Otimização e Processos Estocásticos Aplicados à Economia e Finanças

Compilado em 07 de Setembro de 2007.

Julio Michael Stern [1]
Carlos Alberto de Bragança Pereira [2]
Celma de Oliveira Ribeiro [3]
Cibele Dunder [4]
Fabio Nakano [5]
Marcelo Lauretto [6]

[1]Julio M. Stern (jstern@ime.usp.br) é Ph.D. em Pesquisa Operacional e Engenharia Industrial pela Universidade de Cornell (Ithaca, NY, USA). Atualmente é Professor Livre Docente do Departamento de Ciência da Computação do Instituto de Matemática e Estatística da Universidade de São Paulo, MAC-IME-USP, e consultor na área de Pesquisa Operacional.

[2]Carlos Alberto de Bragança Pereira (cpereira@ime.usp.br) é Ph.D. em Estatística Pela Universidade da Florida, (Talahasse, FL, EUA), Professor Titular do Departamento de Estatística do IME-USP, e consultor na área de Pesquisa Operacional..

[3]Celma de O. Ribeiro (celma@ime.usp.br) é Doutora em Engenharia de Produção pela Escola Politécnica da USP (EPUSP). É docente do Departamento de Engenharia de Produção da EPUSP, atuando em áreas ligadas a otimização e finanças.

[4]Cibele Dunder (dunder@ime.usp.br) é Mestre em Matemática Aplicada pelo Instituto de Matemática da USP, e consultora em áreas ligadas a otimização e finanças.

[5]Fabio Nakano (nakano@ime.usp.br) é Mestre em Computação pelo Instituto de Matemática da USP, e sócio da Supremum Assessoria e Consultoria, que atua em áreas ligadas a Engenharia de Sistemas e Pesquisa Operacional.

[6]Marcelo Lauretto (lauretto@ime.usp.br) é Mestre em Computação pelo Instituto de Matemática da USP, e sócio da Supremum Assessoria e Consultoria, que atua em áreas ligadas a Engenharia de Sistemas e Pesquisa Operacional.

Conteúdo

Prefácio

Este é o livro texto do curso Otimização e Processos Estocásticos Aplicados à Economia e Finanças. Parte deste material é usado no curso Métodos de Otimização em Finanças, ministrado regularmente no Instituto de Matemática e Estatística da Universidade de São Paulo, desde 1993. Este cursi foi também oferecido no LI Seminário Brasileiro de Análise, realizado em Maio de 2000, em Florianópolis, Santa Catarina, e no XIX Congresso Nacional de Matemática Aplicada e Computacional, realizado pela SBMAC em 1996, em Goiânia. Parte deste material foi também utilizado em cursos oferecidos na FIPE - Fundação Instituto de Pesquisas Econômicas da FEA-USP - Faculdade de Economia, Administração e Contabilidade, e no ILA - Instituto de Logística da Aeronáutica.

Os capítulos 1 e 2 provêm uma introdução básica a programação Linera e Não Linear. O capítulos 3 e 4 estudam em detalhe o problema de programação quadrática e sua aplicação à teoria matemática da formação de portfólios. Os modelos abordados baseiam-se na caracterização da fronteira eficiente, do modelo de Markowitz e de suas propriedades. Não há maneira de computar a fronteira eficiente e deduzir suas características sem compreender um dos vários algoritmos de programação quadrática paramétrica. No capítulo 3 apresentamos um algoritmo didático, fundamentado por alguns fatos da teoria de programação linear, apresentados no capítulo 1. Este algoritmo, uma versão modificada de algoritmos de Cottle, Dantzig, Lemke e Wolfe, é muito mais sintético do que o algoritmo originalmente apresentado por Markowitz, e a demonstração de sua corretude muito mais fácil e curta. Todos os fatos deduzidos nestes capítulos são reapresentados (sem demonstração) quando necessários aos modelos financeiros dos capítulos subseqüentes; um leitor menos rigoroso pode omitir estas demonstrações numa primeira leitura. O capítulo 4 apre-

senta a pedra fundamental de toda a moderna teoria de finanças, o modelo de Markowitz, e uma série de modelos derivados, como os modelos de Tobin, Brennan e modelos de índices, bem como modelos de equilíbrio, como CAPM e APT.

Os capítulos 5 e 6 expõem os princípios básicos de Progamação Dinâmica. O capítulo 5, inclui Cadeias de Markov, nos dá uma alternativa simples para modelagem de problemas multiperíodo, exemplificadas pela precificação de contratos de derivativos e políticas de Scarf. O capítulo 6 expande estes conceitos introduzindo noções da teoria de Estimação e Controle. O capítulo 7 discute alguns métodos de Inteligência Artificial, destacando árvores de decisão. O capítulo 8 aborda fundos de pensão, análise atuarial, programação estocástica e outros modelos para gestão dos ativos de fundos com objetivos de longo prazo.

Os apêndices A e B são manuais resumidos da linguagem Matlab e do software CriticalPoint, que são amplamente utilizados no curso. O apêndice C apresenta resumidamente alguns conceitos de Álgebra Linear Computacional, o apêndice D conceitos de probabilidade, estatística e teoria da utilidade, e o apêndice E alguns códigos fonte de programas.

