $z^*A + \zeta^* a \le \delta c$ pois $y'A' \le \delta c'$. Observe também que

$$\zeta^* \ge 1$$

pelos motivos que passamos a expor. Seja z' o vetor definido por $z'[M] = z$ e $z'[0] = \zeta$. Como z' satisfaz as restrições do problema $\mathsf{CD}(A', b', c')$, o lema da dualidade 8.1 garante que $z'b' \le c'x'$. Logo,

$$\zeta^* = y'[0] = y'b' = c'x' \ge z'b' = z'[0] = \zeta\,.$$

Como ζ é positivo e y' é inteiro, temos $\zeta^* \ge 1$.

Suponha agora que vale a segunda alternativa e adote $z^* = y'[M] + y''[M]$ e $\zeta^* = y'[0] + y''[0]$. É claro que

$$\zeta^* \ge y''[0] = y''b' \ge 1 \quad \text{e} \quad |z^*| \le |y'| + |y''| \le 8\dot\omega\,.$$

Ademais, $z^*A + \zeta^* a \le \delta c$ pois $y'A' + y''A' \le \delta c'$. □

15.4 Algoritmo

O objetivo do algoritmo é encontrar um ponto interior do poliedro $Y(A, c)$. O algoritmo envolve o poliedro num tetraedro suficientemente grande,[3] que é então progressivamente "encolhido". A cada iteração, o algoritmo verifica se o centro do tetraedro é um ponto interior do poliedro; em caso negativo, o tetraedro é deformado de modo que continue envolvendo o poliedro mas sofra uma redução de volume. Quando o volume do tetraedro se torna suficientemente pequeno, o algoritmo conclui que o poliedro não tem pontos interiores.

O algoritmo não se aplica a matrizes com menos que duas linhas; mas nesse caso o problema pode ser resolvido por meios elementares (veja apêndice deste capítulo).

> **Algoritmo de Yamnitsky-Levin** *Recebe uma matriz* inteira A *sobre* $M \times N$ *com* $|M| \geq 2$ *e um vetor* inteiro c *sobre* N*; devolve um vetor racional* t *tal que* $t \cdot A < c$ *ou pára sem nada devolver se um tal vetor não existe.*

Ajuste a notação de modo que $M = \{1, .., m\}$. Adote a abreviatura ω para o tamanho do sistema A, c e a abreviatura $\ddot{\omega}$ para a expressão $4\omega^2$. Em suma, $m \geq 2$

$$\omega = \omega(A)\,\omega(c) \quad \text{e} \quad \ddot{\omega} = 4\omega^2\,.$$

Seja $t_0^\star$ o vetor $-4\ddot{\omega}u$, onde u é o vetor de 1s (ou seja, $u_{[j]} = 1$ para todo j em M). $t_0^\star$
Para $i = 1, .., m$, seja $t_i^\star$ o vetor $t_i^\star$

$$-4\ddot{\omega}u + 8\ddot{\omega}mI_{[i,\,]}\,.$$

Cada iteração do algoritmo começa com vetores racionais $t_0, .., t_m$ sobre M e $t_0, .., t_m$
um número racional θ. A primeira iteração começa com $t_i = t_i^\star$ para cada i e θ
com $\theta = (8\ddot{\omega}m)^m$. Cada iteração consiste no seguinte:

CASO 1: $\theta \geq 1/\ddot{\omega}^m$.
Seja t o vetor $\frac{1}{m+1}(t_0 + \cdots + t_m)$. t

> CASO 1.1: $t \cdot A_{[\,,k]} < c_{[k]}$ *para cada* k *em* N.
> Devolva t e pare.
>
> CASO 1.2: $t \cdot A_{[\,,k]} \geq c_{[k]}$ *para algum* k *em* N. k
>
> > CASO 1.2A: $A_{[\,,k]} = o$.
> > Pare sem nada devolver.
> >
> > CASO 1.2B: $A_{[\,,k]} \neq o$.
> > Para cada $i = 0, 1, .., m$, seja π_i o número $(t - t_i) \cdot A_{[\,,k]}$.
> > Escolha h de modo que π_h seja máximo. h

[3] Para efeito desta descrição informal, suponha que o poliedro é limitado (veja exercício 15.4, página 159).

Para $i = 0, 1, \ldots, m$, seja ρ_i o número $1 - \dfrac{1}{m^2 \pi_h}\, \pi_i$.

Sejam $t'_0, \ldots, t'_m$ os vetores definidos pelas equações $t'_h = t_h$ e $t'_i = \dfrac{1}{\rho_i}\, t_i + (1 - \dfrac{1}{\rho_i})\, t_h$ para cada i distinto de h.

Seja θ' o número $\dfrac{\rho_h}{\prod_i \rho_i}\, \theta$.

Comece nova iteração com $t'_0, \ldots, t'_m$ e θ' nos papéis de $t_0, \ldots, t_m$ e θ.

CASO 2: $\theta < 1/\ddot{\omega}^m$.

Pare sem nada devolver. □

Eis algumas observações informais de caráter geométrico que podem ajudar a tornar o algoritmo menos misterioso:

O tetraedro gerado por $t^\star_0, \ldots, t^\star_m$ é tão grande que contém todos os eventuais vértices de $Y(A, c)$. Se $Y(A, c)$ for limitado (veja exercício 15.4, página 159) então $Y(A, c)$ é parte de $[t^\star_0, \ldots, t^\star_m]$.

O número θ é o volume do tetraedro gerado por $t_0, \ldots, t_m$.

No caso 1.2B, o vetor t é o centro do tetraedro gerado por $t_0, \ldots, t_m$ e o número π_i é a projeção do vetor $t - t_i$ na direção $A_{[\,,\,k]}$.

A transformação de $t_0, \ldots, t_m$ em $t'_0, \ldots, t'_m$ deixa fixo o vértice t_h do tetraedro e multiplica por $1/\rho_i$ a aresta que liga os vértices t_i e t_h.

Cada ρ_i é positivo e pode ser maior que 1 ou menor que 1. Mas o fator $\rho_h / \prod_i \rho_i$ é estritamente menor que 1. Portanto, o volume θ' do novo tetraedro é estritamente menor que θ.

Operação de pivotação. Diremos que a operação que transforma $t_0, \ldots, t_m$ em $t'_0, \ldots, t'_m$ no caso 1.2B é uma **pivotação** em torno de h. A operação de pivotação pode ser representada de forma matricial, como mostraremos a seguir. Sejam T e T' as matrizes dos tetraedros gerados por $t_0, \ldots, t_m$ e $t'_0, \ldots, t'_m$ respectivamente. Seja $\check{G}$ a matriz sobre $M{+}0 \times M{+}0$ definida pelas equações

pivotação

T T'

$\check{G}$

$$\check{G}_{[\,,\,i]} = \frac{1}{\rho_i}\, I_{[\,,\,i]}\,, \quad \check{G}_{[i,\,h]} = 1 - \frac{1}{\rho_i}\,, \quad \check{G}_{[h,\,h]} = 1\,,$$

onde as duas primeiras equações valem para cada i distinto de h. (Veja a figura 15.2). A matriz está bem definida pois $\rho_i \neq 0$ para todo i, como veremos adiante. Diremos que $\check{G}$ é a **matriz de pivotação**. É fácil verificar que

$$T' \;=\; \check{G} T \tag{15.c}$$

(em particular, $T'_{[\,,\,0]} = u = \check{G} T_{[\,,\,0]}$). Também é fácil constatar que $\det(\check{G}) = \rho_h / \prod_i \rho_i$, donde

$$\theta' = \det(\check{G})\, \theta\,. \tag{15.d}$$

A matriz $\check{G}$ é inversível. De fato, se $\check{F}$ é a matriz definida por $\check{F}$

$$\check{F}[\,,i] = \rho_i\, I[\,,i]\,, \quad \check{F}[i,h] = 1-\rho_i\,, \quad \check{F}[h,h] = 1\,, \tag{15.e}$$

onde as duas primeiras equações valem para cada i distinto de h, então $\check{F}\check{G} = \check{G}\check{F} = I$.

$$\begin{array}{cccccc} \rho_0 & 0 & 0 & 0 & 0 & 1-\rho_0 \\ 0 & \rho_1 & 0 & 0 & 0 & 1-\rho_1 \\ 0 & 0 & \rho_2 & 0 & 0 & 1-\rho_2 \\ 0 & 0 & 0 & \rho_3 & 0 & 1-\rho_3 \\ 0 & 0 & 0 & 0 & \rho_4 & 1-\rho_4 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array} \qquad \begin{array}{cccccc} 1/\rho_0 & 0 & 0 & 0 & 0 & 1-1/\rho_0 \\ 0 & 1/\rho_1 & 0 & 0 & 0 & 1-1/\rho_1 \\ 0 & 0 & 1/\rho_2 & 0 & 0 & 1-1/\rho_2 \\ 0 & 0 & 0 & 1/\rho_3 & 0 & 1-1/\rho_3 \\ 0 & 0 & 0 & 0 & 1/\rho_4 & 1-1/\rho_4 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array}$$

Figura 15.2: Exemplo com $m = h = 5$. O lado esquerdo da figura descreve $\check{F}$; o lado direito descreve $\check{G}$.

15.5 Invariantes

Para compreender o algoritmo basta observar as seguintes propriedades invariantes

No início de cada iteração,

(i1) *se* $yA \le c$ *e* $y \in [t_0^\star, .., t_m^\star]$ *então* $y \in [t_0, .., t_m]$,
(i2) $\text{vol}\,[t_0, .., t_m] = \theta$,
(i3) $\theta > 0$.

O primeiro invariante afirma[4] que $Y(A,c) \cap [t_0^\star, .., t_m^\star]$ é parte de $[t_0, .., t_m]$. Os invariantes (i1) e (i3) são obviamente verdadeiros no início da primeira iteração. O invariante (i2) também vale no início da primeira iteração, pois $\text{vol}\,[t_0^\star, .., t_m^\star] = (8\ddot{\omega}m)^m = \theta$ de acordo com (15.a). A validade dos invariantes no início das demais iterações será discutida nas próximas seções.

A prova dos invariantes e a estimativa do número de iterações dependem muito mais da geometria de $[t_0, .., t_m]$ que do sistema A, c. Por outro lado, a análise da última iteração não depende de $t_0, .., t_m$ mas apenas de A, c.

15.6 O algoritmo está bem definido

Para garantir que o algoritmo está bem definido é preciso verificar que $\pi_h \neq 0$ e $\rho_i \neq 0$ para todo i em cada iteração. Vamos mostrar que estas condições estão satisfeitas desde que (i2) e (i3) sejam válidas no início da iteração.

[4] Se $Y(A,c)$ for limitado então (i1) afirma que $Y(A,c)$ é parte de $[t_0, .., t_m]$, pois nesse caso $Y(A,c)$ é parte de $[t_0^\star, .., t_m^\star]$.

Comecemos com a seguinte observação. Em cada ocorrência do caso 1.2B temos $\sum \pi_i = \sum (t - t_i) A_{[\,,k]}$. Essa última expressão é igual a $(m{+}1)tA_{[\,,k]} - \sum t_i A_{[\,,k]}$. Como $(m{+}1)t = \sum t_i$, temos necessariamente

$$\sum \pi_i = 0\,. \tag{15.f}$$

Lema 15.6 *Se* (i2) *e* (i3) *valem no início de uma iteração em que ocorre o caso* 1.2B *então* $\pi_h > 0$.

DEMONSTRAÇÃO: Como $\sum \pi_i = 0$, a maximalidade de π_h garante que π_h não é negativo. Suponha por um instante que π_h é nulo. Então $\pi_i = 0$ para todo i, donde $tA_{[\,,k]} = t_i A_{[\,,k]}$ para todo i, e portanto

$$t_0 A_{[\,,k]} \;=\; t_1 A_{[\,,k]} \;=\; \cdots \;=\; t_m A_{[\,,k]}\,.$$

Como $A_{[\,,k]}$ não é nulo por definição do caso 1.2B, a propriedade 15.2 do volume nulo garante que

$$\text{vol}\,[t_0, \ldots, t_m] \;=\; 0\,.$$

Mas isso é inconsistente com os invariantes (i2) e (i3). Portanto, a hipótese $\pi_h = 0$ é insustentável e devemos ter $\pi_h > 0$. □

Suponha que (i2) e (i3) valem no início de uma iteração em que ocorre o caso 1.2B. O lema que acabamos de mostrar, mais a maximalidade de π_h e a hipótese $m \geq 2$, permitem concluir que ρ_i está bem definido e também que $m \geq 2$

$$\rho_i > 0 \tag{15.g}$$

para todo i. Finalmente, convém calcular $\sum \rho_i$. Como $\sum \rho_i = (m+1) + \sum \pi_i/(m^2 \pi_h)$ e $\sum \pi_i = 0$, temos

$$\sum \rho_i = m + 1\,. \tag{15.h}$$

15.7 Última iteração

Suponha que os invariantes (i1) a (i3) valem no início da última iteração do algoritmo, quando ocorrem os casos 1.1, 1.2A ou 2. Então o algoritmo dá a resposta correta, como passamos a mostrar.

Análise do caso 1.1. Suponha que $tA < c$. Então t é ponto interior de $Y(A, c)$. Assim, ao devolver t, o algoritmo está se comportando como previsto.

Análise do caso 1.2A. Suponha que $A_{[\,,k]}$ é nulo e $tA_{[\,,k]} \geq c_{[k]}$ para algum k. Então

$$yA_{[\,,k]} = 0 \geq c_{[k]}$$

para todo vetor y. Logo, o poliedro $Y(A, c)$ não tem ponto interior. Assim, ao parar sem nada devolver, o algoritmo está se comportando da maneira prevista.

Análise do caso 2. Suponha, finalmente, que $\theta < 1/\ddot{\omega}^m$. Então, de acordo com o invariante (i2), $\text{vol}\,[t_0,..,t_m] < 1/\ddot{\omega}^m$. Suponha que, ao contrário do que afirma o algoritmo, $Y(A,c)$ tem um ponto interior. Então, de acordo com o teorema 15.4 do tetraedro interior, existem um inteiro δ entre 1 e $\ddot{\omega}$ e vetores inteiros $z_0,..,z_m$ tais que

$$\text{vol}\,[z_0/\delta,..,z_m/\delta] \;\geq\; 1/\ddot{\omega}^m$$

e, para cada i,

$$z_i/\delta \in Y(A,c) \quad \text{e} \quad |z_i| \;\leq\; 2\ddot{\omega}+1\,.$$

A última desigualdade garante que cada vetor z_i/δ está no tetraedro gerado por $t^\star_0,..,t^\star_m$. De fato,

$$z_i[j]/\delta \;\geq\; (-2\ddot{\omega}-1)/\delta \;\geq\; -2\ddot{\omega}-1 \;\geq\; -4\ddot{\omega}$$

para todo j em M e

$$\textstyle\sum_j z_i[j]/\delta \;\leq\; m(2\ddot{\omega}+1)/\delta \;\leq\; m(2\ddot{\omega}+1) \;\leq\; 4m\ddot{\omega}\,.$$

Como $Y(A,c) \cap [t^\star_0,..,t^\star_m] \subseteq [t_0,..,t_m]$ em virtude do invariante (i1), concluímos que z_i/δ está em $[t_0,..,t_m]$ para cada i. Nestas circunstâncias, a propriedade 15.3 da monotonicidade garante que

$$\text{vol}\,[z_0/\delta,..,z_m/\delta] \;\leq\; \text{vol}\,[t_0,..,t_m]\,.$$

Segue-se que $\text{vol}\,[t_0,..,t_m] \geq 1/\ddot{\omega}^m$, o que é inconsistente com a definição do caso 2. Portanto, a existência de um ponto interior em $Y(A,c)$ é insustentável. Assim, ao parar sem nada devolver o algoritmo está se comportando como prometeu.

15.8 Demonstração dos invariantes

Suponha que os invariantes (i1), (i2) e (i3) valem no início de uma iteração qualquer que não a última. Para mostrar que estas propriedades continuam valendo no início da próxima iteração, basta verificar os seguintes fatos.

Fato 15.7 *No fim do caso* 1.2B, $\text{vol}\,[t'_0,..,t'_m] = \theta'$ *e* $\theta' > 0$.

DEMONSTRAÇÃO: Considere a matriz de pivotação $\check{G}$ e observe que

$$\det(\check{G}) \;=\; \rho_h / \textstyle\prod \rho_i\,.$$

Em virtude de (15.g), $\det(\check{G})$ é positivo. Como $\theta' = \det(\check{G})\;\theta$, o invariante (i2) garante que θ' é positivo. Isto conclui a demonstração da segunda parte. Para demonstrar a primeira parte basta verificar que

$$\text{vol}\,[t'_0,..,t'_m] = \det(\check{G})\,\text{vol}\,[t_0,..,t_m]\,.$$

Sejam T e T' as matrizes dos tetraedros gerados por $t_0, \ldots, t_m$ e $t'_0, \ldots, t'_m$ respectivamente. Como já observamos em (15.c), $T' = \check{G}T$. Pelo teorema 26, o determinante de T' é o produto dos determinantes de $\check{G}$ e T. Portanto,

$$\begin{aligned} \operatorname{vol}[t'_0, \ldots, t'_m] &= \operatorname{absdet}(T') \\ &= \det(\check{G}) \operatorname{absdet}(T) \\ &= \det(\check{G}) \operatorname{vol}[t_0, \ldots, t_m], \end{aligned}$$

como queríamos demonstrar. □

Fato 15.8 *No fim do caso* 1.2B*, se* $yA_{[\,,k]} \leq c_{[k]}$ *e* y *está em* $[t_0, \ldots, t_m]$ *então* y *está em* $[t'_0, \ldots, t'_m]$.

DEMONSTRAÇÃO: Seja y um ponto do tetraedro gerado por $t_0, \ldots, t_m$. Se T denota a matriz do tetraedo, podemos dizer que existe um vetor l sobre $M{+}0$ tal que

$$lT_{[\,,M]} = y, \quad lT_{[\,,0]} = 1 \quad \text{e} \quad l \geq o.$$

Seja $\check{F}$ a inversa de $\check{G}$ definida em (15.e) e seja l' o vetor $l\check{F}$. Mostraremos a seguir que se $yA_{[\,,k]} \leq c_{[k]}$ então

$$l'T'_{[\,,M]} = y, \quad lT'_{[\,,0]} = 1 \quad \text{e} \quad l' \geq o,$$

onde T' é a matriz do tetraedro gerado por $t'_0, \ldots, t'_m$; isso provará que y está em $[t'_0, \ldots, t'_m]$. As duas primeiras relações seguem imediatamente de (15.c):

$$l'T' = (l\check{F})(\check{G}T) = l(\check{F}\check{G})T = lT.$$

Na demonstração da desigualdade $l' \geq 0$, vamos adotar as abreviaturas $\lambda_i = l_{[i]}$ e $\lambda'_i = l'_{[i]}$. Para cada i distinto de h temos

$$\lambda'_i = l\check{F}_{[\,,i]} = \rho_i \lambda_i \geq 0,$$

uma vez que ρ_i não é negativo de acordo com (15.g). Resta apenas mostrar que λ'_h não é negativo. Como

$$\lambda'_h = l\check{F}_{[\,,h]} = \rho_h \lambda_h + \textstyle\sum(1 - \rho_i)\lambda_i$$

e ρ_h não é negativo em virtude de (15.g), basta mostrar que $\sum(1 - \rho_i)\lambda_i$ não é negativo. Isso equivale a mostrar que $\sum \pi_i \lambda_i$ não é negativo, uma vez que $\pi_i = m^2 \pi_h (1 - \rho_i)$ e π_h é positivo em virtude do lema 15.6. Mas

$$\begin{aligned} \textstyle\sum \lambda_i \, \pi_i &= \textstyle\sum \lambda_i \, (t - t_i) \, A_{[\,,k]} \\ &= (\textstyle\sum \lambda_i t - \sum \lambda_i t_i) \, A_{[\,,k]} \\ &= (t - y) \, A_{[\,,k]} \\ &= tA_{[\,,k]} - yA_{[\,,k]} \\ &\geq c_{[k]} - c_{[k]} \\ &= 0, \end{aligned}$$

uma vez que $tA_{[\,,k]} \geq c_{[k]}$ por definição do caso 1.2 e $yA_{[\,,k]} \leq c_{[k]}$ por hipótese. Com isto, concluímos a demonstração de que λ'_h não é negativo. □

15.9 Número de iterações

O número de iterações depende fundamentalmente do número de ocorrências do caso 1.2B. Em cada ocorrência desse caso, θ é multiplicado por

$$\frac{\rho_h}{\prod \rho_i}\,.$$

Diremos que esse número é o **fator de contração** da iteração e mostraremos que ele é significativamente menor que 1. É mais cômodo manipular o inverso do fator de contração. Mostraremos que o logaritmo desse inverso é maior que o número positivo $1\,/\,2(m+1)^2$. contração

Delimitação 15.9 (do fator de contração) *Em cada ocorrência do caso* 1.2B*,*

$$\frac{\prod \rho_i}{\rho_h} \;\geq\; (1-\frac{1}{m^2})^m\,(1-\frac{1}{m})^{-1}\,.$$

DEMONSTRAÇÃO: Para comparar $\prod \rho_i$ com $\sum \rho_i$, convém demonstrar o seguinte lema:

$$\begin{aligned}&\text{se } \sigma_i \geq \varepsilon \geq 0 \text{ para } i = 0,\ldots,k\\ &\text{então } \sigma_0 \cdots \sigma_k \geq (\sigma_0 + \cdots + \sigma_k - k\varepsilon)\,\varepsilon^k\,.\end{aligned}$$

A proposição é trivialmente verdadeira quando k é nulo. Suponha agora que k é positivo e adote como hipótese de indução a desigualdade $\sigma_0 \cdots \sigma_{k-1} \geq \sigma\,\varepsilon^{k-1}$, onde σ denota a expressão $\sigma_0 + \cdots + \sigma_{k-1} - (k-1)\varepsilon$. Então

$$\sigma_0 \cdots \sigma_{k-1}\sigma_k \geq (\sigma\sigma_k)\,\varepsilon^{k-1}\,.$$

Como $\sigma_i \geq \varepsilon$ para todo i, temos $\sigma \geq k\varepsilon - (k-1)\varepsilon = \varepsilon$ e portanto

$$\sigma\sigma_k \;=\; (\sigma + \sigma_k - \varepsilon)\varepsilon + (\sigma - \varepsilon)(\sigma_k - \varepsilon) \;\geq\; (\sigma + \sigma_k - \varepsilon)\,\varepsilon\,.$$

Segue daí que

$$\begin{aligned}(\sigma\sigma_k)\,\varepsilon^{k-1} &\geq (\sigma + \sigma_k - \varepsilon)\,\varepsilon^k\\ &= (\sigma_0 + \cdots + \sigma_{k-1} - (k-1)\varepsilon + \sigma_k - \varepsilon)\,\varepsilon^k\\ &= (\sigma_0 + \cdots + \sigma_k - k\varepsilon)\,\varepsilon^k\,.\end{aligned}$$

Isto encerra a demonstração do lema. Podemos agora aplica-lo à nossa situação concreta. Em virtude da maximalidade de π_h, temos $\rho_i \geq 1 - 1/m^2$ para todo i. Ademais, $\sum \rho_i = m + 1$ de acordo com (15.h). Diante disso, o lema (15.i) — com m, ρ_i e $1 - 1/m^2$ no lugar de k, σ_i e ε — garante que

$$\prod \rho_i \;\geq\; (1 + 1/m)\,(1 - 1/m^2)^m\,.$$

Como $\rho_h = 1 - 1/m^2 = (1 - 1/m)\,(1 + 1/m)$, temos a delimitação desejada. □

m	φ	$\lg \varphi$	ψ
2	9/8	$0.16992\cdots$	$0.055555\cdots$
3	256/243	$0.07518\cdots$	$0.031250\cdots$
4	16875/16384	$0.04259\cdots$	$0.020000\cdots$
5	1990656/1953125	$0.02745\cdots$	$0.013888\cdots$
6	367653125/362797056	$0.01918\cdots$	$0.010204\cdots$

$$\varphi = (1-\frac{1}{m^2})^m\,(1-\frac{1}{m})^{-1} \qquad \psi = \frac{1}{2(m+1)^2}$$

Figura 15.3: Ilustração das delimitações 15.9 e 15.10.

É importante estabelecer uma delimitação simples para o *logaritmo* do fator de contração. O logaritmo na base 2 será denotado por $\lg$. lg

Delimitação 15.10 (do logaritmo do fator de contração) *Em cada ocorrência do caso* 1.2B*,*

$$\lg \frac{\prod \rho_i}{\rho_h} \;>\; \frac{1}{2(m+1)^2}\,.$$

DEMONSTRAÇÃO: Se $\ln$ denota o logaritmo na base e então, para todo λ entre 0 e 1, ln

$$\ln(1-\lambda) = -\lambda - \lambda^2/2 - \lambda^3/3 - \lambda^4/4 - \cdots\,.$$

Observe agora que $1/m^2$ e $1/m$ estão entre 0 e 1, uma vez que $m \geq 2$. Portanto, $m \geq 2$
o logaritmo da delimitação 15.9 produz as desigualdades

$$\begin{aligned}
&m \ln \left(1-\frac{1}{m^2}\right) - \ln\left(1-\frac{1}{m}\right) \;\geq\\
&\quad\geq\; -\frac{1}{m} - \frac{1}{2m^3} - \frac{1}{3m^5} - \frac{1}{4m^7} - \cdots + \frac{1}{m} + \frac{1}{2m^2} + \frac{1}{3m^3} + \cdots\\
&\quad=\; \frac{1}{2m^2}\left(1-\frac{1}{m}\right) + \frac{1}{3m^3}\left(1-\frac{1}{m^2}\right) + \frac{1}{4m^4}\left(1-\frac{1}{m^3}\right) + \cdots\\
&\quad>\; \frac{1}{2m^2}\left(1-\frac{1}{m}\right)\\
&\quad=\; \frac{m-1}{2m^3}\\
&\quad>\; \frac{m-1}{2(m+1)(m+1)(m-1)}\\
&\quad=\; \frac{1}{2(m+1)^2}\,.
\end{aligned}$$

Como $\lg \kappa > \ln \kappa$ quando $\ln \kappa$ é positivo, temos a delimitação desejada. □

Delimitação 15.11 (do número de iterações) *O algoritmo executa não mais que*

$$2m(m+1)^2\,(4\lg\omega+\lg m+7)$$

iterações, onde $\omega=\omega(A)\omega(c)$.

DEMONSTRAÇÃO: No início da primeira iteração temos $\theta=(8\ddot{\omega}m)^m$. No fim de cada ocorrência do caso 1.2B, θ é substituído por $(\rho_h/\prod\rho_i)\,\theta$. No fim da j-ésima iteração teremos $\theta<(8\ddot{\omega}m)^mL^{-j}$, onde L é uma delimitaçao do fator de contração. Em virtude da delimitação 15.10,

$$\lg\theta\;<\;m\lg(8\ddot{\omega}m)-j\frac{1}{2(m+1)^2}$$

no início da $j+1$-ésima iteração. Quando

$$j\;\geq\;2(m+1)^2\,m(\lg\ddot{\omega}+\lg(8m\ddot{\omega}))\,,$$

teremos $\lg\theta<-m\lg\ddot{\omega}$ e portanto $\theta<\ddot{\omega}^{-m}$, desigualdade esta que caracteriza o caso 2 do algoritmo. Mas

$$\begin{aligned}\lg\ddot{\omega}+\lg(8\ddot{\omega}m) &= 2\lg\ddot{\omega}+\lg m+3\\ &= 2\lg(4\omega^2)+\lg m+3\\ &= 4\lg\omega+\lg m+7\,.\end{aligned}$$

Portanto, $2m(m+1)^2\,(4\lg\omega+\lg m+7)$ é uma delimitação superior para o número de iterações. □

A menos que o sistema A,c seja extremamente esparso (menos que $7+\lg m$ componentes não-nulos), teremos $\lg\omega\geq\lg m+7$, e portanto o número de iterações é limitado por

$$10\,(m+1)^3\;\lg\omega\,. \tag{15.i}$$

15.10 Número de operações aritméticas

Seja n uma abreviatura para $|N|$. Não é difícil verificar que o algoritmo executa não mais $2mn+m^2+m+2$ operações aritméticas para decidir que caso se aplica e não mais que $6m^2+6m+4$ operações aritméticas ao longo da execução do caso 1.2B. Assim, o número de operações aritméticas em cada iteração é limitado por $7m^2+2mn+7m+6$. Quando $m\geq2$ e $n\geq1$, esse número não passa de n

$$7m(m+n+1)\,.$$

A propósito, as operações aritméticas envolvem números racionais; mas é possível reorganizar o algoritmo de modo que ele opere apenas com inteiros.

Em virtude de (15.i), o número total de operações aritméticas durante a execução do algoritmo é limitado por

$$70\,(m+1)^4\,(m+n+1)\;\lg\omega$$

(a menos que o sistema seja extremamente esparso).

15.11 Conclusão

O algoritmo de Yamnitsky-Levin recebe um sistema inteiro A, c e devolve um ponto interior do poliedro $Y(A, c)$ se tal ponto existir. O número de operações aritméticas que o algoritmo executa no pior caso está na ordem de

$$m^4(m+n)\lg\omega\,,$$

onde ω é o produto de todos os números da forma $1+\alpha$ sendo α o valor absoluto de um componente do sistema $A\,, c$.

Infelizmente, nossa análise não leva em conta o custo de cada operação aritmética; esse custo é considerável, pois os números envolvidos podem ter numeradores e denominadores enormes. É possível reescrever o algoritmo, com a introdução de arredondamentos apropriados [GLS88, p.80] [GLS93] [Sch86, p.166], de modo que o consumo total de tempo — que leva em conta o custo de cada operação aritmética — ainda seja limitado por um polinômio em m, n e $\lg\omega$.

15.12 Apêndice: Uma só linha

Considere o problema do ponto interior no caso em que a matriz A tem uma só linha ($m = 1$). O algoritmo de Yamnitsky-Levin não se aplica diretamente, pois depende da desigualdade $m \geq 2$. Mas não é difícil escrever uma variante do algoritmo para esse caso. Nessa variante, os tetraedros são intervalos e o algoritmo nada mais é que uma busca binária.

Nosso algoritmo recebe vetores inteiros a e c sobre N e devolve um número racional t tal que $ta < c$ ou pára sem nada devolver se tal número não existe. Cada iteração começa com números racionais t_0, t_1 e θ. A primeira iteração começa com $t_0 = t_0^\star$, $t_1 = t_1^\star$ e $\theta = 8\ddot{\omega}$, onde $t_0^\star = -4\ddot{\omega}$, $t_1^\star = +4\ddot{\omega}$ e $\ddot{\omega} = 4\omega(a)^2\omega(c)^2$. Cada iteração consiste no seguinte:

CASO 1: $\theta \geq \ddot{\omega}^{-1}$.
Seja t o número $\frac{1}{2}t_0 + \frac{1}{2}t_1$.

CASO 1.1: $ta_{[k]} < c_{[k]}$ *para cada* k *em* N.
Devolva t e pare.

CASO 1.2: $ta_{[k]} \geq c_{[k]}$ *para algum* k *em* N.

CASO 1.2A: $a_{[k]} = 0$.
Pare sem nada devolver.

CASO 1.2B: $a_{[k]} \neq 0$.
Sejam π_0 e π_1 os números $(t - t_0)\,a_{[k]}$ e $(t - t_1)\,a_{[k]}$ respectivamente.
Defina t_0' e t_1' da seguinte maneira:
se $\pi_0 \geq 0$ então $t_0' = t_0$ e $t_1' = t$ senão $t_1' = t_1$ e $t_0' = t$.
Seja θ' o número $\theta/2$.
Comece nova iteração com t_0', t_1' e θ' nos papéis de t_0, t_1 e θ.

CASO 2: $\theta < \ddot{\omega}^{-1}$.
Pare sem nada devolver. □

O funcionamento do algoritmo pode ser descrito pelos seguintes invariantes: No início de cada iteração,

(i1) *se* $ya \le c$ *e* $t_0^\star \le y \le t_1^\star$ *então* $t_0 \le y \le t_1$,
(i2) $t_1 = t_0 + \theta$,
(i3) $\theta > 0$.

Para mostrar a validade do invariante (i1) basta verificar que no fim do caso 1.2B temos

$$\text{se } ya_{[k]} \le c_{[k]} \text{ e } t_0 \le y \le t_1 \text{ então } t_0' \le y \le t_1'. \tag{15.j}$$

A demonstração desse fato é um caso particular do que discutimos na seção 15.8. Suponha que y é um número tal que $t_0 \le y \le t_1$ e $ya_{[k]} \le c_{[k]}$. É claro que existe um número λ entre 0 e 1 tal que

$$\lambda t_0 + (1-\lambda)t_1 = y.$$

É preciso mostrar que existe λ' entre 0 e 1 tal que $y = \lambda' t_0' + (1-\lambda')t_1'$. Como estamos no caso 1.2, temos $ta_{[k]} \ge c_{[k]}$. Como $ya_{[k]} \le c_{[k]}$, vale um das alternativas:

$$t_0 a_{[k]} \le c_{[k]} \le t_1 a_{[k]} \quad \text{ou} \quad t_0 a_{[k]} \ge c_{[k]} \ge t_1 a_{[k]}.$$

Digamos que vale a primeira alternativa (a demonstração é análoga no outro caso). Como $ya_{[k]} \le c_{[k]}$, concluímos que y está mais próximo de t_0 que de t_1 e portanto $\lambda \ge 1/2$. Seja λ' o número $2\lambda - 1$ e observe que λ'

$$0 \le \lambda' \le 1.$$

Como $ta_{[k]} \ge c_{[k]} \ge t_0 a_{[k]}$, temos $\pi_0 \ge 0$ e portanto $t_0' = t_0$ e $t_1' = t$. É fácil verificar agora que

$$\lambda' t_0' + (1-\lambda')t_1' = \lambda t_0 + (1 - \lambda t_1) = y.$$

Isso conclui a demonstração de (15.j).

No caso 2, o poliedro $Y(a,c)$ é vazio, ou seja, não existe número y tal que $ya \le c$. A demonstração desse fato segue do teorema 15.4 do tetraedro interior, que poderia ser enunciado assim nesse caso especial:

> Se existe um número y tal que $ya < c$ então existem um inteiro δ entre 1 e $\ddot{\omega}$ e números inteiros z_0 e z_1 entre $-2\ddot{\omega}$ e $2\ddot{\omega}+1$ tais que
>
> $$z_0 + 1 \le z_1, \quad z_0 a \le \delta c \quad \text{e} \quad z_1 a \le \delta c,$$
>
> onde $\ddot{\omega}$ é o número $4\omega(a)^2\omega(c)^2$.

Exercícios

15.1 Verifique a proposição 15.1 (fórmula do volume) e a proposição 15.2 (do volume nulo) no caso $m = 1$ e no caso $m = 2$.

15.2 Dê uma prova elementar do teorema 15.4 (tetraedro interior) no caso $|M| = |N| = 1$. Enuncie o teorema 15.4 e o lema 15.5 no caso $|M| = 1$ (e N arbitrário); demonstre essas versões especializadas.