Material suplementar está à disposição no site *www.ime.usp.br/∼ jstern*

Capítulo 1

Programação Linear

No processo de modelagem de sistemas, invariavelmente deparamos com duas questões: se o modelo se ajusta adequadamente ao problema sendo analisado e se o modelo proposto é implementável sob o ponto de vista computacional. Algumas classes de modelos conseguem atingir um relativo equilíbrio entre estes dois aspectos, sendo aplicáveis a uma extensa categoria de problemas reais, facilmente implementáveis e fornecendo soluções robustas. Programação linear insere-se nesta categoria.

Em um problema de programação linear procura-se encontrar uma solução que maximize ou minimize um funcional linear, dentro de um conjunto descrito a partir de restrições lineares. Inúmeros problemas podem ser modelados desta forma, tanto em finanças quanto em engenharia, mas essencialmente a principal vantagem destes modelos reside nas propriedades decorrentes da sua estrutura, que possibilitam a construção de algoritmos bastante simples e eficientes.

Iremos explorar algumas características dos problemas de programação linear, apresentando um algoritmo para sua resolução denominado simplex. Este algoritmo destaca-se tanto por sua simplicidade quanto ampla utilização. Estudaremos ainda o conceito de dualidade e procuraremos apresentar um exemplo de aplicação em finanças.

1.1 Notação Matricial

Inicialmente definimos algumas notações matriciais. O operador $r:s:t$, lê-se - *de* r *até* t com *passo* s, indica o vetor $[r, r+s, r+2s, \ldots t]$ no correspondente domínio de íncices. $r:t$ é uma abreviação de $r:1:t$. Usualmente escrevemos uma matriz, A, como o índice de linha subscrito, e o índice de coluna superscrito. Assim, A_i^j é o elemento na i-ésima linha e j-ésima coluna da matriz A. Vetores de índices podem ser usados para montar uma matriz extraindo de uma matriz maior um determinado sub-conjunto de linhas e colunas. Por exemplo $A_{1:m/2}^{n/2:n}$ é o bloco nordeste, i.e. o bloco com as primeiras linhas e ultimas

colunas, de A. Alternativamente, podemos escrever uma matriz com índices de linha e coluna entre parenteses, i.e. podemos escrever o bloco nordeste como $A(1\!:\!m/2, n/2\!:\!n)$.

Exemplo: Dadas as matrizes

$$A = \begin{bmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \\ 31 & 32 & 33 \end{bmatrix} \;,\quad r = \begin{bmatrix} 1 & 3 \end{bmatrix} \;,\quad s = \begin{bmatrix} 3 & 1 & 2 \end{bmatrix} \;,$$

$$A^s_r = \begin{bmatrix} 13 & 11 & 12 \\ 33 & 31 & 32 \end{bmatrix} \;.$$

$V > 0$ é uma matriz positiva definida. O operador diag, quando aplicado a uma matriz quadrada, extrai o vetor na diagonal principal, e quando aplicado a um vetor, produz a matriz diagonal correspondente.

$$\mathrm{diag}(A) = \begin{bmatrix} A^1_1 \\ A^2_2 \\ \vdots \\ A^n_n \end{bmatrix} \quad,\quad \mathrm{diag}(a) = \begin{bmatrix} a_1 & 0 & \dots & 0 \\ 0 & a_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_n \end{bmatrix}$$

Uma lista de matrizes pode ser indexada por índices subscritos ou superscritos à esquerda. No caso de matrizes blocadas, estes indices à esquerda indicam os blocos de linhas (subscritos) e colunas (superscritos), como por exemplo na matriz

$$A = \begin{bmatrix} {}^1_1A & {}^2_1A & \dots & {}^s_1A \\ {}^1_2A & {}^2_2A & \dots & {}^s_2A \\ \vdots & \vdots & \ddots & \vdots \\ {}^1_rA & {}^2_rA & \dots & {}^s_rA \end{bmatrix}$$

Assim, ${}^s_rA^j_i$ é o elemento na i-ésima linha e j-ésima coluna do bloco situado no r-ésimo bloco de linhas e s-ésimo bloco de colunas da matriz A. Alternativamente, podemos escrever os índices de bloco entre chaves, i.e. podemos escrever ${}^s_rA^j_i$ como $A\{r,s\}(i,j)$.

O operador Vec empilha as colunas da matriz argumento em um único vetor.
O produto de Kronecker (ou produto direto, ou tensorial), $\otimes$, é definido como segue:

$$\mathrm{Vec}(U^{1:n}) = \begin{bmatrix} u^1 \\ u^2 \\ \vdots \\ u^n \end{bmatrix} \quad,\quad A \otimes B = \begin{bmatrix} A^1_1B & A^2_1B & \dots & A^n_1B \\ A^1_2B & A^2_2B & \dots & A^n_2B \\ \vdots & \vdots & \ddots & \vdots \\ A^1_mB & A^2_mB & \dots & A^n_mB \end{bmatrix}$$

Uma *matriz de permutação* é uma matriz obtida pela permutação de linhas ou colunas na matriz identidade. Realizar, na matriz identidade, uma dada permutação de linhas, nos

fornece a correspondente matriz de permutação de linhas; Analogamente, uma permutação de colunas da identidade fornece a correspondente matriz de permutação de colunas.