15.3 Suponha que $Y(A, c)$ tem um ponto interior. Mostre que $Y(A, c)$ contém um cubo suficientemente grande. Mais especificamente, mostre que existe um inteiro δ entre 1 e $2\ddot{\omega}$ e um vetor inteiro z^* tais que $|z^*| \leq 4\ddot{\omega}$ e

$$(z^* + v)A \;\leq\; \delta c$$

para todo vetor v sobre M com componentes em $\{-1, +1\}$, onde $\ddot{\omega} = 4\omega(A)^2\omega(c)^2$.

15.4 O poliedro $Y(A, c)$ é **limitado** (seja seções C.5 e D.5) se existe um número ψ tal que $|y| \leq \psi$ para todo y em $Y(A, c)$. Suponha que $Y(A, c)$ é limitado. Mostre que $Y(A, c) \subseteq [t_0^\star, \ldots, t_m^\star]$ no início da primeira iteração do algoritmo de Yamnitsky-Levin.

15.5 Suponha que $Y(A, c)$ não é vazio. Mostre que, no início da primeira iteração do algoritmo de Yamnitsky-Levin, $Y(A, c) \cap [t_0^\star, \ldots, t_m^\star]$ não é vazio.

15.6 Escreva e analise uma versão especializada do algoritmo de Yamnitsky-Levin para o caso $m = 2$.

15.7 Escreva o algoritmo de Yamnitsky-Levin em uma linguagem mais formal, mais próxima de PASCAL ou C. Escreva o algoritmo de modo que ele manipule somente números inteiros.

Parte V

Apêndices

Apêndice A

Simplex Dual

O algoritmo Simplex (secao 6.3) foi criado para resolver o problema canônico primal (seção 7.1). O algoritmo Simplex Dual, que esboçaremos neste apêndice, resolve o problema canônico dual. Para evitar confusão, passaremos a nos referir ao primeiro algoritmo como Simplex Primal.

Simplex Primal

O Simplex Primal procura, primeiro, estabelecer a desigualdade $E_{[\,,n]} \geq o$ e, depois, a desigualdade $E_{[m,\,]} \geq o$. Para fazer isso, examina uma nova linha da matriz em cada iteração. O Simplex Dual, ao contrário, examina uma nova *coluna* da matriz em cada iteração e procura, primeiro, estabelecer a desigualdade $E_{[m,\,]} \geq o$ e só depois a desigualdade $E_{[\,,n]} \geq o$. Ambos os algoritmos fazem pivotação "de linhas", isto é, em cada iteração somam a cada linha de E uma combinação linear apropriada das outras linhas.

A.1 Matrizes simples no sentido dual

O conceito de matriz dual-simples é semelhante ao conceito de matriz simples; para evitar confusão, usaremos a expressão *primal-simples* (veja seção 3.1) em referências ao último. Suponha que M e N são conjuntos de índices e que m e n são elementos de M e N respectivamente. Diremos que uma matriz E sobre $M \times N$ é **dual-simples** (ou **simples no sentido dual**) em relação ao par m, n de índices se for de um dos três tipos definidos a seguir.

primal-simples

dual-simples

Uma matriz E é **dual-simples solúvel** com relação ao par m, n se for primal-simples solúvel com relação ao par n, m, ou seja, se existem partes P de $M - m$ e Q de $N - n$ tais que

$$\begin{aligned} & E_{[P,Q]} \text{ é de bijeção}\,, \quad E_{[P,n]} \geq o\,, \\ & E_{[M-m-P,\,N]} = O\,, \\ E_{[m,\,N-n-Q]} \geq o\,, \quad & E_{[m,Q]} = o\,. \end{aligned}$$

O conjunto P é a **base de linhas** e o conjunto Q é uma **base de colunas** da matriz.

bases

Uma matriz E é **dual-simples inviável** com relação ao par m, n se existe

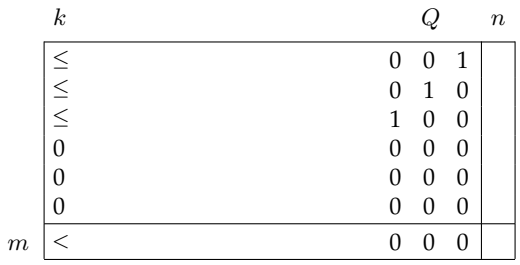

Figura A.1: Matriz dual-simples inviável.

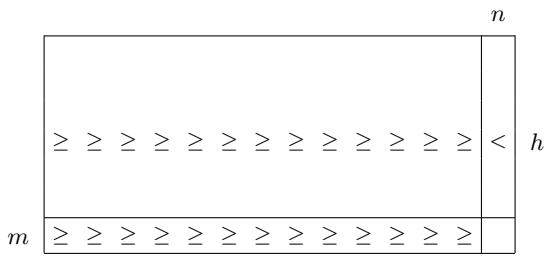

Figura A.2: Matriz dual-simples ilimitada.

uma parte P de $M - m$, uma parte Q de $N - n$ e um elemento k em $N - n - Q$ tais que

$$\begin{aligned} E_{[P,k]} &\leq o\,, \quad E_{[P,Q]} \text{ é de bijeção}\,,\\ E_{[M-m-P,k]} &= o\,, \quad E_{[M-m-P,Q]} = O\,,\\ E_{[m,k]} &< 0\,, \quad E_{[m,Q]} = o\,. \end{aligned}$$

Note que esta definição é semelhante — mas não idêntica — à de uma matriz primal-simples ilimitada. Os conjuntos P e Q são as **bases** da matriz e k é o índice de uma **coluna de inviabilidade**.

coluna de inviabilidade

Uma matriz E é **dual-simples ilimitada** com relação a m, n se $E_{[m,\,N]} \geq o$ e existe h em $M - m$ tal que

$$E_{[h,\,N-n]} \leq o \text{ e } E_{[h,n]} > 0 \quad \text{ou} \quad E_{[h,\,N-n]} \geq o \text{ e } E_{[h,n]} < 0\,.$$

Esta definição é semelhante — mas não idêntica — à de uma matriz primal-simples inviável. O elemento h de $M-m$ é o índice de uma **linha de ilimitação**.

linha de ilimitação

A.2 Esboço do Simplex Dual

Para fazer um primeiro esboço do Simplex Dual, suponha que E é uma matriz sobre $M \times N$ tal que

(i0) $\quad f_{[Q]} = o\,,\ f_{[N-n-Q]} \geq o\,,$

(i1) $E_{[P,Q]}$ é uma matriz de bijeção e $E_{[M-m-P,N-n]} = O$,

onde P é uma parte de $M-m$, Q é uma parte de $N-n$, e f é a linha m da matriz, ou seja, $f = E_{[m,\,]}$. Nosso objetivo é transformar E, por meio de sucessivas operações de pivotação, numa matriz que seja dual-simples com relação a m, n.

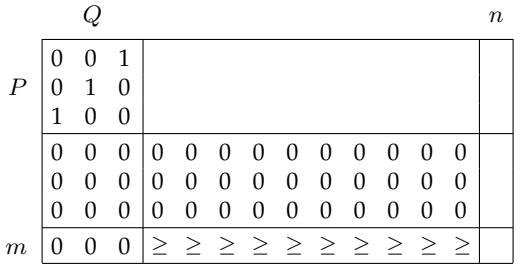

Figura A.3: Matriz E no início de uma iteração do Simplex Dual.

Se $E_{[M-m-P,n]} \neq o$ então E é dual-simples ilimitada. Se $E_{[M-m-P,n]}$ é nulo e $E_{[P,n]} \geq o$ então E é dual-simples solúvel. Suponha agora que $E_{[M-m-P,n]}$ é nulo mas $E_{[p,n]}$ é negativo para algum p em P. Para tornar a matriz "mais simples", o Simplex Dual considera a possibilidade de executar uma pivotação em torno de p, k, com k escolhido em $N-n-Q$ de modo que a validade dos invariantes (i0) e (i1) seja preservada. Uma pivotação em torno de p, k consiste em substituir a matriz E pela matriz E' definida pelas equações

$$E'_{[p,\,]} = \frac{1}{E_{[p,k]}}\, E_{[p,\,]} \quad \text{e} \quad E'_{[i,\,]} = E_{[i,\,]} - \frac{E_{[i,k]}}{E_{[p,k]}}\, E_{[p,\,]}$$

para cada i em $M-p$. Ao mesmo tempo, Q é substituído pelo conjunto $Q' = Q - q + k$, onde q é o único elemento de Q tal que $E_{[p,q]} = 1$. Para garantir que $E'_{[p,n]}$ não seja negativo, é preciso escolher p e k de modo que

$$E_{[p,k]} < 0\,. \tag{A.a}$$

É claro que depois da pivotação o invariante (i1) continuará valendo com E' e Q' no lugar de E e Q. Resta entender as condições sobre k que garantem a preservação de (i0). Se denotarmos por f' o vetor $E'_{[m,\,]}$ então é claro que

$$f' = f - \frac{f_{[k]}}{E_{[p,k]}}\, E_{[p,\,]}\,.$$

É fácil verificar que $f'_{[Q']}$ é nulo e portanto a primeira parte de (i0) vale. Resta considerar a segunda parte de (i0), supondo satisfeita a condição (A.a). É claro que $f'_{[q]} \geq f_{[q]}$. De modo mais geral, para cada j em $N-n$, temos $f'_{[j]} \geq f_{[j]}$ se $E_{[p,j]} \geq 0$ e $f'_{[j]} \leq f_{[j]}$ em caso contrário. Portanto, para garantir $f'_{[N-n]} \geq o$ é necessário e suficiente que k satisfaça a condição

$$f_{[j]} \;\geq\; \frac{f_{[k]}}{E_{[p,k]}}\, E_{[p,j]} \tag{A.b}$$

para todo j em $N - n$ tal que $E[p, j]$ é negativo. As condições (A.a) e (A.b) são o ponto de partida do Simplex Dual. Basta escolher p em P e k em $N - n$ que satisfaçam as condições e executar uma pivotação em torno de p, k. Se não existe k que satisfaça (A.a) então p é uma linha de ilimitação. A nova matriz E' não é necessariamente "mais simples" que E, uma vez que $E'[P, n]$ pode ter mais componentes negativos que $E[P, n]$. Mas o Simplex Dual espera, assim mesmo, estar fazendo algum progresso.

$$
\begin{array}{rrrrrrr}
2 & -1 & 1 & 1 & 0 & 0 & -5 \\
4 & 1 & 2 & 0 & 1 & 0 & 11 \\
3 & 4 & -5 & 0 & 0 & 1 & 8 \\
3 & 1 & 0 & 0 & 0 & 0 & 0
\end{array}
$$

$$
\begin{array}{rrrrrrr}
-2 & 1 & -1 & -1 & 0 & 0 & 5 \\
6 & 0 & 3 & 1 & 1 & 0 & 6 \\
11 & 0 & -1 & 4 & 0 & 1 & -12 \\
5 & 0 & 1 & 1 & 0 & 0 & -5
\end{array}
$$

$$
\begin{array}{rrrrrrr}
-13 & 1 & 0 & -5 & 0 & -1 & 17 \\
39 & 0 & 0 & 13 & 1 & 3 & -30 \\
-11 & 0 & 1 & -4 & 0 & -1 & 12 \\
16 & 0 & 0 & 5 & 0 & 1 & -17
\end{array}
$$

Figura A.4: Exemplo de aplicação do Simplex Dual. A figura registra o valor de E no início de sucessivas iterações. Na primeira iteração os elementos de Q são $4, 5, 6$ e os elementos de P são $1, 2, 3$. A última matriz é dual-simples ilimitada com relação à linha 4 e coluna 7.

A.3 Heurística Simplex Dual

Dada uma matriz D sobre $M \times N$ e elementos m e n de M e N respectivamente, o objetivo do Simplex Dual é transformar D, por meio de sucessivas operações de pivotação, numa matriz que seja dual-simples com relação a m, n. A tarefa de escrever o algoritmo completo ficará a cargo do leitor; cuidaremos aqui apenas da fase II, que formaliza o esboço feito na seção anterior.

Heurística Simplex Dual (fase II) *Recebe uma matriz D sobre $M \times N$, elementos m e n de M e N respectivamente e partes P_0 e Q_0 de $M - m$ e $N - n$ respectivamente tais que*

$$
\begin{aligned}
&D[m, Q_0] = o\,, \;\; D[m, N-n-Q_0] \geq o\,, \\
&D[P_0, Q_0] \text{ é uma matriz de bijeção e } D[M-m-P_0, N-n] = O\,;
\end{aligned}
$$

se convergir, *devolve matrizes F e G tais que $FG = I$, $G[\,, m] = I[\,, m]$ e a matriz GD é dual-simples (solúvel ou ilimitada) com relação ao par m, n.*

Cada iteração começa com uma parte Q de $N-n$, uma parte P de $M-m$ e matrizes F, G e E. A primeira iteração começa com $Q = Q_0$, $P = P_0$, $F = G = I$ e com $E = D$. Cada iteração consiste no seguinte, com $f = E_{[m,\,]}$:

CASO 0: $E_{[M-m-P,\,n]} \neq o$.
Devolva F e G e pare (E é dual-simples ilimitada).

CASO 1: $E_{[M-m-P,\,n]} = o$ e $E_{[p,\,n]} < 0$ para algum p em P .
Seja K^* o conjunto de todos os k em $N-n$ para os quais $E_{[p,\,k]} < 0$.

CASO 1A: K^* é vazio .
Devolva F e G e pare (E é dual-simples ilimitada).

CASO 1B: K^* não é vazio .
Escolha k em K^* de modo que $f_{[k]}/E_{[p,\,k]}$ seja máximo.
Seja F', G', E' o resultado da pivotação de F, G, E em torno de p, k.
Seja q um elemento de Q tal que $E_{[p,\,q]} = 1$.
Comece nova iteração com $Q - q + k$, F', G' e E'
nos papéis de Q, F, G e E.

CASO 2: $E_{[M-m-P,\,n]} = o$ e $E_{[P,\,n]} \geq o$.
Devolva F e G e pare (E é dual-simples solúvel). □

A operação de pivotação é definida exatamente como no Simplex Primal. O comportamento da heurística é descrita pelos seguintes

Invariantes *No início de cada iteração,*

(i0) $f_{[Q]} = o$ e $f_{[N-n-Q]} \geq o$,
(i1) $E_{[P,\,Q]}$ *é uma matriz de bijeção* e $E_{[M-m-P,\,N-n]} = O$,
(i2) $FG = I$ e $GD = E$,

onde f *é o vetor* $E_{[m,\,]}$. f

As demonstrações dos invariantes são análogas às correspondentes demonstração no Simplex Primal. A cada ocorrência do caso 1B temos

$$f'_{[n]} \;=\; f_{[n]} - \frac{f_{[k]}}{E_{[p,\,k]}}\, E_{[p,\,n]} \;\leq\; f_{[n]}\,,$$

sendo $f' = E'_{[m,\,]}$, pois $E_{[p,\,n]}$ e $E_{[p,\,k]}$ são negativos e $f_{[k]} \geq 0$. Portanto, numa f'
seqüência de ocorrências do caso 1B, a correspondente seqüência de valores de $f_{[j]}$ é monotônica. Se for estritamente monotônica, a seqüência será finita, pois os valores correspondentes da variável Q serão todos distintos. Infelizmente, a monotonia pode não ser estrita e a heurística pode não convergir.

$$
\begin{array}{rrrr|rrrr|rrrr}
1 & 0 & 0 & 0 & -6 & 1 & 2 & 4 & 1 & 0 & 0 & 14 \\
0 & 1 & 0 & 0 & 3 & -2 & -1 & -5 & 0 & 1 & 0 & -25 \\
0 & 0 & 1 & 0 & -2 & 1 & 0 & 2 & 0 & 0 & 1 & 14 \\
0 & 0 & 0 & 1 & 5 & 3 & 3 & 6 & 0 & 0 & 0 & 0 \\
\\
1 & 4/5 & 0 & 0 & -18/5 & -3/5 & 6/5 & 0 & 1 & 4/5 & 0 & -6 \\
0 & -1/5 & 0 & 0 & -3/5 & 2/5 & 1/5 & 1 & 0 & -1/5 & 0 & 5 \\
0 & 2/5 & 1 & 0 & -4/5 & 1/5 & -2/5 & 0 & 0 & 2/5 & 1 & 4 \\
0 & 6/5 & 0 & 1 & 43/5 & 3/5 & 9/5 & 0 & 0 & 6/5 & 0 & -30 \\
\\
-5/3 & -4/3 & 0 & 0 & 6 & 1 & -2 & 0 & -5/3 & -4/3 & 0 & 10 \\
2/3 & 1/3 & 0 & 0 & -3 & 0 & 1 & 1 & 2/3 & 1/3 & 0 & 1 \\
1/3 & 2/3 & 1 & 0 & -2 & 0 & 0 & 0 & 1/3 & 2/3 & 1 & 2 \\
1 & 2 & 0 & 1 & 5 & 0 & 3 & 0 & 1 & 2 & 0 & -36
\end{array}
$$

Figura A.5: Exemplo de aplicação da heurística Simplex Dual. A figura registra os valores de G e E no início de sucessivas iterações. No início da primeira iteração, os elementos de Q são $5, 6, 7$ e os elementos de P são $1, 2, 3$. A última matriz E é dual-simples solúvel com relação a $4, 8$.

A.4 Algoritmo Simplex Dual

Para transformar a heurística Simplex Dual num algoritmo basta usar uma versão apropriada da regra lexicográfica ou da regra de Bland (veja capítulo 5) a que recorremos no Simplex Primal.

Como no caso do Simplex Primal, podemos reformular a notação fazendo $A = D_{[M-m,\,N-n]}$, $c = D_{[m,\,N-n]}$ e $b = D_{[M-m,\,n]}$. Diremos que o terno A, c, b é um **sistema dual**. O sistema é **simples** se a correspondente matriz D for dual-simples. Nessa notação, o algoritmo poderia ser formulado assim:

sistema dual

> *Recebe um sistema dual A, c, b e devolve matrizes F e G e um vetor g tais que $FG = I$ e o sistema dual $GA\,, c + gA\,, Gb$ é simples.*

A.5 Problema canônico dual

De acordo com a seção 8.1, o problema canônico dual $\mathsf{CD}(A, c, b)$ consiste no seguinte: dado um sistema dual A, c, b sobre $M \times N$, encontrar um vetor y em $Y(A, c)$ que maximize yb. Como de hábito, $Y(A, c)$ é o conjunto de todos os vetores y para os quais $yA \le c$.

Sistemas duais simples. O problema canônico dual $\mathsf{CD}(A, c, b)$ pode ser resolvido por mera inspeção quando o sistema dual A, c, b é simples. Suponha inicialmente que o sistema dual A, c, b é simples solúvel e tem bases P e Q. O vetor $y = o$ está em $Y(A, c)$ pois $yA = o \le c$. Por outro lado, yb é máximo

pois $y'b \leq 0$ para todo y' em $Y(A,c)$. De fato,

$$y'b \;=\; y'_{[P]}b_{[P]} \;\leq\; 0$$

pois $b_{[M-P]} = o$, $b_{[P]} \geq o$ e $y'_{[P]} \leq 0$, uma vez que $y'_{[P]}A_{[P,Q]} = y'A_{[\,,Q]} \leq c_{[Q]} = o$. Concluímos assim que o vetor nulo é solução do problema $\mathsf{CD}(A,c,b)$.

Suponha agora que A,c,b é simples inviável com bases P e Q e coluna de inviabilidade k. Então $Y(A,c)$ é vazio pelo seguinte motivo. Suponha que y está em $Y(A,c)$. Então $y_{[P]}A_{[P,k]} = (yA)_{[k]} \leq c_{[k]} < 0$, donde

$$y_{[p]} > 0$$

para algum p em P. Se q é o elemento de Q para o qual $A_{[p,q]} = 1$, temos $y_{[p]}A_{[p,q]} = y_{[p]} > 0$. Isto contradiz nossa hipótese de que $yA \leq c$. Portanto, $\mathsf{CD}(A,c,b)$ é inviável.

Suponha, finalmente, que A,c,b é simples ilimitado com linha de ilimitação h. Suponha $A_{[h,\,]} \leq o$ e $b_{[h]}$ é positivo (o outro caso é análogo). Tome $y'_{[h]} = 1$ e $y'_{[M-h]}$ nulo. Então

$$y'A = y'_{[h]}A_{[h,\,]} \leq o \leq c \quad \text{e} \quad y'b > 0\,.$$

Para qualquer λ positivo, $\lambda y'$ está em $Y(A,c)$ e $\lambda y'b$ é tanto maior quanto maior for λ. Portanto, $\mathsf{CD}(A,c,b)$ é ilimitado.

Sistemas duais arbitrários. Suponha que A,c,b não é simples. Submeta A,c,b ao algoritmo Simplex Dual, que devolverá matrizes F e G e um vetor g tais que $FG = I$ e o sistema dual $GA, c + gA, Gb$ é simples. Considere a função que leva qualquer vetor y no vetor

$$(y + g)\,F\,.$$

Esta função é uma bijeção de $Y(A,c)$ em $Y(GA, c+gA)$. A inversa dessa bijeção leva qualquer elemento z de $Y(GA, c + gA)$ no elemento

$$zG - g$$

de $Y(A,c)$. A existência da bijeção mostra que $Y(A,c)$ é vazio se e só se $Y(GA, c + gA)$ é vazio. Portanto, $\mathsf{CD}(A,c,b)$ é inviável se e só se $\mathsf{CD}(GA, c + gA, Gb)$ é inviável.

Considere agora os valores das funções objetivo nos dois problemas. O valor de um elemento y de $Y(A,c)$ no problema $\mathsf{CD}(A,c,b)$ é yb, enquanto o correspondente valor de $(y + g)F$ no problema $\mathsf{CD}(GA, c + gA, Gb)$ é

$$((y + g)F)(Gb) = yb + gb\,.$$

Portanto, os valores de vetores correspondentes nos dois problemas diferem por gb, que é constante. Segue daí que (1) o primeiro problema é ilimitado se e só se o segundo é ilimitado e (2) o primeiro problema tem solução se e só se o segundo tem solução. Mais especificamente, se y é solução do primeiro problema então $(y + g)F$ é solução do segundo; e se z é solução do segundo problema então $zG - g$ é solução do primeiro.

Exercícios

A.1 Mostre que os três tipos de matriz dual-simples são disjuntos (assim, por exemplo, uma matriz dual-simples solúvel com relação a m, n não pode ser, ao mesmo tempo, dual-simples inviável).

A.2 Escreva uma versão completa da heurística Simplex Dual. A heurística deve receber uma matriz arbitrária D sobre $M \times N$ e elementos m e n de M e N respectivamente; se convergir, deve devolver matrizes F e G tais que $FG = I$, $G_{[\,,m]} = I_{[\,,m]}$ e a matriz GD é dual-simples (solúvel, inviável ou ilimitada) com relação ao par m, n.

A.3 Escreva e analise o algoritmo Simplex Dual com regra lexicográfica. Escreva e analise o algoritmo Simplex Dual com regra de Bland.

Apêndice B

Análise de sensibilidade

Como a solução de um problema canônico é afetada por pequenas variações da função objetivo? Pequenas variações de c não afetam a solução do problema $\mathsf{CP}(A, b, c)$. Analogamente, pequenas variações de b não afetam a solução do problema $\mathsf{CD}(A, c, b)$.

A primeira seção deste apêndice trata dessas questões no caso em que apenas um dos componentes de c ou b varia. A segunda seção procura descrever, de maneira mais global e abstrata, a natureza da dependência entre o valor ótimo dos problemas e os parâmetros b e c. Este apêndice só depende dos capítulos 4, 6, 7 e 8.

B.1 Variação de um só componente

Seja A, b, c um sistema sobre $M \times N$ e suponha que existem matrizes F e G e um vetor y tais que $FG = I$ e o sistema $GA, Gb, c - yA$ é simples solúvel e tem bases P e Q. Portanto,

$$\begin{array}{rcl} & (GA)_{[P,\,Q]} \text{ é de bijeção}\,, & (Gb)_{[P]} \geq o\,, \\ (GA)_{[M-P,\,N-Q]} = O\,, & (GA)_{[M-P,\,Q]} = O\,, & (Gb)_{[M-P]} = o\,, \\ (c - yA)_{[N-Q]} \geq o\,, & (c - yA)_{[Q]} = o\,. & \end{array}$$

É claro que y é solução do problema $\mathsf{CD}(A, c, b)$. É claro que o vetor básico x do sistema $GA, Gb, c - yA$ é solução do problema $\mathsf{CP}(A, b, c)$. Ademais, $cx = yb$. Diremos que esse número é o **valor ótimo** do par de problemas.

Suponha agora que b é substituído por um vetor $\check{b}$, ligeiramente diferente. Quão próximo $\check{b}$ deve estar de b para que o sistema $GA, G\check{b}, c - yA$ seja simples solúvel e tenha a mesma base Q de colunas?

Suponha, em seguida, que c é substituído por um vetor ligeiramente diferente $\check{c}$. Quão próximo $\check{c}$ deve ser de c para que o sistema $GA, Gb, \check{c} - \check{y}A$ seja simples solúvel, com base de colunas Q, para algum $\check{y}$?

Variação de um componente de b

Suponha que $\check{b}$ difere de b em apenas um componente: $\check{b}_{[m]} = b_{[m]} + \beta$ para algum m em M e $\check{b}_{[i]} = b_{[i]}$ para todo i distinto de m.

Fato *Se $G_{[M-P,\,m]} = o$ e β respeita as restrições a seguir então o sistema $GA, G\check{b}, c - yA$ é simples solúvel e tem base de colunas Q. As restrições são*

$$\frac{-G_{[p,\,]}\,b}{G_{[p,\,m]}} \;\leq\; \beta \;\leq\; \frac{-G_{[p,\,]}\,b}{G_{[p,\,m]}}\,,$$

onde a desigualdade esquerda se aplica a todo p em P tal que $G_{[p,\,m]}$ é positivo e a desigualdade direita se aplica a todo p em P tal que $G_{[p,\,m]}$ é negativo.

DEMONSTRAÇÃO: É óbvio que $G\check{b} = Gb + \beta G_{[\,,\,m]}$. Como $G_{[M-P,\,m]}$ é nulo, temos $(G\check{b})_{[M-P]} = (Gb)_{[M-P]} = o$. Como β satisfaz as desigualdades, temos

$$\beta\, G_{[p,\,m]} \;\geq\; -G_{[p,\,]}\,b$$

para todo p em P, donde $(G\check{b})_{[P]} \geq o$. Portanto, o sistema $GA, G\check{b}, c - yA$ é simples solúvel e tem base de colunas Q. □

Eis algumas observações óbvias. 1. O lado esquerdo das restrições sobre β pode ser vazio; o lado direito também pode ser vazio. 2. Nas restrições sobre β, a expressão à esquerda tem valor não-positivo e a expressão à direita tem valor não-negativo. As restrições estão certamente satisfeitas quando $\beta = 0$. 3. A condição $G_{[M-P,\,m]} = o$ significa, em particular, que m deve estar em P. De fato, se G foi gerada pelo algoritmo Simplex então $G_{[M-P,\,m]} = I_{[M-P,\,m]} \neq o$ quando m está em $M - P$.

Respeitadas as condições acima, é óbvio que y é solução do novo problema $\mathsf{CD}(A, c, \check{b})$. O valor ótimo do problema, $y\check{b}$, varia linearmente com β:

$$y\check{b} = yb + \beta y_{[m]}\,.$$

Quanto ao problema $\mathsf{CP}(A, \check{b}, c)$, este terá por solução o vetor básico, digamos $\check{x}$, do sistema $GA, G\check{b}, c - yA$. É claro que $c\check{x} = y\check{b}$.

Variação de c fora da base

Suponha que $\check{c}$ difere de c em apenas um elemento de $N - Q$, ou seja, suponha que $\check{c}_{[n]} = c_{[n]} + \gamma$ para algum n em $N - Q$ e $\check{c}_{[j]} = c_{[j]}$ para j distinto de n.

Fato *Se $\gamma \geq (yA - c)_{[n]}$ então o sistema $GA, Gb, \check{c} - yA$ é simples solúvel com base de colunas Q.*

DEMONSTRAÇÃO: Como $\check{c}_{[Q]} = c_{[Q]}$, temos $(yA)_{[Q]} = \check{c}_{[Q]}$. Ademais, $(yA)_{[j]} \leq \check{c}_{[j]}$ para todo j distinto de n. Finalmente, $(yA)_{[n]} \leq c_{[n]} + \gamma = \check{c}_{[n]}$. Logo, o sistema $GA, Gb, \check{c} - yA$ é simples solúvel e tem base de colunas Q. □

Respeitada a restrição sobre γ, é óbvio que x é solução do novo problema $\mathsf{CP}(A, b, \check{c})$ e que $\check{c}x = cx$. Quanto ao problema dual, é claro que y é solução de $\mathsf{CD}(A, \check{c}, b)$ e que $yb = \check{c}x$.

Variação de c na base

Suponha que $\check{c}$ só difere de c em um elemento de Q. Mais especificamente, suponha que $\check{c}_{[n]} = c_{[n]} + \gamma$ para algum n em Q e $\check{c}_{[j]} = c_{[j]}$ para todo j distinto de n. Seja m o único elemento de P tal que $(GA)_{[m,\,n]} = 1$. m m

Fato *Se γ respeita as restrições a seguir então existe um vetor $\check{y}$ tal que o sistema $GA, Gb, \check{c} - \check{y}A$ é simples solúvel com base de colunas Q. As restrições são*

$$\frac{(c - yA)_{[k]}}{(GA)_{[m,\,k]}} \;\leq\; \gamma \;\leq\; \frac{(c - yA)_{[k]}}{(GA)_{[m,\,k]}}\,,$$

onde a desigualdade esquerda se aplica a todo k em $N - Q$ tal que $(GA)_{[m,\,k]}$ é negativo e a desigualdade direita se aplica a todo k em $N - Q$ tal que $(GA)_{[m,\,k]}$ é positivo.

DEMONSTRAÇÃO: Adote a abreviatura $E = GA$. Seja $\check{y}$ o vetor $y + \gamma G_{[m,\,]}$. É claro que

$$\check{y}A = yA + \gamma E_{[r,\,]}\,. \tag{B.a}$$

Se restringirmos (B.a) a Q teremos $(\check{y}A)_{[Q]} = (yA)_{[Q]} + \gamma E_{[m,\,Q]} = c_{[Q]} + \gamma I_{[n,\,Q]}$. Logo,

$$(\check{y}A)_{[Q]} = \check{c}_{[Q]}\,. \tag{B.b}$$

Se restringirmos (B.a) a um elemento k de $N - Q$ teremos

$$(\check{y}A)_{[k]} = (yA)_{[k]} + \gamma E_{[m,\,k]}\,.$$

Se γ satisfaz as desigualdade expostas no enunciado então $\gamma E_{[m,\,k]} \leq c_{[k]} - (yA)_{[k]}$ e portanto $(\check{y}A)_{[k]} \leq c_{[k]}$. Como $c_{[N-Q]} = \check{c}_{[N-Q]}$, temos

$$(\check{y}A)_{[N-Q]} \leq \check{c}_{[N-Q]}\,.$$

Segue daí e de (B.b) que o sistema $GA, Gb, \check{c} - \check{y}A$ é simples e tem base de colunas Q. □

É óbvio que o lado esquerdo das restrições sobre γ pode ser vazio; o lado direito também pode ser vazio. É claro também que as restrições estão satisfeitas quando $\gamma = 0$.

Satisfeitas as restrições sobre γ, o vetor básico x do sistema $GA, Gb, c - yA$ também é vetor básico do sistema $GA, Gb, \check{c} - \check{y}A$. Ademais,

$$\check{c}x = cx + \gamma x_{[n]}\,,$$

donde $\check{c}x$ varia linearmente com γ. Quanto ao problema dual, é claro que $\check{y}$ é solução de $\mathsf{CD}(A, \check{c}, b)$ e que $\check{y}b = \check{c}x$.

B.2 Exemplo

Retomemos o exemplo discutido na seção 7.5 do capítulo 7. Para conveniência do leitor, vamos repetir a descrição do problema. Imagine que uma empresa fabrica quatro modelos de um produto. Cada produto passa por dois estágios de fabricação. O primeiro estágio dispõe de não mais que 600 homens-hora e o segundo de não mais que 400 homens-hora. Digamos que o número de homens-hora necessários, em cada estágio, para a fabricação de cada modelo do produto impõe as seguintes restrições:

$$\begin{aligned} 4x_1 + 9x_2 + 7x_3 + 10x_4 &\leq 600\,, \\ 1x_1 + 1x_2 + 3x_3 + 40x_4 &\leq 400\,, \end{aligned}$$

onde x_i é o número de unidades do modelo i. Queremos planejar a produção de modo a maximizar o lucro total, que é dado por $12x_1 + 20x_2 + 18x_3 + 40x_4$.

Nosso problema equivale ao problema canônico primal $\mathsf{CP}(A, b, c)$ com A, b e c descritos na figura B.1. Seja G a matriz e y o vetor descritos na figura B.2.

$$\begin{array}{rrrrrrr} 4 & 9 & 7 & 10 & 1 & 0 & 600 \\ 1 & 1 & 3 & 40 & 0 & 1 & 400 \\ -12 & -20 & -18 & -40 & 0 & 0 & \end{array}$$

Figura B.1: Sistema A, b, c.

$$\begin{array}{rr} 4/15 & -1/15 \\ -1/150 & 4/150 \\ -44/15 & -4/15 \end{array}$$

Figura B.2: Matriz G e vetor y.

$$\begin{array}{rrrrrrr} 1 & 35/15 & 25/15 & 0 & 4/15 & -1/15 & 400/3 \\ 0 & -5/150 & 5/150 & 1 & -1/150 & 4/150 & 20/3 \\ 0 & 20/3 & 10/3 & 0 & 44/15 & 4/15 & \end{array}$$

Figura B.3: Sistema $GA, Gb, c - yA$.

$$\begin{array}{rrrrrrr} 4 & 9 & 7 & 10 & 1 & 0 & \beta + 600 \\ 1 & 1 & 3 & 40 & 0 & 1 & 400 \\ -12 & -20 & -18 & -40 & 0 & 0 & \end{array}$$

Figura B.4: Sistema $A, \check{b}, c$.

$$\begin{array}{rrrrrrr} 1 & 35/15 & 25/15 & 0 & 4/15 & -1/15 & 4\beta/15 + 400/3 \\ 0 & -5/150 & 5/150 & 1 & -1/150 & 4/150 & n\beta/150 + 20/3 \\ 0 & 20/3 & 10/3 & 0 & 44/15 & 4/15 & \end{array}$$

Figura B.5: Sistema $GA, G\check{b}, c - yA$.

O sistema $GA, Gb, c - yA$ (figura B.3) é simples solúvel; sua base é composta pelas colunas 1 e 4. Portanto, um plano de produção ótimo requer a fabricação de apenas dois modelos:

$$x_1 = \tfrac{1}{3}\,400\,, \quad x_2 = x_3 = 0\,, \quad x_4 = \tfrac{1}{3}\,20\,, \quad x_5 = x_6 = 0\,. \tag{B.c}$$

Com este plano, teremos $cx = -\frac{1}{3}\,5600$.