Dada uma (matriz de) permutação de linhas, P e uma (matriz de) permutação de colunas, Q, o correspondente vetor de índices de linha (coluna) permutados são

$$p = (P \begin{bmatrix} 1 \\ 2 \\ \vdots \\ m \end{bmatrix})'$$

$$q = \begin{bmatrix} 1 & 2 & \dots & n \end{bmatrix} Q$$

Realizar uma permutação de linhas (de colunas) numa matriz qualquer A, de modo a obter a matriz permutada $\tilde{A}$, equivale a multiplicá-la, à esquerda (à direita), pela correspondente matriz de permutação de linhas (de colunas). Ademais, se p (q) é o correspondente vetor de índices de linha (de coluna) permutados,

$$\tilde{A}_i^j = (PA)_i^j = A_{p(i)}^j$$

$$\tilde{A}_i^j = (AQ)_i^j = A_i^{q(j)} \ .$$

Exemplo: Dadas as matrizes

$$A = \begin{bmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \\ 31 & 32 & 33 \end{bmatrix} \ , \quad P = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \ , \quad Q = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \ ,$$

$$p = q = \begin{bmatrix} 3 & 1 & 2 \end{bmatrix} \ , \quad PA = \begin{bmatrix} 31 & 32 & 33 \\ 11 & 12 & 13 \\ 21 & 22 & 23 \end{bmatrix} \ , \quad AQ = \begin{bmatrix} 13 & 11 & 12 \\ 23 & 21 & 22 \\ 33 & 31 & 32 \end{bmatrix} \ .$$

Uma matriz quadrada, A, é *simétrica* sse for igual a transposta, isto é, sse $A = A'$. Uma *permutação simétrica* de uma matriz quadrada A é uma permutação da forma $\tilde{A} = PAP'$, onde P é uma matriz de permutação.

Uma matriz quadrada, A, é *ortogonal* sse sua inversa for igual a sua transposta, isto é, sse $A^{-1} = A'$. (a) Matrizes de permutação são ortogonais. (b) Uma permutação simétrica de uma matriz simétrica é ainda uma matriz simétrica.

1.2 Convexidade

Um ponto $y(l)$ é *combinação convexa* de m pontos de $\mathcal{R}^n$, dados pelas colunas da matriz X $n \times m$, sse

$$\forall i \ , \quad y(l)_i = \sum_{j=1}^{m} l_j * X_i^j \ , \quad l_j \geq 0 \ \mid \ \sum_{j=1}^{m} l_j = 1$$

ou, equivalentemente, em notação matricial

$$y(l) = \sum_{i=1}^{m} l_i * X^j \ , \quad l_j \geq 0 \ \mid \ \sum_{j=1}^{m} l_j = 1$$

ou ainda, substituindo as somatórias por produtos internos,

$$y(l) = Xl \ , \ \ l \geq 0 \mid \mathbf{1}'l = 1$$

Em particular, um ponto $y(\lambda)$ é combinação convexa de 2 pontos, z e w, se

$$y(\lambda) = (1 - \lambda)z + \lambda w \ , \ \ \lambda \in [0,\ 1] \ .$$

Geometricamente, estes são os pontos no segmento de reta que vai de z a w.

Um conjunto, $C \in \mathcal{R}^n$, é *convexo* sse contiver qualquer combinação convexa de dois quaisquer de seus pontos. Um conjunto, $C \in \mathcal{R}^n$, é *limitado* sse a distância entre quaisquer dois de seus pontos é limitada:

$$\exists \delta \mid \forall x1,\ x2 \in C \ , \ \ ||x1 - x2|| \leq \delta$$

Um conjunto não limitado é dito *ilimitado*. As figuras 1.1 e 1.2 apresentam alguns exemplos de conjuntos conforme as definições acima.

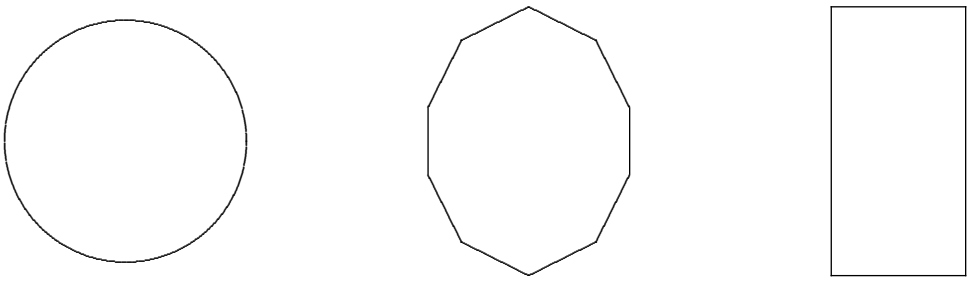


Figura 1.1: Conjuntos convexos

Se denomina Ponto Extremal de um conjunto convexo C, a todo ponto x que não pode ser representado como combinação convexa de dois pontos de C distintos de x. O Perfil de C, $ext(C)$, é o conjunto de todos os pontos extremais de C.

Se denomina Casco Convexo (Fechado) de um conjunto C, $ch(C)$ ($cch(C)$), á intersecção de todos os conjuntos convexos (fechados) que contem C.

Teorema: Um conjunto C, convexo e compacto, i.e. convexo, fechado e limitado, é igual ao casco convexo fechado de seu perfil, i.e. $C = cch(ext(C))$.

Teorema: O casco convexo de um conjunto finito de pontos, V, é o conjunto de todas as combinações convexas de pontos de V, i.e. se $V = \{x_i, i = 1 \ldots n\}$, então $ch(V) = \{x \mid x = [x_1, \ldots x_n]l, l \geq 0, \mathbf{1}'l = 1\}$.

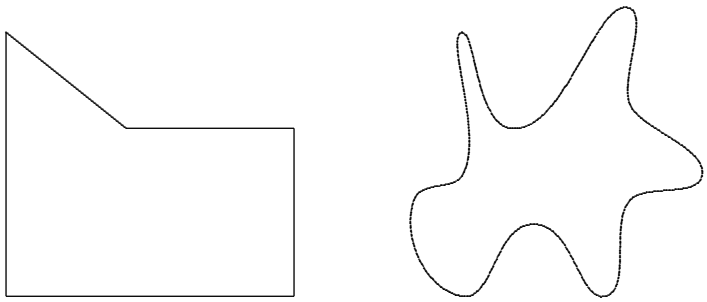


Figura 1.2: Conjuntos não convexos

O *epígrafo* de uma curva em $\mathcal{R}^2$, $y = f(x)$, $x \in [a, b]$, é o conjunto definido como $epig(f) \equiv \{(x, y) \mid x \in [a, b] \wedge y \geq f(x)\}$. Uma curva é convexa sse seu epígrafo é convexo. Uma curva, $f(x)$, é *côncava* sse $-f(x)$ for convexa.

Teorema: Uma curva, $y = f(x)$, $\mathcal{R} \mapsto \mathcal{R}$, continuamente diferenciável, e com primeira derivada sempre crescente, é convexa.

1.3 Poliedros

Uma *restrição* (não linear), em $\mathcal{R}^n$, é uma inequação da forma $g(x) \leq 0$, $g : \mathcal{R}^n \mapsto \mathcal{R}$. A *região viável* definida por m restrições, $g(x) \leq 0$, $g : \mathcal{R}^n \mapsto \mathcal{R}^m$, é o conjuntos dos *pontos viáveis* $\{x \mid g(x) \leq 0\}$. Dizemos que, num ponto viável x, a restrição $g_i(x)$ é *justa* ou *ativa* sse valer a igualdade ($g_i(x) = 0$), e *folgada* ou *inativa* caso contrário ($g_i(x) < 0$).

Um *poliedro* em $\mathcal{R}^n$ é uma região viável definida por *restrições lineares*: $Ax \leq d$. Podemos sempre compor uma restrição de igualdade, $a'x = \delta$, com um par de restrições de desigualdade, $a'x \leq \delta$ e $a'x \geq \delta$.

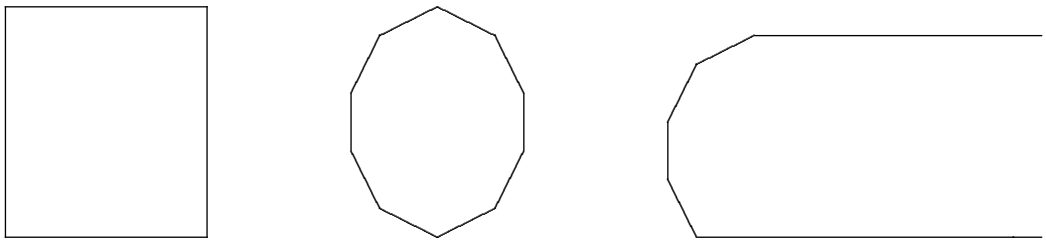


Figura 1.3: Poliedros

A figura 1.3 acima mostra alguns poliedros.

Teorema: Poliedros são convexos, mas não necessariamente limitados.

Faces

Uma *face* de dimensão k, de um poliedro em $\mathcal{R}^n$ com m restrições de igualdade, é uma região viável que obedece justamente a $n-m-k$ das restrições de desigualdade do poliedro. Em outras palavras: um ponto que obedece jstamente a r restrições de desigualdade está sobre uma face de dimensão $k = n - m - r$. Um *vértice* é uma face de dimensão 0. Um *lado* é uma face de dimensão 1, um *ponto interior* do poliedro tem todas as restrições de desigualdade folgadas.

Eventualmente poderíamos ter, num dado vértice, $n-m+1$ restrições de desigualdade justas. Este vértice seria super-determinado, pois obedeceria a $(n - m + 1) + m = n + 1$ equações. Todavia esta é uma situação instável: uma pequena perturbação nos dados levaria este vértice *degenerado* em 1 ou vários vértices não degenerados, vide figura 1.4 Doravante assumiremos sempre que situações como esta não ocorrem, i.e. assumiremos a *hipótese de não degenerescência*;