Suponha agora que a disponibilidade de mão de obra no primeiro estágio é alterada para $600 + \beta$. Nosso problema passa a ser definido pelo sistema $A, \check{b}, c$ descrito na figura B.4.

Não é difícil verificar que se $-500 \leq \beta \leq 1000$ então o sistema $GA, G\check{b}, c - yA$ (figura B.5) é simples solúvel e a base continua composta pelas colunas 1 e 4. O plano de produção ótimo continua limitado aos modelos 1 e 4:

$$\begin{aligned} \check{x}_1 &= \tfrac{1}{3}\,400 + \tfrac{1}{15}\,4\beta\,, & \check{x}_2 &= \check{x}_3 = 0\,, \\ \check{x}_4 &= \tfrac{1}{3}\,20 - \tfrac{1}{150}\,\beta\,, & \check{x}_5 &= \check{x}_6 = 0\,. \end{aligned}$$

Com este plano de produção teremos $c\check{x} = -\frac{1}{3}\,5600 - \frac{1}{15}\,44\,\beta$.

Suponha a seguir que b permanece constante mas c é substituído por um vetor $\check{c}$. Digamos que $\check{c}_2 = c_2 + \gamma$ e $\check{c}$ coincide com c em todos os componentes. É fácil verificar que se $\gamma \geq -\frac{1}{3}\,20$ então o sistema $GA, Gb, \check{c} - yA$ é simples solúvel. O plano de produção x descrito em (B.c) continua ótimo e e $\check{c}x$ continua valendo $-\frac{1}{3}\,5600$.

Suponha, finalmente, que $\check{c}_1 = c_1 + \gamma$ e $\check{c}$ coincide com c em todos os demais componentes (figura B.6). Seja $\check{y}$ o vetor especificado na figura B.7. É claro que o sistema $GA, Gb, \check{c} - \check{y}A$ (figura B.8) é simples solúvel desde que $-4 \leq \gamma \leq 2$. O vetor x descrito em (B.c) continua sendo um plano de produção ótimo e $\check{c}x = -\frac{1}{3}\,5600 + \frac{1}{3}\,400\,\gamma$.

$$\begin{array}{rrrrrrr} 4 & 9 & 7 & 10 & 1 & 0 & 600 \\ 1 & 1 & 3 & 40 & 0 & 1 & 400 \\ \gamma - 12 & -20 & -18 & -40 & 0 & 0 & \end{array}$$

Figura B.6: Sistema $A, b, \check{c}$.

$$\begin{array}{rr} 4/15 & -1/15 \\ -1/150 & 4/150 \\ 4\gamma/15 - 44/15 & -\gamma/15 - 4/15 \end{array}$$

Figura B.7: Matriz G e vetor $\check{y}$.

$$\begin{array}{rrrrrrr} 1 & 35/15 & 25/15 & 0 & 4/15 & -1/15 & 400/3 \\ 0 & -5/150 & 5/150 & 1 & -1/150 & 4/150 & 20/3 \\ 0 & -7\gamma/3 + 20/3 & -5\gamma/3 + 10/3 & 0 & -4\gamma/15 + 44/15 & \gamma/15 + 4/15 & \end{array}$$

Figura B.8: Sistema $GA, Gb, \check{c} - \check{y}A$.

B.3 Valor ótimo como função de b e c

Considere os problemas canônicos $\mathsf{CP}(A, b, c)$ e $\mathsf{CD}(A, c, b)$. Se o primeiro problema tem solução então o seu **valor ótimo** é o número $\min_x cx$ para x variando em $X(A, b)$. Analogamente, se o segundo problema tem solução então o seu **valor ótimo** é o número $\max_y yb$ com y variando em $Y(A, c)$. De acordo com o teorema da dualidade 8.5, se um dos problemas tem solução então ambos têm solução e seus valores ótimos coincidem. O valor ótimo comum será denotado por

valor ótimo

$$\varphi(b, c)\,.$$

Diremos que esse é o valor ótimo **do sistema** A, b, c. Como varia $\varphi(b, c)$ em função de b e de c, supondo A fixa?

É preciso começar por esclarecer o *domínio* da função φ. Seja B o conjunto de todos os vetores b para os quais $X(A, b)$ não é vazio, isto é, o conjunto de todos os vetores b para os quais existe $x \geq o$ tal que $Ax = b$. (Há quem diga que B é o **cone de** A.)

B

Analogamente, seja C o conjunto de todos os vetores c para os quais $Y(A, c)$ não é vazio, isto é, o conjunto de todos os vetores c para os quais existe y tal que $yA \leq c$. De acordo com o teorema da dualidade 8.5, os problemas $\mathsf{CP}(A, b, c)$ e $\mathsf{CD}(A, c, b)$ têm solução se e só se b está em B e c está em C. Portanto, o domínio de φ é o produto cartesiano

C

$$B \times C\,.$$

conjunto convexo

É fácil verificar que o conjunto B é **convexo**, ou seja, que para todo par b_1, b_2 de elementos de B e todo par λ_1, λ_2 de números não-negativos, se $\lambda_1 + \lambda_2 = 1$ então $\lambda_1 b_1 + \lambda_2 b_2$ está em B. Eis a demonstração desse fato. Suponha que x_1 um vetor em $X(A, b_1)$ e x_2 um vetor em $X(A, b_2)$. Para qualquer par λ_1, λ_2 com as propriedades acima, seja b' o vetor $\lambda_1 b_1 + \lambda_2 b_2$ e x' o vetor $\lambda_1 x_1 + \lambda_2 x_2$. Então $x' \geq o$ e $Ax' = \lambda_1 Ax_1 + \lambda_2 Ax_2 = \lambda_1 b_1 + \lambda_2 b_2 = b'$, donde x' está em $X(A, b')$. A existência de um tal x' mostra que b' está em B.

É igualmente fácil verificar que o conjunto C é convexo, ou seja, que para todo par c_1, c_2 de elementos de C e todo par λ_1, λ_2 de números não-negativos, se $\lambda_1 + \lambda_2 = 1$ então $\lambda_1 c_1 + \lambda_2 c_2$ está em C.

Mostraremos a seguir que, quando o segundo parâmetro está fixo, a função φ é convexa, contínua e linear por trechos.

Fato B.1 (φ é convexa no primeiro parâmetro) *Seja c um elemento de C. Para todo par b_1, b_2 de elementos de B e todo par λ_1, λ_2 de números não-negativos tais que $\lambda_1 + \lambda_2 = 1$ tem-se*

$$\varphi(\lambda_1 b_1 + \lambda_2 b_2\,, c) \;\leq\; \lambda_1\, \varphi(b_1, c) \;+\; \lambda_2\, \varphi(b_2, c)\,.$$

DEMONSTRAÇÃO: Seja x_1 um vetor em $X(A, b_1)$ tal que $\varphi(b_1, c) = cx_1$. Seja x_2 um vetor em $X(A, b_2)$ tal que $\varphi(b_2, c) = cx_2$. Para todo par λ_1, λ_2 com as propriedades definidas no enunciado, seja b' o vetor $\lambda_1 b_1 + \lambda_2 b_2$ e x' o vetor $\lambda_1 x_1 + \lambda_2 x_2$. Então x' está em $X(A, b')$ e b' está em B (como mostramos acima) e portanto $\varphi(b', c) \leq cx' = \lambda_1 cx_1 + \lambda_2 cx_2 = \lambda_1 \varphi(b_1, c) + \lambda_2 \varphi(b_2, c)$. □

Por definição, $\varphi(b, c) = \max_y yb$ para y variando em $Y(A, c)$. Mas é possível dizer mais: existe uma parte *finita* Y' de $Y(A, c)$ tal que $\varphi(b, c) = \max_y yb$ para y variando em Y'.

Fato B.2 (φ é linear por trechos e contínua no primeiro parâmetro) *Para todo c em C existem vetores $y_1, y_2, \ldots, y_k$ sobre M tais que*

$$\varphi(b, c) \;=\; \max\{\, y_1 b, y_2 b, \ldots, y_k b \,\}$$

para todo b em B.

DEMONSTRAÇÃO: Seja c um elemento fixo de C. Seja $\mathcal{Q}$ conjunto das partes Q de N dotadas da seguinte propriedade: existem uma matriz inversível G e um vetor y tais que o sistema $GA, Gb, c - yA$ é simples solúvel com base de colunas Q para algum b em B. É óbvio que $\mathcal{Q}$ é finito.

Para cada Q em $\mathcal{Q}$, escolha um par G_Q, y_Q de modo que, para algum b, o sistema $G_Q A, G_Q b, c - y_Q A$ seja simples solúvel com base de colunas Q. Em particular, y_Q está em $Y(A, c)$ para todo Q em $\mathcal{Q}$. O conjunto de todos os vetores da forma y_Q com Q em $\mathcal{Q}$ é o que estamos procurando, como mostraremos a seguir.

Seja b um vetor em B e seja x_b uma solução do problema $\mathsf{CP}(A, b, c)$. É claro que $\varphi(b, c) = cx_b$. Vamos mostrar que

$$cx_b = \max_Q \; y_Q b\,,$$

onde o máximo é tomado sobre todo Q em $\mathcal{Q}$. Para todo Q em $\mathcal{Q}$, y_Q está em $Y(A, c)$ e portanto o lema da dualidade 8.1 garante que $cx_b \geq y_Q b$. Resta mostrar que $cx_b = y_Q b$ para algum Q em $\mathcal{Q}$.

Como c está em C, e portanto $Y(A, c)$ não é vazio, a análise do algoritmo Simplex mostra que existe uma parte Q de N, uma matriz inversível G e um vetor y tais que o sistema $GA, Gb, c - yA$ é simples solúvel com base de colunas Q. Logo, Q está em $\mathcal{Q}$. Seja x o vetor básico do sistema simples solúvel $GA, Gb, c - yA$, isto é, $x_{[N-Q]}$ é nulo e $(GA)x = Gb$. Observe que $cx = yb$ e que $cx = cx_b$. Mas x também é o vetor básico do sistema $G_Q A, G_Q b, c - yA$, pois

$$x_{[N-Q]} = o \quad \text{e} \quad G_Q A x = G_Q b\,.$$

Segue que $(c - y_Q A)x = 0$, donde $cx = y_Q b$. Portanto $cx_b = cx = y_Q b$, como queríamos demonstrar. □

Portanto, para descrever $\varphi(b, c)$ completamente como função de b basta calcular os vetores $y_1, \ldots, y_k$. Infelizmente, isto não é prático pois k cresce exponencialmente com $|N|$.

Demonstra-se analogamente que, como função do segundo parâmetro, φ é côncava, contínua e linear por trechos.

Fato B.3 (φ é côncava no segundo parâmetro) *Seja b um elemento de B. Para todo par c_1, c_2 de elementos de C e todo par λ_1, λ_2 de números não-negativos tais que $\lambda_1 + \lambda_2 = 1$ tem-se*

$$\varphi(b, \lambda_1 c_1 + \lambda_2 c_2) \;\geq\; \lambda_1\, \varphi(b, c_1) \;+\; \lambda_2\, \varphi(b, c_2)\,.$$

Fato B.4 (φ é linear por trechos e contínua no segundo parâmetro) *Para todo b em B existem vetores $x_1, x_2, \ldots, x_k$ sobre N tais que*

$$\varphi(b, c) \;=\; \min\{\, cx_1, cx_2, \ldots, cx_k \,\}$$

para todo c em C.

B.4 Conclusão

Para qualquer matriz fixa A, os problemas canônicos $\mathsf{CP}(A, b, c)$ e $\mathsf{CD}(A, c, b)$ têm solução se e só se o par b, c está em $B \times C$, onde B é o conjunto dos vetores b para os quais $X(A, b)$ não é vazio e C é o conjunto dos vetores c para os quais $Y(A, c)$ não é vazio.

Para b fixo em B, a função que a cada c em C associa o valor ótimo $\varphi(b, c)$ do sistema A, b, c é contínua, linear por trechos, e côncava. Para c fixo em C, a função que a cada b em B associa o valor ótimo $\varphi(b, c)$ do sistema A, b, c é contínua, linear por trechos, e convexa.

A solução x do problema $\mathsf{CP}(A, b, c)$ não é afetada por variações suficientemente pequenas de c. Analogamente, a solução y do problema $\mathsf{CD}(A, c, b)$ não se altera se b sofre variações suficientemente pequenas.

Apêndice C

Poliedro canônico primal

Este apêndice estuda algumas propriedades geométricas do poliedro canônico primal

$$X(A, b)\,.$$

Esse é o conjunto de todos os vetores $x \geq o$ tais que $Ax = b$. Suporemos, ao longo do apêndice, que A é uma matriz sobre $M \times N$ e que b é um vetor sobre M.

C.1 Dependência linear

Comecemos por examinar, em uma linguagem adequada ao nosso contexto, os conceitos de dependência e independência linear.

matriz de injeção

Uma matriz J sobre $M \times K$ é **de injeção** se existe uma parte P de M tal que $J_{[P,\,K]}$ é uma matriz de bijeção e $J_{[M-P,\,K]} = O$. Por exemplo, se E é uma matriz escalonada com base de colunas Q então $E_{[\,,\,Q]}$ é uma matriz de injeção.

Proposição C.1 *Para qualquer parte K de N, vale uma e apenas uma das seguintes alternativas:*

(1) existe uma matriz G tal que $GA_{[\,,\,K]}$ é de injeção;
(2) existe um vetor u tal que $Au = o$, $u_{[K]} \neq o$ e $u_{[N-K]} = o$.

DEMONSTRAÇÃO: De acordo com o algoritmo de Gauss-Jordan aplicado à matriz $A_{[\,,\,K]}$, existem matrizes F e G tais que $FG = I$ e $GA_{[\,,\,K]}$ é escalonada. Digamos que a base de colunas da matriz escalonada é Q. É claro que $GA_{[\,,\,Q]}$ é uma matriz de injeção. Se $Q = K$ então vale a alternativa *(1)*. Suponha agora que $Q \subset K$ e seja k um elemento de $K - Q$. É evidente que existe um vetor u sobre N tal que

$$u_{[k]} = 1\,, \quad u_{[N-Q-k]} = o \quad \text{e} \quad GAu = o\,.$$

É claro que $u_{[N-K]} = o$, $u_{[K]} \neq o$ e $Au = FGAu = o$. Portanto, vale a alternativa *(2)*.

Resta mostrar que as duas alternativas não podem ser simultaneamente verdadeiras. Suponha pois que vale *(2)*. Como $u_{[N-K]}$ é nulo,

$$GA_{[\,,K]}\,u_{[K]} \;=\; GAu \;=\; o\,,$$

uma vez que $Au = o$. Mas $u_{[K]}$ não é nulo, e portanto $GA_{[\,,K]}$ não pode ser uma matriz de injeção. Portanto, a alternativa *(1)* não vale. □

Na linguagem da álgebra linear, a alternativa *(1)* afirma que o conjunto de colunas da matriz $A_{[\,,K]}$ é linearmente independente enquanto *(2)* afirma que o conjunto de colunas da matriz $A_{[\,,K]}$ é linearmente dependente. No presente contexto, convém atribuir as propriedades de dependência e independência linear aos subconjuntos de N. Diremos, pois, que K é **linearmente independente em** A se vale a alternativa *(1)* e **linearmente dependente em** A se vale a alternativa *(2)*. A proposição C.1 mostra que as duas propriedades são complementares.

Usaremos as abreviaturas **li** e **ld** para as expressões "linearmente independente" e "linearmente dependente". É óbvio que todo subconjunto de um conjunto li também é li e que todo superconjunto de um conjunto ld também é ld. li ld

A demonstração da proposição C.1 pode ser facilmente convertida em um algoritmo que, ao receber A e K, decide se K é li ou ld. No primeiro caso, o algoritmo devolve uma matriz G para atestar que K é li; no segundo caso, o algoritmo devolve um vetor u para atestar que K é ld. algoritmo

C.2 Combinações convexas

Suponha que $x_1, \ldots, x_k$ são vetores sobre N. Uma **combinação afim** desses vetores é qualquer combinação linear $\lambda_1 x_1 + \cdots + \lambda_k x_k$ tal que

$$\lambda_1 + \cdots + \lambda_k \;=\; 1\,.$$

(Em termos geométricos, um vetor é combinação afim de x_1 e x_2 se estiver "na reta que passa pelos pontos x_1 e x_2".) Uma **combinação convexa** dos vetores $x_1, \ldots, x_k$ é qualquer combinação afim $\lambda_1 x_1 + \cdots + \lambda_k x_k$ tal que

$$\lambda_i \;\geq\; 0$$

para todo i. (Em termos geométricos, um vetor é combinação convexa de x_1 e x_2 se estiver "no segmento de reta que une os pontos x_1 e x_2".) O poliedro canônico primal é um conjunto convexo no seguinte sentido:

Proposição C.2 *Para qualquer parte finita Z de $X(A,b)$, toda combinação convexa de elementos de Z está em $X(A,b)$.*

DEMONSTRAÇÃO: Digamos que os elementos de Z são $z_1, \ldots, z_k$ e considere uma combinação convexa $\sum \lambda_i z_i$. Como $Az_i = b$ para todo i,

$$A(\textstyle\sum \lambda_i z_i) \;=\; \sum \lambda_i A z_i \;=\; \sum \lambda_i b \;=\; (\sum \lambda_i)\, b \;=\; b\,,$$

uma vez que $\sum \lambda_i = 1$. Por outro lado, $\sum \lambda_i z_i \geq 0$, uma vez que $\lambda_i \geq 0$ e $z_i \geq o$ para todo i. Portanto, $\sum \lambda_i z_i$ está em $X(A, b)$. □

A **envoltória convexa** de um conjunto finito Z de vetores é o conjunto de todas as combinações convexas de elementos de Z. A envoltória convexa de Z será denotada por $[Z]$. A proposição C.2 mostra que $[Z] \subseteq X(A, b)$ para toda parte finita Z de $X(A, b)$. Em certas circunstâncias, $X(A, b) = [Z]$ para um certo conjunto finito Z. A caracterização de um tal Z é o principal objetivo deste apêndice. $[Z]$

C.3 Vértices

A **folga**, ou **suporte**, de um elemento x de $X(A, b)$ é o conjunto de todos os índices j em N para os quais $x_{[j]} > 0$. A folga de x será denotada por folga suporte

$$S(x)\,.$$

Um vetor x em $X(A, b)$ é **básico** se sua folga $S(x)$ é minimal, ou seja, se não existe x' em $X(A, b)$ tal que $S(x') \subset S(x)$. Vetores básicos de $X(A, b)$ também são conhecidos como **vértices**. É óbvio que todo poliedro não-vazio $X(A, b)$ tem pelo menos um vértice. vetor básico

$$\begin{array}{rrrrrrrrr}
0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 9 \\
0 & 0 & 1 & 0 & -3 & 0 & 1 & 2 & 8 \\
0 & 1 & 0 & 0 & -4 & 7 & -1 & 3 & 7 \\
1 & 0 & 0 & 0 & 3 & 6 & 1 & 1 & 6 \\
\\
6 & 7 & 8 & 9 & 0 & 0 & 0 & 0 & \\
5 & 8 & 7 & 8 & 0 & 0 & 1 & 0 & \\
0 & 0 & 8 & 9 & 0 & 1 & 0 & 0 &
\end{array}$$

Figura C.1: O topo da figura define um sistema A, b. Os vetores x, x' e x'' definidos nesta ordem pelas três últimas linhas da figura estão todos em $X(A, b)$. A folga de x' não é minimal porque $S(x') \supset S(x)$. A folga de x é minimal porque qualquer vetor x''' em $X(A, b)$ tal que $S(x''') \subseteq S(x)$ é necessariamente igual a x. A folga de x'' é menor que $S(x)$, embora não seja parte de $S(x)$.