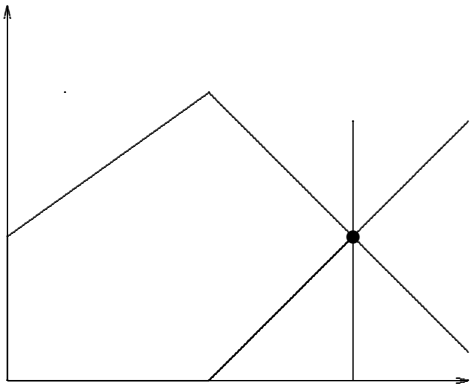


Figura 1.4: Vértice degenerado

Forma Padrão

Um poliedro na *forma padrão*, em $\mathcal{R}^n$, é dado pelas n *restrições de sinal* (i.e. $x_i \geq 0$) e $m < n$ *restrições de igualdade*:

$$P_A = \{x \geq 0 \ \mid \ Ax = d\} \quad , \ A \ m \times n$$

Podemos sempre reescrever um poliedro na forma padrão, ainda que num espaço de dimensão maior, usando os seguintes artifícios:

1. Substitua variáveis irrestritas, pela diferença de duas variáveis positivas, $x^+ - x^-$ onde $x^+ = max\{0, x\}$ e $x^- = max\{0, -x\}$.

2. Acrescente uma variável de folga $\chi \geq 0$ a cada inequação,

$$a'x \leq \delta \;\Leftrightarrow\; \left[\begin{array}{cc} a & 1 \end{array}\right] \left[\begin{array}{c} x \\ \chi \end{array}\right] = \delta \quad .$$

Da definição de vértice vemos que num poliedro padrão, P_A, um vértice é um ponto viável com $n - m$ restrições de sinal justas, i.e. $n - m$ variáveis nulas; estas são as *variáveis residuais* do vértice. Permutemos as linhas do vetor x de modo a colocar as variáveis residuais nas $n-m$ últimas posições, estando portanto as variáveis restantes, as *variáveis básicas*, nas m primeiras posições. Aplicando a mesma permutação às colunas de A, as primeiras m colunas da permutação de A formam a matriz B, ou *base* do vértice, e as demais colunas de A a *matriz residual*, R com dimensão $m \times (n - m)$. Isto é, particionamos as colunas de A como

$$A = \left[\begin{array}{cc} A^b & A^r \end{array}\right] = \left[\begin{array}{cc} B & R \end{array}\right]$$

Na forma blocada é fácil explicitar as variáveis não nulas. Reescrevendo $x \geq 0 \mid Ax = d$ como

$$\left[\begin{array}{c} x_b \\ x_r \end{array}\right] \geq 0 \mid \left[\begin{array}{cc} B & R \end{array}\right] \left[\begin{array}{c} x_b \\ x_r \end{array}\right] = d$$

temos

$$x_b = B^{-1}\left[d - Rx_r\right]$$

Anulando as variáveis residuais, segue que

$$x_b = B^{-1}d \quad .$$

Da definição de degenerecência vemos que um vértice de um poliedro padrão é degenerado sse tem, além das variáveis residuais, ao menos uma variável básica nula.

1.4 Método Simplex

O problema de programação linear (PPL) padrão é o de minimizar uma função linear dentro de um poliedro padrão:

$$\min \;\; cx, x \geq 0 \mid Ax = d \quad .$$

Suponha sabermos quais as variáveis residuais (i.e. as restrições justas) de um dado vértice, e consequentemente conhecida uma matriz básica B. Permutando as linhas de x e as colunas de c e A, para a forma blocada de variáveis básicas e residuais, reescrevemos o PPL como:

$$\min \quad \left[\begin{array}{cc} c^b & c^r \end{array}\right] \left[\begin{array}{c} x_b \\ x_r \end{array}\right] \;, \quad x \geq 0 \mid \left[\begin{array}{cc} B & R \end{array}\right] \left[\begin{array}{c} x_b \\ x_r \end{array}\right] = d \quad .$$

Utilizaremos a notação $\tilde{d} \equiv B^{-1}d$ e $\tilde{R} \equiv B^{-1}R$. Se mudarmos um único elemento, o j-ésimo de x_r, permitindo que ele se torne positivo (i.e. $x_{r(j)} > 0$), a *solução básica*, x_b se escreve como

$$\begin{aligned} x_b &= \tilde{d} - \tilde{R}x_r \\ &= \tilde{d} - x_{r(j)}\tilde{R}^j \end{aligned}$$

Esta solução permanece viável enquanto não negativa. Pela hipótese de não degenerescência, $\tilde{d} > 0$, e podemos aumentar o valor de $x_{r(j)}$, mantendo a solução básica viável, até um patamar $\epsilon > 0$, quando alguma variável básica é anulada.

O valor da função objetivo desta solução básica é

$$\begin{aligned} cx &= c^b x_b + c^r x_r \\ &= c^b B^{-1}[d - Rx_r] + c^r x_r \\ &= c^b\tilde{d} + (c^r - c^b\tilde{R})x_r \\ &\equiv \varphi - zx_r \\ &= \varphi - z^j x_{r(j)} \end{aligned}$$

O vetor z é denominado o *custo reduzido* na base.

Os parágrafos precedentes sugerem o seguinte algoritmo para gerar uma sequência de vértices viáveis de valores decrescentes, a partir de um vértice inicial:

Algoritmo Simplex:

1. Procure um índice residual, j, tal que $z^j > 0$.
2. Compute, para $k \in K \equiv \{l \mid \tilde{R}^j_l > 0\}$, $\epsilon_k = \tilde{d}_k/\tilde{R}^j_k$, e $i = \text{Argmin}_{k\in K}\ \epsilon_k$, i.e. $\epsilon(i) = \min_k \epsilon_k$.
3. Faça a variável $x_{r(j)}$ básica, e $x_{b(i)}$ residual.
4. Volte ao passo 1.