Nossa definição de vetores básicos tem um caráter "intrínseco": ela não depende do particular sistema A, b usado para definir o poliedro $X(A, b)$. Já a seguinte observação caracteriza os vetores básicos em termos de propriedades da matriz A.

Proposição C.3 *Um vetor x em $X(A, b)$ é básico se e só se $S(x)$ é li em A.*

DEMONSTRAÇÃO: Suponha que x não é básico. Então existe um vetor x' em $X(A,b)$ tal que $S(x') \subset S(x)$. Seja u o vetor $x - x'$ e observe que $Au = Ax - Ax' = b - b = o$. Observe também que

$$u_{[S]} \neq o \quad \text{e} \quad u_{[N-S]} = o\,,$$

onde S é uma abreviatura para $S(x)$. A existência de um tal vetor u mostra que S é ld.

Agora considere a recíproca. Suponha que x está em $X(A,b)$ e $S(x)$ é ld. Então existe um vetor u tal que

$$Au = o\,, \quad u_{[S]} \neq o \quad \text{e} \quad u_{[N-S]} = o\,,$$

onde $S = S(x)$. Vamos mostrar que S não é minimal. Ajuste a notação, trocando o sinal de u se necessário, de modo que $u_{[j]} > 0$ para algum j em S. Seja λ o maior número tal que

$$\lambda \;\leq\; x_{[j]}/u_{[j]} \tag{C.a}$$

para todo j em S tal que $u_{[j]} > 0$. Observe agora que $x - \lambda u \geq o$. De fato, se $u_{[j]} > 0$ então $x_{[j]} - \lambda u_{[j]} \geq 0$ em virtude da maneira como λ foi definido e se $u_{[j]} \leq 0$ então $x_{[j]} - \lambda u_{[j]} \geq 0$ uma vez que $\lambda \geq 0$.

Como $x - \lambda u \geq 0$ e $A(x - \lambda u) = Ax - \lambda Au = Ax = b$, o vetor $x - \lambda u$ está em $X(A,b)$. Por outro lado, como $u_{[j]} = 0$ sempre que $x_{[j]} = 0$, temos

$$S(x - \lambda u) \;\subseteq\; S\,.$$

Ademais, a inclusão é estrita, uma vez que (C.a) vale com igualdade para algum j em S. Concluimos assim que S não é minimal e portanto x não é básico. □

Esta demonstração, juntamente com a demonstração da proposição C.1, sugere um algoritmo que, ao receber um elemento x de $X(A,b)$, decide que x é um vértice ou devolve um vetor x' em $X(A,b)$ tal que $S(x') \subset S(x)$. algoritmo

Corolário C.4 *Para quaisquer vértices z e z' de $X(A,b)$, se $S(z) = S(z')$ então $z = z'$.*

DEMONSTRAÇÃO: Seja u o vetor $z - z'$. É claro que $Au = Az - Az' = b - b = o$. É claro também que $u_{[N-S]} = o$, onde S é o valor comum de $S(z)$ e $S(z')$. Como S é li, concluimos que $u_{[S]} = o$. Portanto $u = o$ e assim $z = z'$. □

De acordo com o corolário C.4, o número de vértices de $X(A,b)$ é menor que número de subconjuntos de N, ou seja, menor que $2^{|N|}$. Grosso modo, esta delimitação é a melhor possível: existem poliedros com até $\binom{|N|}{|N|/2}$ vértices, e esse número é maior que $2^{|N|/2}$.

C.4 Soluções do problema canônico primal

As soluções do problema canônico primal são vértices, como veremos a seguir.

Fato C.5 *Se o problema* $\mathsf{CP}(A, b, c)$ *tem solução então alguma das soluções é um vértice de* $X(A, b)$.

DEMONSTRAÇÃO: Suponha que o problema $\mathsf{CP}(A, b, c)$ tem solução. Então, de acordo com o algoritmo Simplex, existem matrizes F e G e um vetor y tais que $FG = I$ e o sistema $GA, Gb, c - yA$ é simples. Digamos que Q é uma base de colunas do sistema simples. Então $GA_{[\,,Q]}$ é uma matriz de injeção e portanto Q é li. Seja x o vetor básico associado à base Q. Então $S(x) \subseteq Q$ e portanto $S(x)$ é li. De acordo com a proposição C.3, x é um vértice. □

Cada iteração da segunda fase do algoritmo Simplex (seção 4.2, página 43) começa, implicitamente, com um vértice de $X(A, b)$. Se o vértice não for satisfatório, o algoritmo caminha para um dos vértices "vizinhos". Os mecanismos de convergência — como a regra lexicográfica e a regra de Bland discutidos no capítulo 5 — evitam que um mesmo vértice seja examinado mais de uma vez. A execução do algoritmo termina ao encontrar um vértice que seja solução do problema (ou ao constatar que o problema é inviável ou ilimitado).

Fato C.6 *Se* z *é um vértice de* $X(A, b)$ *então existe um vetor* c *tal que* z *é a única solução do problema* $\mathsf{CP}(A, b, c)$.

DEMONSTRAÇÃO: Adote a abreviatura $S = S(z)$. Seja c o vetor definido pelas equações $c_{[S]} = o$ e $c_{[j]} = 1$ para cada j em $N - S$. É evidente que $cx \geq 0$ para todo x em $X(A, b)$. Mas $cz = 0$, uma vez que $z_{[N-S]} = o$. Logo, z minimiza cz e portanto é solução do problema $\mathsf{CP}(A, b, c)$.

Resta mostrar que z é a única solução do problema. Suponha pois que $cx = cz$ para algum x em $X(A, b)$. Então $cx = 0$ e portanto $x_{[N-S]} = o$, donde $S(x) \subseteq S(z)$. Como $S(z)$ é minimal, é preciso ter $S(x) = S(z)$, donde x também é vértice. Agora o corolário C.4 permite concluir que $x = z$. □

C.5 Poliedros limitados

O poliedro $X(A, b)$ é **limitado** se existe um número ψ tal que $x_{[j]} \leq \psi$ para todo x em $X(A, b)$ e todo j. O conceito é geometricamente intuitivo, mas um tanto abstrato, uma vez que não está claro, de imediato, como verificar algoritmicamente se um dado poliedro é ou não limitado. Também não está claro, de imediato, como *certificar* o caráter limitado ou ilimitado de um poliedro. Ainda assim é útil restringir a atenção aos poliedros limitados pois eles têm uma estrutura muito simples.

Teorema C.7 (da decomposição) *Se $X(A,b)$ é limitado então todo elemento de $X(A,b)$ é combinação convexa de vértices.*

DEMONSTRAÇÃO: Seja x um elemento de $X(A,b)$. Se x é um vértice então a proposição é trivialmente verdadeira. Suponha agora que x não é um vértice e adote como hipótese de indução a validade da proposição para todo elemento x' de $X(A,b)$ tal que $S(x') \subset S(x)$.

De acordo com a proposição C.3, $S(x)$ é ld em A. Portanto, existe um vetor u tal que

$$Au = o\,, \quad u_{[S]} \neq o \quad \text{e} \quad u_{[N-S]} = o\,,$$

onde S é uma abreviatura para $S(x)$. É preciso investigar agora para que valores de λ o vetor $x \pm \lambda u$ está em $X(A,b)$. É claro que $A(x \pm \lambda u) = b$ para qualquer λ; resta estudar a validade de $x \pm \lambda u \geq 0$.

Suponha por um instante que $u \leq o$. Então, para qualquer λ positivo, o vetor $x - \lambda u$ estará em $X(A,b)$. Como $u_{[j]} \neq 0$ para algum j, o valor de $(x - \lambda u)_{[j]}$ será tanto maior quanto maior for λ, e isso é inconsistente com o caráter limitado de $X(A,b)$. É forçoso concluir, portanto, que u tem pelo menos um componente positivo.

A partir daqui os cálculos são iguais aos da demonstração da proposição C.3. Seja λ o maior número que satisfaz a restrição

$$\lambda \;\leq\; x_{[k]}/u_{[k]}$$

para todo k em S tal que $u_{[k]} > 0$. Não é difícil verificar que $x - \lambda u$ está em $X(A,b)$. Também é fácil constatar que $S(x - \lambda u) \subset S$. Nossa hipótese de indução garante então que $x - \lambda u$ é uma combinação convexa de vértices.

Um raciocínio paralelo ao que acabamos de fazer permite concluir que u tem pelo menos um componente negativo e portanto existe um número λ' tal que $x + \lambda' u$ está em $X(A,b)$ e $S(x + \lambda' u) \subset S(x)$. A partir daí, nossa hipótese de indução garante que $x + \lambda' u$ é uma combinação convexa de vértices.

Como x é combinação convexa de $x - \lambda u$ e $x + \lambda' u$ (de fato, $x = \frac{\lambda'}{\lambda+\lambda'}(x - \lambda u) + \frac{\lambda}{\lambda+\lambda'}(x + \lambda' u)$), e esses dois vetores são combinações convexas de vértices, também x é combinação convexa de vértices. □

Esta demonstração pode ser facilmente convertida num algoritmo que, ao receber um sistema A, b e um elemento x de $X(A,b)$, constata que $X(A,b)$ não é limitado ou devolve vértices $z_1, \ldots, z_p$ e números não-negativos $\lambda_1, \ldots, \lambda_p$ tais que $\sum \lambda_i = 1$ e $x = \sum \lambda_i z_i$.

O teorema C.7 tem o seguinte corolário imediato: Se $X(A,b)$ é limitado então $X(A,b) = [Z]$, onde Z é o conjunto dos vértices do poliedro. Isso mostra que, em termos geométricos, os vértices estão na "casca" ou "fronteira" de $X(A,b)$.

Exercícios

C.1 Suponha que W é um conjunto finito de vetores sobre N e c um vetor sobre N. Mostre que o mínimo de cx para x em $[W]$ é igual ao mínimo de cw para w em W.

C.2 Um conjunto li K é **maximal** se nenhum superconjunto próprio de K é li. Mostre que todos os conjuntos li maximais têm a mesma cardinalidade (veja exercício 2.11).

C.3 Mostre que o número de vértices de $X(A,b)$ não passa de $\binom{|N|}{|Q|}$, onde Q é qualquer conjunto li maximal (veja exercício C.2) em A.

C.4 Suponha que um elemento x de $X(A,b)$ é combinação convexa de dois elementos de $X(A,b)-\{x\}$; mostre que x não é um vértice. Agora suponha que um elemento x de $X(A,b)$ não é vértice e mostre que x é combinação convexa de dois vetores em $X(A,b)-\{x\}$.

C.5 Suponha que $X(A,b) = [W]$, onde W é um conjunto finito de vetores. Mostre que $X(A,b)$ é limitado.

C.6 Suponha que o poliedro $X(A,b)$ é limitado e mostre que $X(A,b)$ é vazio ou $X(A,o) = \{o\}$. Suponha que $X(A,o) = \{o\}$ e mostre que $X(A,b)$ é limitado.

C.7 Escreva um algoritmo que receba um sistema A,b e decida se o poliedro $X(A,b)$ é ou não limitado.

C.8 Suponha que $X(A,b)$ é limitado. Mostre que, para qualquer vetor c, o problema $\mathsf{CP}(A,b,c)$ é inviável ou tem solução.

C.9 Mostre que todo vetor em $X(A,b)$ é da forma $x+r$ onde x é uma combinação convexa de vértices e r um elemento de $X(A,o)$.

C.10 Um elemento r de $X(A,o)-\{o\}$ é um **raio** se seu suporte, $S(r)$, é minimal. Uma **combinação cônica** de vetores $x_1,\ldots,x_k$ é qualquer combinação linear $\lambda_1 x_1 + \cdots + \lambda_k x_k$ tal que $\lambda_i \geq 0$ para todo i. Mostre que todo elemento de $X(A,o)$ é combinação cônica de raios.

Apêndice D

Poliedro canônico dual

Este apêndice estuda algumas propriedades geométricas do poliedro canônico dual

$$Y(A, c)\,.$$

Esse é o conjunto de todos os vetores y tais que $yA \leq c$. Suporemos, ao longo do apêndice, que A é uma matriz sobre $M \times N$ e que c é um vetor sobre N.

D.1 Conjuntos geradores

A seguinte observação é uma espécie de "dual" da proposição C.1.

Proposição D.1 *Para qualquer parte K de N, vale uma e apenas uma das seguintes alternativas:*

(1) existem matrizes F e G tais que $FG = I$ e GA é uma matriz escalonada cuja base de colunas é parte de K;
(2) existe um vetor v tal que $(vA)_{[K]} = o$ mas $(vA)_{[N-K]} \neq o$.

DEMONSTRAÇÃO: Seja B a matriz $A_{[\,,K]}$. De acordo com algoritmo de Gauss-Jordan, existem matrizes F e G tais que $FG = I$ e GB é escalonada. Seja Q uma base de colunas de GB. Se a matriz GA é escalonada então vale a alternativa *(1)*. Suponha agora que GA não é escalonada. Então $(GA)_{[i,\,]} \neq o$ para algum i em $M - P$, onde P é a base de linhas de GB. Seja v o vetor $G_{[i,\,]}$. Então

$$(vA)_{[K]} \;=\; (GA)_{[i,\,K]} \;=\; (GB)_{[i,\,]} \;=\; o\,.$$

Mas $(vA)_{[N-K]} \neq o$, uma vez que $vA = (GA)_{[i,\,]} \neq o$. Assim, vale a alternativa *(2)*.

Resta mostrar que as duas alternativas não podem ser simultaneamente verdadeiras. Suponha que vale a alternativa *(1)*. Seja E a matriz escalonada GA, seja P a base de linhas de E e seja Q uma base de colunas de E tal que

$Q \subseteq K$. Tome qualquer vetor v tal que $(vA)_{[K]} = o$ e seja w o vetor vF. Como $E_{[M-P,\,]} = O$, temos

$$w_{[P]}E_{[P,\,]} \;=\; wE \;=\; vFGA \;=\; vA\,.$$

Em particular, $w_{[P]}E_{[P,Q]} = (vA)_{[Q]}$. Mas $(vA)_{[Q]} = o$ uma vez que $Q \subseteq K$. Logo,

$$w_{[P]} = o\,,$$

já que $E_{[P,Q]}$ é uma matriz de bijeção. Segue daí que $vA = w_{[P]}E_{[P,\,]} = o$. Portanto, a alternativa *(2)* não pode valer. □

Na linguagem da álgebra linear, a alternativa *(1)* afirma que cada coluna de $A_{[\,,N-K]}$ é uma combinação linear das colunas de $A_{[\,,K]}$ enquanto a alternativa *(2)* afirma que alguma coluna da matriz $A_{[\,,N-K]}$ é linearmente independente das colunas de $A_{[\,,K]}$. Diremos que K é um **gerador de** A se vale a alternativa *(1)*. É óbvio que todo superconjunto de um gerador também é gerador. gerador

A demonstração da proposição D.1 pode ser facilmente convertida em um algoritmo que, ao receber A e K, decide se K é ou não um gerador. No primeiro caso, o algoritmo devolve um par F, G de matrizes para atestar o caráter gerador de K; no segundo caso, o algoritmo devolve um vetor v para atestar que K não é um gerador. algoritmo

D.2 Combinações convexas

O poliedro canônico dual é um conjunto convexo no seguinte sentido:

Proposição D.2 *Para qualquer parte finita Z de $Y(A,c)$, toda combinação convexa de elementos de Z está em $Y(A,c)$.*

DEMONSTRAÇÃO: Seja $\sum \lambda_i z_i$ uma combinação convexa dos elementos de Z. Como $z_i A \le c$ e $\lambda_i \ge 0$ para todo i,

$$(\textstyle\sum \lambda_i z_i)A \;=\; \sum \lambda_i z_i A \;\le\; \sum \lambda_i c \;=\; (\sum \lambda_i)\, c \;=\; c\,,$$

uma vez que $\sum \lambda_i = 1$. Portanto, $\sum \lambda_i z_i$ está em $Y(A,c)$. □

A proposição D.2 mostra que $[Z] \subseteq Y(A,c)$ para toda parte finita Z de $Y(A,c)$, onde $[Z]$ é a envoltória convexa de Z. Em certas circunstâncias, $Y(A,c) = [Z]$ para um certo conjunto finito Z. A caracterização de um tal Z é o principal objetivo deste apêndice.

D.3 Vetores básicos e faces minimais

A **folga** de um elemento y de $Y(A,c)$ é o conjunto de todos os índices j em N para os quais $(yA)_{[j]} < c_{[j]}$. A folga de y será denotada por

folga

$$S(y)\,.$$

Um vetor y em $Y(A,c)$ é **básico** se $S(y)$ é minimal, ou seja, se não existe y' em $Y(A,c)$ tal que $S(y') \subset S(y)$. É óbvio que todo poliedro não-vazio $Y(A,c)$ tem pelo menos um vetor básico. A seguinte proposição caracteriza os vetores básicos em termos de propriedades da matriz A.

vetor básico

Proposição D.3 *Um vetor y em $Y(A,c)$ é básico se e só se $N - S(y)$ é um gerador de A.*

DEMONSTRAÇÃO: Seja y um vetor em $Y(A,c)$ e suponha que y não básico, ou seja, que $S(y)$ não é minimal. Então existe um vetor y' em $Y(A,c)$ tal que $S' \subset S$, onde S' e S são abreviaturas para $S(y')$ e $S(y)$ respectivamente. Vamos mostrar que $N - S$ não é um gerador de A. Seja v o vetor $y - y'$ e observe que

$$(vA)_{[N-S]} \;=\; (yA)_{[N-S]} - (y'A)_{[N-S]} \;=\; c_{[N-S]} - c_{[N-S]} \;=\; o\,.$$

Agora tome qualquer k em $S - S'$ e observe que

$$(vA)_{[k]} \;=\; (yA)_{[k]} - (y'A)_{[k]} \;\neq\; 0\,,$$

uma vez que $(yA)_{[k]} < c_{[k]}$ enquanto $(y'A)_{[k]} = c_{[k]}$. Assim, de acordo com a proposição D.1, $N - S$ não é um gerador de A.

Agora considere a recíproca. Seja y um vetor em $Y(A,c)$ e suponha que $N - S$ não é gerador de A, onde S é $S(y)$. Então existe um vetor v tal que

$$(vA)_{[N-S]} = o \quad \text{mas} \quad (vA)_{[S]} \neq o\,.$$

Ajuste a notação, trocando v por $-v$ se necessário, de modo que $(vA)_{[j]} > 0$ para algum j em S. Seja λ o maior número tal que

$$\lambda \;\leq\; \frac{(c - yA)_{[j]}}{(vA)_{[j]}} \tag{D.a}$$

para todo j em S tal que $(vA)_{[j]} > 0$. É fácil verificar que $yA + \lambda vA \leq c$. De fato, se $(vA)_{[j]} > 0$ então $(yA)_{[j]} + \lambda(vA)_{[j]} \leq c_{[j]}$ em virtude da maneira como λ foi escolhido; e se $(vA)_{[j]} \leq 0$ então $(yA)_{[j]} + \lambda(vA)_{[j]} \leq c_{[j]}$ uma vez que $\lambda \geq 0$. Portanto, $y + \lambda v$ está em $Y(A,c)$. Por outro lado, como $(vA)_{[j]} = 0$ sempre que $(yA)_{[j]} = c_{[j]}$, temos

$$S(y + \lambda v) \;\subseteq\; S\,.$$

Ademais, a inclusão é estrita, uma vez que (D.a) vale com igualdade para algum j em S. Logo, S não é minimal e portanto y não é básico. □

Essa demonstração, juntamente com a demonstração da proposição D.1, sugere um algoritmo que, ao receber um elemento y de $Y(A,c)$, decide que y é um vetor básico ou devolve um vetor y' em $Y(A,c)$ tal que $S(y') \subset S(y)$. algoritmo

Corolário D.4 *Para quaisquer vetores básicos z e z', se $S(z) = S(z')$ então $zA = z'A$.*

DEMONSTRAÇÃO: Seja v o vetor $z - z'$ e seja S é o valor comum de $S(z)$ e $S(z')$. É claro que

$$(vA)_{[N-S]} \;=\; (zA)_{[N-S]} - (z'A)_{[N-S]} \;=\; c_{[N-S]} - c_{[N-S]} \;=\; o\,,$$

Como $N-S$ é um gerador de A, a proposição D.1 garante que $(vA)_{[S]} = o$ Logo, $vA = o$ e portanto $zA = z'A$. □

Uma **face minimal** de $Y(A,b)$ é o conjunto de todos os vetores básicos que têm uma mesma folga. Assim, dois vetores básicos, digamos z e z', pertencem à mesma face minimal se e só se $S(z) = S(z')$. O corolário D.4 mostra que z e z' estão na mesma face minimal se e só se $zA = z'A$. Toda face minimal é um conjunto afim, no seguinte sentido. face minimal

Proposição D.5 *Toda combinação afim de vetores de uma face minimal pertence à face minimal.*

DEMONSTRAÇÃO: Digamos que os vetores básicos $z_1, \ldots, z_k$ pertencem a uma face minimal Y'. Agora suponha que $\lambda_1, \ldots, \lambda_k$ são números tais que $\sum \lambda_i = 1$. Como $z_i A = z_1 A$ para todo i temos também

$$(\textstyle\sum \lambda_i z_i)A \;=\; \sum \lambda_i (z_i A) \;=\; (\sum \lambda_i) z_1 A \;=\; z_1 A\,.$$

Portanto, $\sum \lambda_i z_i$ é um vetor básico de $Y(A,c)$ e pertence à face minimal Y'. □

Se uma face minimal contém um único vetor diremos que esse vetor é um **vértice** de $Y(A,c)$. Em outras palavras, um vetor básico z é um vértice se o conjunto unitário $\{z\}$ é uma face minimal. vértice

Se o conjunto de linhas de A é li — ou seja, se existe uma matriz G tal que GA é escalonada e tem base de linhas M — então a igualdade $zA = z'A$ implica em $z = z'$, e portanto todo vetor básico é um vértice.

D.4 Soluções do problema canônico dual

As soluções do problema canônico dual são vetores básicos, como veremos a seguir.

Fato D.6 *Se o problema* $\mathsf{CD}(A, c, b)$ *tem solução então alguma das soluções é um vetor básico de* $Y(A, c)$.

DEMONSTRAÇÃO: Suponha que o problema $\mathsf{CD}(A, c, b)$ tem solução. Então, de acordo com o algoritmo Simplex, existem matrizes F e G e um vetor y tais que $FG = I$ e o sistema $GA, Gb, c - yA$ é simples. Digamos que Q é uma base de colunas do sistema simples. É claro que Q é um gerador de A. Como $(c - yA)_{[Q]}$ é nulo, Q é parte de $N - S(y)$, donde $N - S(y)$ é um gerador de A. Assim, de acordo com a proposição D.3, y é um vetor básico. □

Fato D.7 *Para todo vetor básico* z *de* $Y(A, c)$*, existe um vetor* b *tal que os vetores da face minimal que contém* z *são as únicas soluções do problema* $\mathsf{CD}(A, c, b)$.

DEMONSTRAÇÃO: Adote a abreviatura $S = S(z)$. Seja A' a matriz $A_{[\,, N-S]}$ e seja b o vetor $A'u$, onde u é o vetor 1s sobre $N - S$ (isto é, $u_{[j]} = 1$ para todo j em $N - S$). Seja c' o vetor $c_{[N-S]}$. É evidente que $yb = yA'u \leq c'u$ para todo y em $Y(A, c)$. Mas $zb = zA'u = c'u$. Logo, z maximiza zb e portanto é solução do problema $\mathsf{CD}(A, c, b)$.

Suponha agora y é uma solução do problema $\mathsf{CD}(A, c, b)$; vamos mostrar y és básico e pertence à mesma face minimal que z. Como $yb = c'u$, temos necessariamente $yA' = c'$ e portanto $S(y) \subseteq S(z)$. Como $S(z)$ é minimal, é preciso ter $S(y) = S(z)$, donde y é um vetor básico. Agora o corolário D.4 permite concluir que $yA = zA$, garantindo assim que y está na mesma face minimal que z. □

D.5 Poliedros limitados

O poliedro canônico dual $Y(A, c)$ é **limitado** se existe um número ψ tal que $|y_{[i]}| \leq \psi$ para todo y em $Y(A, c)$ e todo i. O conceito é um tanto abstrato, uma vez que não está claro, de imediato, como verificar algoritmicamente se um dado poliedro é ou não limitado. Ainda assim é útil restringir a atenção aos poliedros limitados pois eles têm uma estrutura muito simples.

Fato D.8 *Se* $Y(A, c)$ *é limitado então todos os seus vetores básicos são vértices.*

DEMONSTRAÇÃO: Suponha que uma face minimal contém dois vetores básicos distintos, digamos z e z'. De acordo com a proposição D.5, o vetor $\lambda z + (1 - \lambda)z'$ também está em $Y(A, b)$, qualquer que seja λ. Seja i um índice tal que $z_{[i]} \neq z'_{[i]}$. Como

$$|(\lambda z + (1 - \lambda)z')_{[i]}| \;\geq\; |\lambda(z_{[i]} - z'_{[i]})| - |z'_{[i]}|\,,$$

vemos que o valor absoluto do componente i de $\lambda z + (1-\lambda)z'$ é tanto maior quanto maior for o valor absoluto de λ. Mas isso é incompatível com o caráter limitado de $Y(A,c)$. Concluimos assim que toda face minimal de $Y(A,c)$ é unitária. □

Teorema D.9 (da decomposição dual) *Se $Y(A,c)$ é limitado então todo elemento de $Y(A,c)$ é combinação convexa de vértices.*

DEMONSTRAÇÃO: - Seja y um elemento de $Y(A,b)$. Se y é um vértice então a proposição é trivialmente verdadeira. Suponha agora que y não é um vértice e adote como hipótese de indução a validade da proposição para todo elemento y' de $Y(A,c)$ tal que $S(y') \subset S(y)$.

De acordo com a proposição D.3, $S(y)$ não é gerador de A. Portanto, existe um vetor v tal que

$$(vA)_{[N-S]} = o \quad \text{e} \quad (vA)_{[S]} \neq o\,,$$

onde $S = S(y)$. Suponha por um instante que $vA \leq o$. Então, para qualquer número positivo λ teremos $(y+\lambda v)A \leq c$, donde $y+\lambda v$ estará em $Y(A,c)$. Como $v_{[i]}$ não é nulo para algum i, o valor absoluto de $(y+\lambda v)_{[i]}$ será tanto maior quanto maior for λ, e isso é inconsistente com o caráter limitado de $Y(A,c)$. É forçoso concluir, portanto, que vA tem pelo menos um componente positivo.

A partir daqui os cáculos são iguais aos que fizemos na demosntração da proposição D.3. Seja λ o maior número que satisfaz a restrição λ

$$\lambda \;\leq\; \frac{(c - yA)_{[j]}}{(vA)_{[j]}}$$

para todo j em S tal que $(vA)_{[j]} > 0$. Então $yA + \lambda vA \leq c$ e portanto $y + \lambda v$ está em $Y(A,c)$. Ademais, $S(y+\lambda v) \subset S$. Nossa hipótese de indução garante agora que $y + \lambda v$ é uma combinação convexa de vértices.

Um raciocínio análogo ao que acabamos de fazer permite concluir que vA tem pelo menos um componente negativo e portanto existe um número λ' tal que $y - \lambda' v$ está em $Y(A,c)$ e $S(y-\lambda' v) \subset S(y)$. A partir daí, nossa hipótese de indução garante que $y - \lambda' v$ é uma combinação convexa de vértices.

Como y é combinação convexa de $y + \lambda v$ e $y - \lambda' v$, e esses dois vetores são combinações convexas de vértices, também y é combinação convexa de vértices. □

Exercícios

D.1 Suponha que W é um conjunto finito de vetores sobre M e b um vetor sobre M. Mostre que o máximo de yb para y em $[W]$ é igual ao máximo de yw para w em W.

D.2 Um conjunto gerador K é **minimal** se nenhum subconjunto próprio de K é gerador. Mostre que todos os conjuntos geradores minimais têm a mesma cardinalidade.

D.3 Mostre que todo conjunto gerador minimal tem a mesma cardinalidade que qualquer conjunto li minimal (veja exercício C.2).

D.4 Mostre que o número de faces minimais de $Y(A, c)$ não passa de $\binom{|N|}{|Q|}$, onde Q é qualquer conjunto gerador minimal de A.

D.5 Suponha que um elemento y de $Y(A, c)$ é combinação convexa de dois elementos, digamos y' e y'' de $Y(A, c) - \{y\}$. Suponha que y é básico. Mostre que y' e y'' também são básicos e pertencem à mesma face minimal que y.

D.6 Suponha que $Y(A, c)$ possui um vértice, ou seja, uma face minimal unitária. Mostre que todas as faces minimais são unitárias.

D.7 Suponha que $Y(A, c) = [W]$, onde W é um conjunto finito de vetores. Mostre que $Y(A, c)$ é limitado.

D.8 Suponha que o poliedro $Y(A, c)$ é limitado e mostre que $Y(A, c)$ é vazio ou $Y(A, o) = \{o\}$. Suponha que $Y(A, o) = \{o\}$ e mostre que $Y(A, c)$ é limitado.

D.9 Escreva um algoritmo que receba um sistema A, c e decida se o poliedro $Y(A, c)$ é ou não limitado.

D.10 Suponha que $Y(A, c)$ é limitado. Mostre que, para qualquer vetor b, o problema $\mathsf{CD}(A, c, b)$ é inviável ou tem solução.

D.11 Mostre que todo vetor em $Y(A, c)$ é da forma $y + r$ onde x é uma combinação convexa de vetores básicos e r um elemento de $Y(A, o)$.

Apêndice E

Exercícios resolvidos

Este apêndice contém as soluções de alguns dos exercícios propostos no texto.

E.1 Solução do exercício 2.5

A figura E.1 descreve o algoritmo de Gauss-Jordan em uma linguagem mais formal que aquela do texto. Vamos supor que os elementos de M são $1, 2, \ldots, m$ e os elementos de N são $1, 2, \ldots, n$.

As bases P e Q são representadas por um vetor φ sobre M: se i está em P então $\varphi[i]$ é único q em Q tal que $E[i, q] = 1$; se i está em $M - P$ então $\varphi[i] = 0$.

O pseudocódigo da figura E.1 supõe que nosso computador é capaz de executar aritmética racional exata. Supõe também que, durante a pivotação, as operações aritméticas que envolvem um operando nulo não consomem tempo algum (senão, as operações relativas às colunas do conjunto $Q + k$ deveriam ser executadas apenas implicitamente). Nossa solução usa o espaço de memória de maneira pouco econômica, pois armazena as matrizes E, G e F em separado.

E.2 Solução do exercício 2.11

Suponha que G_1 e G_2 são matrizes inversíveis tais que G_1D e G_2D são escalonadas. Queremos mostrar que as bases de colunas das duas matrizes escalonadas têm a mesma cardinalidade.

Solução: Sejam Q_1 e Q_2 bases de G_1D e G_2D respectivamente. Seja G uma matriz inversível tal que GD é escalonada e possui uma base de colunas Q de cardinalidade $|Q_1|$; escolha G de modo a minimizar $Q_2 - Q$. Suponha agora que

$$Q_2 - Q \neq \emptyset \tag{E.a}$$

e seja q_2 um elemento desse conjunto. Seja P a base de linhas da matriz $E = GD$. Há uma parte P' de P tal que $E_{[P', Q \cap Q_2]}$ e $E_{[P-P', Q-Q_2]}$ são matrizes de

```
E, G, φ ← D, I, o
para h ← 1 até m faça
    k ← 1
    enquanto k ≤ n e E[h, k] = 0 faça k ← k + 1
    se k ≤ n então
        α ← E[h, k]
        para j ← 1 até m faça G[h, j] ← G[h, j]/α
        para j ← 1 até n faça E[h, j] ← E[h, j]/α
        para i ← 1 até m faça
            se i ≠ h então
                α ← E[i, k]
                para j ← 1 até m faça G[i, j] ← G[i, j] − α · G[h, j]
                para j ← 1 até n faça E[i, j] ← E[i, j] − α · E[h, j]
        φ[h] ← k
para j ← 1 até m faça    ▷ gera matriz F
    se φ[j] = 0 então
        para i ← 1 até m faça F[i, j] ← 0
        F[j, j] ← 1
    senão
        para i ← 1 até m faça F[i, j] ← D[i, φ[j]]
```

Figura E.1: Formalização do algoritmo de Gauss-Jordan (exercício 2.5). A primeira linha atribui valores iniciais às variáveis E, G e φ respectivamente. O primeiro bloco de código cuida da operação de pivotação. O segundo, gera a matriz F a partir do vetor φ.

bijeção. Seja x um vetor sobre Q_2 tal que

$$E_{[P',\,Q_2]} \cdot x = o \quad \text{e} \quad x_{[q_2]} = 1\,. \tag{E.b}$$

Agora suponha, por um instante, que

$$E_{[P-P',\,q_2]} = o\,. \tag{E.c}$$

Então os vetores $E_{[P,\,Q_2]} \cdot x$ e $E_{[\,,\,Q_2]} \cdot x$ são nulos, ou seja,

$$GD_2x = o\,, \tag{E.d}$$

onde D_2 denota a matriz $D_{[\,,\,Q_2]}$. Seja F uma matriz tal que $FG = I$. Se multiplicarmos ambos os lados de (E.d) por G_2F, teremos

$$G_2D_2x = o\,.$$

Como Q_2 é uma base de colunas da matriz escalonada G_2D, o vetor x é nulo, o que é inconsistente com (E.b). (Na linguagem da álgebra da linear, diríamos simplesmente que o conjunto de colunas da matriz G_2D_2 é linearmente independente.) Concluímos assim que a hipótese (E.c) é insustentável, ou seja, que

existe p em $P - P'$ tal que $E_{[p,\,q_2]} \neq 0$. É possível, agora, executar uma pivotação em torno de p, q_2.