O Simplex não pode prosseguir se $z \leq 0$ no primeiro passo, ou se no segundo passo o mínimo for tomado sobre um conjunto vazio. O segundo caso corresponde a termos um PPL ilimitado. No primeiro caso o vértice corrente é a solução ótima do PPL!

Trocar o status básica/residual de um par de variáveis é denominado *pivotar*. Após cada pivotamento necessitamos recomputar a inversa da base, B^{-1}, o que pode ser feito com um mínimo de esforço.

No restante desta seção usaremos o simplex para resolver alguns problemas simples, interpretando geometricamente o processo de solução. Na próxima seção provaremos formalmente a corretude do algoritmo simplex.

Exemplo

Consideremos o PPL $\min[-1,-1]x,\ 0 \leq x \leq 1$.
Este PPL pode ser reescrito na forma padrão com

$$c = \left[\begin{array}{cccc} -1 & -1 & 0 & 0 \end{array}\right] \quad A = \left[\begin{array}{cccc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{array}\right] \quad d = \left[\begin{array}{c} 1 \\ 1 \end{array}\right]$$

Nos é dado um vertice inicial, $x = [0,0]$.
Passo 1: $r = [1,2]$, $b = [3,4]$, $B = A(:,b) = I$, $R = A(:,r) = I$,
$-z = c^r - c^b\tilde{R} = [-1,-1] - [0,0] \Rightarrow z = [1,1]$, $j = 1$, $r(j) = 1$,
$x_b = \tilde{d} - \epsilon\tilde{R}^j = \left[\begin{array}{c} 1 \\ 1 \end{array}\right] - \epsilon \left[\begin{array}{c} 1 \\ 0 \end{array}\right] \Rightarrow \epsilon^* = 1,\ i = 1,\ b(i) = 3$
Passo 2: $r = [3,2]$, $b = [1,4]$, $B = A(:,b) = I$, $R = A(:,r) = I$,
$-z = c^r - c^b\tilde{R} = [0,-1] - [-1,0] \Rightarrow z = [-1,1]$, $j = 2$, $r(j) = 2$,
$x_b = \tilde{d} - \epsilon\tilde{R}^j = \left[\begin{array}{c} 1 \\ 1 \end{array}\right] - \epsilon \left[\begin{array}{c} 0 \\ 1 \end{array}\right] \Rightarrow \epsilon^* = 1,\ i = 2,\ b(i) = 4$
Passo 3: $r = [3,4]$, $b = [1,2]$, $B = A(:,b) = I$, $R = A(:,r) = I$,
$-z = c^r - c^b\tilde{R} = [0,0] - [-1,-1] \Rightarrow z = [-1,-1] < 0$

Obtendo um Vértice Viável

Usaremos o PPL auxiliar

$$\min \left[\begin{array}{cc} 0 & 1 \end{array}\right] \left[\begin{array}{c} x \\ y \end{array}\right] \quad | \quad \left[\begin{array}{c} x \\ y \end{array}\right] \geq 0 \ \wedge \ \left[\begin{array}{cc} A & diag(sign(d)) \end{array}\right] \left[\begin{array}{c} x \\ y \end{array}\right] = d$$

Um vértice inicial para este PPL é $\left[\begin{array}{cc} 0 & abs(d') \end{array}\right]'$. Se o problema auxiliar tiver valor ótimo zero, a solução ótima fornece um vértice viável para o problema original; caso contrário, o problema original é inviável.

1.5 Dualidade

Dado um PPL qualquer, que chamaremos de PPL *Primal*, definiremos um outro PPL, o PPL *Dual* (do PPL primal). A teoria de dualidade trata das relações entre a solução de um dado PPL, e a solução do seu dual.

Dado um PPL na forma canônica, o problema primal (PPLP):

$$\min \ \ cx \mid x \geq 0 \wedge Ax \geq d \ \ ,$$

definimos seu dual como (PPLD):

$$\max \ \ y'd \mid y \geq 0 \wedge y'A \leq c \ \ ,$$

O primal canônico e seu dual tem uma interpretação econômica intuitiva: O primal pode ser interpretado como o clásico problema da ração: A_i^j é a quantidade de nutriente do tipo j encontrada em uma unidade de alimento do tipo i. c^i é o custo de uma unidade de alimento do tipo i, e d_j a necessidade diária mínima do nutriente j. A solução ótima do primal, x^* nos fornece a ração nutricionalmente viável de custo mínimo. O dual pode ser interpretado como um fabricante de nutrientes sintéticos, procurando o "valor de mercado" para sua linha de nutrientes. A receita do fabricante por ração é a função objetivo do dual a ser maximizada. Para manter sua linha de produtos competitiva, nenhum alimento natural deve fornecer nutrientes mais barato que a correspondente mistura de nutrientes sintéticos, estas são as restrições do dual. Os preços dos nutrientes, y^* podem também ser interpretados como preços marginais: (dentro de um pequeno intervalo) de quanto aumentaria o valor do alimento i se nele conseguissemos incrementar a concentração do nutriente j. A corretude destas interpretações é demostrada pelas propriedades de dualidade discutidas a seguir.

Lema 1 O dual do dual é o próprio primal.

Prova: Basta observar que o (PPLD) é equivalente a

$$\min \quad -y'd \mid y \geq 0 \wedge -y'A \geq -c \quad ,$$

Teorema Fraco de Dualidade Se x e y são soluções viáveis, respectivamente do primal e do dual, então existe um intervalo (gap) não negativo entre o valor da solução dual e o valor da solução primal:

$$cx \geq y'd \quad .$$

Prova: Por viabilidade, $Ax \geq d$ e $y \geq 0$, portanto $y'Ax \geq y'd$. Analogamente, $y'A \leq c$ e $x \geq 0$ donde $y'Ax \leq cx$; Portanto $cx \geq y'd$.

QED.

Corolário 1 Se tivermos um par de soluções viáveis, x para o PPLP, e y para o PPLD, e se o valor das soluções primal e dual coincidirem, i.e. $cx^* = (y^*)'d$, então ambas as soluções são ótimas.

Corolário 2

Se o primal for um problema ilimitado, então o dual é inviável.

Da mesma forma que podemos reescrever qualquer PPL na forma padrão, podemos reescrever qualquer PPL na forma da definição do primal ou dual. (Verifique). Em vista do lema 1, vemos que a relação de dualidade está definida entre pares de problemas de PL, independentemente da forma com que estão escritos.

Lema 2

Dado o primal na forma padrão (PPLP):

$$\min \;\; cx \mid x \geq 0 \wedge Ax = d \;\;,$$

o seu dual é (PPLD):

$$\max \;\; y'd \mid y \in \mathcal{R}^m \wedge y'A \leq c \;\;,$$

Teorema (Prova de corretude do Simplex)

Provaremos que o simplex termina em um vértice ótimo. Na parada do Simplex tínhamos $z = -(c^r - c^b B^{-1} R) \leq 0$. Consideremos $y' = c^b B^{-1}$ como candidato a solução dual.

$$\begin{aligned}
&\left[\begin{array}{cc} c^b & c^r \end{array}\right] - y' \left[\begin{array}{cc} B & R \end{array}\right] \\
&= \left[\begin{array}{cc} c^b & c^r \end{array}\right] - c^b B^{-1} \left[\begin{array}{cc} B & R \end{array}\right] \\
&= \left[\begin{array}{cc} c^b & c^r \end{array}\right] - c^b \left[\begin{array}{cc} I & \tilde{R} \end{array}\right] \\
&= \left[\begin{array}{cc} c^b & c^r \end{array}\right] - \left[\begin{array}{cc} c^b & c^b \tilde{R} \end{array}\right] \\
&= \left[\begin{array}{cc} 0 & -z \end{array}\right] \geq 0
\end{aligned}$$

Portanto y é uma solução dual viável. Ademais, seu valor (como solução dual) é $y'd = c^b B^{-1} d = c^b \tilde{d} = \varphi$, e pelo corolário 1 ambas as soluções são ótimas.

Teorema Forte de Dualidade Se o problema primal for viável e limitado, assim será o dual; além disso, o valor das soluções ótimas do primal e do dual coincidem.

Prova: Construtiva, através do algoritmo simplex.

Teorema (Folgas Complementares)

Sejam x e y' soluções viáveis de um PPL padrão e de seu dual. Estas soluções são ótimas sse $w'x = 0$, onde $w = (c - y'A)$. Os vetores x e w representam as folgas nas restrições de desigualdade do PPLP e do PPLD. Como $x \geq 0$, $w \geq 0$ e $w'x = 0$, cada termo do produto escalar se anula, i.e. $w_j x_j = 0$, ou equivalentemente, se a j-ésima restrição de desigualdade é folgada no primal, então a correspondente restrição no dual deve ser justa, e vice-versa; dai o nome folgas complementares.

Prova: Se as soluções são ótimas, poderíamos tê-las obtido com o algoritmo Simplex. Como na prova de corretude do Simplex,

$$(c - y'A)x = \left[\begin{array}{cc} 0 & z \end{array}\right] \left[\begin{array}{c} x_b \\ 0 \end{array}\right] = 0$$

Se $(c - y'A)x = 0$, então $y'(Ax) = cx$, ou $y'd = cx$, e pelo primeiro corolário do teorema fraco de dualidade, ambas as soluções são ótimas.

Forma geral de Dualidade

Daremos a seguir o dual de um PPL em forma geral. Indicamos por um asterisco, $(*)$, um vetor com variáveis irrestritas.