Seja $\check{F}$ a matriz elementar cuja coluna saliente, p, é igual a $E_{[\,,\,q_2]}$. Seja $\check{G}$ a inversa de $\check{F}$. É claro que a matriz $\check{G}G$ é inversível e que $\check{G}E$ é escalonada e tem base de colunas $Q - q + q_2$, onde q é o único elemento de Q que possui a propriedade $E_{[p,\,q]} \neq 0$. Está claro que $|Q - q + q_2| = |Q_1|$ e $Q_2 - (Q - q + q_2)$ é parte própria de $Q_2 - Q$, o que torna a existência da matriz $\check{G}G$ é incompatível com nossa escolha de G. Logo, a hipótese (E.a) é insustentável. Segue daí que

$$|Q_2| \leq |Q_1| \,.$$

Se repetirmos o raciocínio todo depois de inverter os papéis de Q_1 e Q_2 concluiremos que $|Q_1| \leq |Q_2|$.

Corolário. Para qualquer matriz D, qualquer matriz inversível G_1 tal que $G_1 D$ é escalonada e qualquer matriz inversível G_2 tal que $G_2 D$ é escalonada, as bases de linhas das duas matrizes escalonadas têm a mesma cardinalidade.

E.3 Solução do exercício 2.13

Dada uma matriz A sobre $M \times N$, um vetor b sobre M e um vetor c sobre N, encontrar x tal que $Ax = b$ e cx é mínimo.

Solução: Use o algoritmo de Gauss-Jordan para obter uma matriz inversível G e um vetor g tais que GA é escalonada e

$$(c - gA)_{[Q]} = o \,,$$

onde Q é uma base de colunas de GA. Seja P a base de linhas de GA.

CASO 1: $(Gb)_{[M-P]} \neq o$. Neste caso o problema não tem solução porque não existe x tal que $Ax = b$. De fato, se um tal x existisse teríamos a contradição $o = (GA)_{[M-P]}\, x = (Gb)_{[M-P]} \neq o$.

CASO 2: $(Gb)_{[M-P]} = o$ e $c - gA = o$. Seja, x o vetor definido pelas condições

$$x_{[N-Q]} = o \quad \text{e} \quad (GA)_{[\,,\,Q]}\, x_{[Q]} = Gb \,.$$

Como $GAx = Gb$ e G é inversível, temos $Ax = b$. Observe agora que $(c - gA)x$ é nulo, donde $cx = gb$. Por outro lado, para qualquer x' tal que $Ax' = b$ teremos $(c - gA)x' = o$, donde $cx' = gb$. Portanto, x é uma solução do problema.

CASO 3: $(Gb)_{[M-P]} = o$ e $c - gA \neq o$. Seja j um índice tal que $(c - gA)_{[j]} \neq 0$; é claro que $j \in N - Q$. Seja x o vetor definido pelas seguintes condições:

$$(c - gA)_{[j]}\, x_{[j]} < 0 \,, \quad x_{[(N-Q)-j]} = o \quad \text{e} \quad (GA)x = Gb \,.$$

```
E, G, φ, h ← D, I, o, 1
repita    ▷ primeira fase
    k_1 ← k_2 ← n
    se E[h, n] ≥ 0 então
        k_1 ← 0
        repita k_1 ← k_1 + 1
            até que k_1 = n ou E[h, k_1] > 0
    se E[h, n] ≤ 0 então
        k_2 ← 0
        repita k_2 ← k_2 + 1
            até que k_2 = n ou E[h, k_2] < 0
    k ← min (k_1, k_2)
    se k < n então
        p ← LINHA-MÍNIMA (h, k)
        se E[p, n]/E[p, k] ≥ E[h, n]/E[h, k] então
            PIVOTAÇÃO (h, k)
            h ← h + 1
        senão PIVOTAÇÃO (p, k)
    senão
        se E[h, n] = 0 então h ← h + 1
        senão pare (h é linha de inviabilidade)
até que h = m
▷ fim da primeira fase e início da segunda
repita
    k ← 0
    repita k ← k + 1
        até que k = n ou E[m, k] < 0
    se k < n então
        p ← LINHA-MÍNIMA (m, k)
        se p = m então pare (k é coluna de ilimitação)
        senão PIVOTAÇÃO (p, k)
até que k = n
```

Figura E.2: Formalização da heurística Simplex (exercício 4.2). A primeira linha atribui valores iniciais às variáveis E, G, φ e h respectivamente. As funções LINHA-MÍNIMA e PIVOTAÇÃO estão descritas nas figuras seguintes. A matriz F poderia ser determinada, a partir de D e φ, como na figura E.1.

É claro que $Ax = b$. É claro também que $cx - gb = (c - gA)x = (c - gA)_{[j]}\, x_{[j]}$, donde

$$cx = (c - gA)_{[j]}\, x_{[j]} + gb\,.$$

Logo, cx é tanto menor quanto maior for o valor absoluto de $x_{[j]}$. Concluímos assim que o problema não tem solução.

```
LINHA-MÍNIMA (h, k)
  p ← 0
  repita p ← p + 1
      até que p = h ou E[p, k] > 0
  para i ← p + 1 até h − 1 faça
      se E[i, k] > 0 e E[i, n]/E[i, k] < E[p, n]/E[p, k] então
          p ← i
  devolva p
```

Figura E.3: A função LINHA-MÍNIMA (usada no código da figura E.2) recebe índices h e k e devolve p entre 1 e $h-1$ tal que $E[p,k]$ é positivo e $E[p,n]/E[p,k]$ é mínimo; se tal p não existe, a função devolve h.

```
PIVOTAÇÃO (h, k)
  α ← E[h, k]
  para j ← 1 até m faça
      G[h, j] ← G[h, j]/α
  para j ← 1 até n faça
      E[h, j] ← E[h, j]/α
  para i ← 1 até m faça
      se i ≠ h então
          α ← −E[i, k]
          para j ← 1 até m faça
              G[i, j] ← G[i, j] − α · G[h, j]
          para j ← 1 até n faça
              E[i, j] ← E[i, j] − α · E[h, j]
  φ[h] ← k      ▷ k entra na base
```

Figura E.4: A função PIVOTAÇÃO (usada no código da figura E.2) recebe índices h e k e executa uma pivotação de G, E em torno de h, k.

E.4 Solução do exercício 4.2

As figuras E.2 a E.4 descrevem a heurística Simplex em uma linguagem mais formal que aquela usada no texto. Esta formalização supõe que os elementos de M são $1, 2, \ldots, m$, os elementos de N são $1, 2, \ldots, n$, a coluna especial é n e a linha especial é m.

Se convergir, a heurística produz não só matrizes G e E mas também índices h e k. Se $1 \leq h < m$ então E é simples inviável e h é uma linha de inviabilidade. Se $1 \leq k < n$ então E é simples ilimitada e k é uma coluna de ilimitação. Em caso contrário, E é simples solúvel.

Os conjuntos P e Q são representados por um vetor φ sobre M definido da seguinte maneira: se i está em P então $\varphi[i]$ é único q em Q tal que $E[i,q] = 1$; senão $\varphi[i] = 0$.

Esta formalização do Simplex supõe um computador capaz de executar aritmética racional exata. Supõe também que, durante a pivotação, as operações aritméticas que envolvem um operando nulo não consomem tempo algum (senão, as operações relativas às colunas do conjunto $Q+k$ deveriam ser executadas apenas implicitamente). Além disso, esta formalização usa o espaço de memória de maneira pouco econômica.

E.5 Solução do exercício 4.3

É instrutivo analisar uma versão especializada da heurística Simplex para matrizes com apenas duas linhas. Essa versão sempre converge; a análise da convergência contém um prenúncio do algoritmo Simplex Lexicográfico.

Suponha que M é um conjunto com apenas dois elementos, h e m. Nossa versão especializada do Simplex recebe uma matriz D sobre $M \times N$ e um elemento n de N e devolve uma matriz G tal que $G_{[h,h]} \neq 0$, $G_{[\,,m]} = I_{[\,,m]}$ e GD é simples com relação à coluna n e à linha m. É claro que a matriz G é inversível; sua inversa F é definida pelas equações $F_{[h,h]} = 1/G_{[h,h]}$, $F_{[m,h]} = -G_{[m,h]}/G_{[h,h]}$ e $F_{[\,,m]} = I_{[\,,m]}$. O algoritmo consiste no seguinte:

CASO 1: $D_{[h,\,N-n]} \leq o$ e $D_{[h,n]} > 0$ ou $D_{[h,\,N-n]} \geq o$ e $D_{[h,n]} < 0$.
Devolva I e pare (D é simples inviável).

CASO 2: $D_{[h,\,N]} = o$.
Devolva I e pare (D é simples solúvel se $D_{[m,\,N-n]} \geq o$ e
simples ilimitada em caso contrário).

CASO 3: $D_{[h,\dot{q}]} > 0$ e $D_{[h,n]} \geq 0$ ou $D_{[h,\dot{q}]} < 0$ e $D_{[h,n]} \leq 0$ para
algum $\dot{q}$ em $N - n$.
Seja $\dot{G}, \dot{E}$ o resultado da pivotação de I, D em torno de $h, \dot{q}$.
Execute o processo iterativo descrito a seguir.
Cada iteração começa com um elemento q de $N - n$ e matrizes G e E.
A primeira iteração começa com $q = \dot{q}$, $G = \dot{G}$ e $E = \dot{E}$.
Cada iteração consiste no seguinte:

CASO 3.1: $E_{[m,k]} < 0$ e $E_{[h,k]} > 0$ para algum k em $N - n$.
Seja G', E' o resultado da pivotação de G, E em torno de h, k.
Comece nova iteração com G' e E' nos papéis de G e E.

CASO 3.2: $E_{[m,k]} < 0$ e $E_{[h,k]} \leq 0$ para algum k em $N - n$.
Devolva G e pare (E é simples ilimitada).

CASO 3.3: $E_{[m,\,N-n]} \geq o$.
Devolva G e pare (E é simples solúvel).

É óbvio que o algoritmo produz o resultado desejado nos casos 1 e 2. Resta analisar o processo iterativo dentro do caso 3. É fácil verificar os seguintes invariantes:

no início de cada iteração,

(i0) $E_{[h,n]} \geq 0$ **e** $E_{[h,\dot{q}]} > 0$,
(i1) $E_{[\,,q]} = I_{[\,,h]}$,
(i2) $G_{[h,h]} \neq 0$,
(i3) $E = GD$,
(i4) $G_{[\,,m]} = I_{[\,,m]}$,
(i6) $G_{[m,h]}\, D_{[h,q]} = -D_{[m,q]}$ **e**
(i7) $D_{[h,q]}\, G_{[h,h]} = 1$.

Esses invariantes valem, em particular, no início da última iteração. Portanto, a matriz G que o algoritmo devolve nos casos 3.2 e 3.3 tem as propriedades desejadas.

	q	$\dot{q}$	n
h	1	$>$	$\geq$
m	0		

Figura E.5: Matriz E no início de uma iteração.

Resta provar a convergência do algoritmo. O processo iterativo dentro do caso 3 termina depois de um número finito de iterações pelos motivos que passamos a expor. Os invariantes (i4), (i6) e (i7) têm a seguinte conseqüência imediata: se duas iterações diferentes começam com o mesmo valor da variável q então os correspondentes valores de G são iguais. Em virtude de (i3), os correspondentes valores de $E_{[m,\dot{q}]}$ também são iguais. Por outro lado, no fim de cada ocorrência do caso 3.1,

$$E'_{[m,\dot{q}]} > E_{[m,\dot{q}]}\,.$$

De fato, $E'_{[m,\dot{q}]} = E_{[m,\dot{q}]} + \alpha E_{[h,\dot{q}]}$, onde $\alpha = -E_{[m,k]}/E_{[h,k]}$. Como α e $E_{[h,\dot{q}]}$ são positivos, temos a desigualdade desejada.

Essas observações levam à seguinte conclusão: duas iterações diferentes jamais começam com o mesmo valor de q. Logo, o número de iterações não excede $|N| - 1$.

Exercícios

E.1 Aplique a heurística Simplex às matrizes descritas abaixo.

$$\begin{array}{cccccc} 2 & 2 & 2 & 2 & 2 & 0 \\ 2 & 1 & 0 & -1 & -2 & 0 \end{array} \qquad \begin{array}{cccccc} 2 & 2 & 2 & -2 & 2 & 0 \\ 2 & 1 & 0 & -1 & -2 & 0 \end{array}$$

E.2 Mostre como eliminar o processo iterativo dentro do caso 3 mediante uma escolha criteriosa de k.

Referências Bibliográficas

[AC78] D. Avis and V. Chvátal. Notes on Bland's pivoting rule. In *Mathematical Programming Study*, volume 8, pages 24–34. North-Holland, 1978. 57

[AMO93] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms and Applications*. Prentice Hall, 1993. 122

[Bea55] E. M. L. Beale. Cycling in the dual simplex algorithm. *Naval Research Logistics Quarterly*, 2:269–275, 1955. 49

[Bla77] R. Bland. New finite pivoting rules for the simplex method. *Mathematics of Operations Research*, 2:103–107, 1977. 62

[Bor87] K. H. Borgwardt. *The Simplex Method — A Probabilistic Approach*. Springer-Verlag, 1987. 57

[CCPS98] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. *Combinatorial Optimization*. John Wiley, 1998. 122

[Chi53] F. Chio. Mémoire sur les fonctions connues sous le nom de résultantes ou de dèterminants. Turin, 1853. 111, 117

[Chv83] V. Chvátal. *Linear Programming*. W. H. Freeman, 1983. i, 22, 23, 46, 61, 69, 85, 86, 141

[CLRS01] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press and McGraw-Hill, second edition, 2001. 23

[Dan63] G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1963. 69

[Edm67] J. Edmonds. Systems of distinct representatives and linear algebra. *J. of Research of the National Bureau of Standards, B*, 71:241–245, 1967. 122

[EG84] J. Edmonds and R. Giles. Total dual integrality of linear inequality systems. In W. R. Pulleyblank, editor, *Progress in Combinatorial Optimization*, pages 117–129, Toronto, 1984. Academic Press. Proceedings of Jubilee Conference, University of Waterloo, 1982. 131

[Eve80] H. Eves. *Elementary Matrix Theory*. Dover, 1980. Republication of the 1966 Allyn and Bacon edition. 111, 117

[GKP94] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics: A Foundation for Computer Science*. Addison-Wesley, second edition, 1994. ii

[GL81] P. Gács and L. Lovász. Khachiyan's algorithm for linear programming. In *Mathematical Programming Study*, volume 14, pages 61–68. North-Holland, 1981. 133, 141

[GL96] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, third edition, 1996. i

[GLS88] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, first edition, 1988. 133, 157

[GLS93] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, second edition, 1993. 133, 157

[Gon89] C. C. Gonzaga. An algorithm for solving linear programming problems in $O(n^3L)$ operations. In N. Megiddo, editor, *Progress in Mathematical Programming: Interior Point and Related Methods*, pages 1–28. Springer-Verlag, New York, 1989. 133

[Gon92] C. C. Gonzaga. Path following methods for linear programming. *SIAM Review*, 34(2):167–227, 1992. *A survey*. 133

[Jor20] W. Jordan. *Handbuch der Vermessungskunde*, volume I. Metzler, Stuttgart, seventh edition, 1920. 13

[Kan39] L. V. Kantorovich. Mathematical methods of organizing and planning production. *Publication House of the Leningrad State University*, 1939. [Translated in *Management Science*, 6 (1960, p.366–422]. 69

[Kar84] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984. 133

[Kha79] L. G. Khachiyan. A polynomial algorithm for linear programming. *Soviet Mathematical Doklady*, 20:191–194, 1979. *Translation of Doklady Akademii Nauk SSSR, vol.244, pp.1093–1096.* 133, 141

[KM72] V. Klee and G. J. Minty. How good is the simplex algorithm? In O. Shisha, editor, *Inequalities, III*, pages 159–175. Academic Press, 1972. 57

[LKS91] J. K. Lenstra, A. H. G. Rinnooy Kan, and A. Schrijver, editors. *History of Mathematical Programming: a Collection of Personal Reminiscences*. North-Holland, 1991. 69

[PS82] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice Hall, 1982. Second printing by Dover, 1998. 133

[Sch86] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley, 1986. 45, 111, 122, 131, 133, 141, 157

[Sch03] A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Number 24 in Algorithms and Combinatorics. Springer, 2003. [Three volumes]. 131, 133, 141

[YL82] B. Yamnitsky and L. A. Levin. An old linear programming algorithm runs in polynomial time. In *23rd Annual Symposium on Foundations of Computer Science*, pages 327–328. IEEE, 1982. 141

Índice Remissivo

Alfabeto grego

α	A	alfa	ι	I	iota	ρ	P	rô
β	B	beta	κ	K	kapa	σ	Σ	sigma
γ	Γ	gama	λ	Λ	lambda	τ	T	tau
δ	Δ	delta	μ	M	mü	υ	Υ	upsilon
ε	E	epsilon	ν	N	nü	φ	Φ	fi
ζ	Z	zeta	ξ	Ξ	ksi	χ	X	qui
η	H	eta	o	O	ômicron	ψ	Ψ	psi
θ	Θ	téta	π	Π	pi	ω	Ω	ômega