PPLP:

$$\min \left[\begin{array}{ccc} {}^1c & {}^2c & {}^3c \end{array} \right] \left[\begin{array}{c} {}_1x \\ {}_2x \\ {}_3x \end{array} \right] \quad \begin{array}{ccc} {}_1x & \geq & 0 \\ {}_2x & * & \\ {}_3x & \leq & 0 \end{array} \quad | \quad \left[\begin{array}{ccc} {}^1_1A & {}^2_1A & {}^3_1A \\ {}^1_2A & {}^2_2A & {}^3_2A \\ {}^1_3A & {}^2_3A & {}^3_3A \end{array} \right] \left[\begin{array}{c} {}_1x \\ {}_2x \\ {}_3x \end{array} \right] \quad \begin{array}{ccc} \leq & {}_1d \\ = & {}_2d \\ \geq & {}_3d \end{array}$$

PPLD:

$$\max \left[\begin{array}{ccc} {}_1d' & {}_2d' & {}_3d' \end{array} \right] \left[\begin{array}{c} {}_1y \\ {}_2y \\ {}_3y \end{array} \right] \quad \begin{array}{ccc} {}_1y & \leq & 0 \\ {}_2y & * & \\ {}_3y & \geq & 0 \end{array} \quad | \quad \left[\begin{array}{ccc} {}^1_1A & {}^2_1A & {}^3_1A \\ {}^1_2A & {}^2_2A & {}^3_2A \\ {}^1_3A & {}^2_3A & {}^3_3A \end{array} \right]' \left[\begin{array}{c} {}_1y \\ {}_2y \\ {}_3y \end{array} \right] \quad \begin{array}{ccc} \leq & {}^1c' \\ = & {}^2c' \\ \geq & {}^3c' \end{array}$$

Como corolário, ilustremos alguns pares de problemas primal/dual

$$PPLP: \ \max cx \mid Ax \geq d \wedge x \geq 0 \quad PPLD: \ \min y'd \mid y'A \geq c \wedge y \leq 0$$

$$PPLP: \ \max cx \mid Ax \leq d \wedge x \in \mathcal{R}^n \quad PPLD: \ \max y'd \mid y'A = c \wedge y \leq 0$$

1.6 Problema Linear Paramétrico

Consideremos agora o Problema Linear Paramétrico (PLP), i.e. uma família de PPLs em função de um parâmetro, η. Como primeiro exemplo consideremos uma parametrização linear do vetor do lado direito,

$$\min cx \quad x \geq 0 \mid Ax = d \quad , \quad d = t + \eta p$$

Suponhamos ter resolvido o PPL acima, viável e limitado, para um certo valor $\eta = \eta'$.

Na base ótima B, a solução básica x_b pode ser escrita como

$$\tilde{d} = B^{-1}(t + \eta p) = \tilde{t} + \eta' \tilde{p}$$

Analisemos agora o comportamento deste problema quando variamos o parâmetro η, ou seja, consideremos o comportamento da solução do PPL para valores de η próximos de η'.

Consideremos η crescente: Se $\tilde{p} \geq 0$, podemos aumentar o parâmetro η sem jamais tornar $\tilde{d}$ inviável, i.e. sem jamais violar uma restrição de sinal. Caso contrário, podemos aumentar o parâmetro apenas até o valor crítico

$$\eta_j = -\tilde{t}_j / \tilde{p}_j \quad , \quad k = arg \min_{j | \tilde{p}_j < 0} \eta_j \quad , \quad \eta_1 = \eta_k \ .$$

Analogamente, consideremos o comportamento da solução para η decrescente. Se $\tilde{p} \leq 0$, podemos diminuir o parâmetro η sem jamais tornar $\tilde{d}$ inviável, i.e. sem jamais desobedecer uma restrição de sinal. Caso contrário, podemos diminuir o parâmetro apenas até o valor crítico

$$\eta_0 = \eta_k \ \ , \ \ \eta_j = -\tilde{t}_j/\tilde{p}_j \ \ , \quad k = arg \max_{j|\tilde{p}_j>0} \eta_j \ .$$

Para η entre este par de parâmetros críticos, $\eta_0 < \eta' < \eta_1$, $\eta(\lambda) = (1-\lambda)\eta_0 + \lambda\eta_1$, vemos que a solução do problema será a correspondente combinação convexa das soluções críticas:

$$x(\eta(\lambda)) = (1-\lambda)x(\eta_0) + \lambda x(\eta_1) \ .$$

Como $x(\eta)$ é linear entre um par de parâmetros críticos consecutivos, o valor ótimo da solução do PPL com parâmetro $\eta = (1-\lambda)\eta_0 + \lambda\eta_1$, $vopt(\eta)$, é uma função linear em η. Portanto o gráfico de $vopt(\eta)$, para $\eta_0 \leq \eta < \eta_1$, será um segmento de reta.

1.7 Modelo de Sharpe

Devido a simplicidade da estrutura dos problemas de programação linear, muitas vezes modelos não lineares são simplificados de forma a recair no caso linear. Este artifício tem sido bastante empregado em problemas de composição de portfólios, onde o objetivo é determinar uma carteira de ativos que minimize o risco e maximize o retorno do investidor.

Consideremos que o retorno de cada um dos ativos se escreva como

$$r_i = a_i + e_i + b_i(a_0 + e_0), \ i \in \{1 \dots n\}$$

onde a_0 é o retorno esperado "do mercado", e_0 um erro aleatório aferando este retorno, a_i é o retorno acima do mercado esperado do i-ésimo ativo, e_i um erro afetando somente o retorno deste ativo, e b_i uma medida da reação do ativo i às mudanças do índice de mercado. Admitamos ainda termos ruidos brancos não correlacionados, i.e. $E(e_i) = 0$, $Var(e_i) = s_{i,i} = s_i^2$, $Cov(e_i, e_j) = s_{i,j} = 0, \ i \neq j$. Em finanças é usual a notação α, β e σ para os vetores a, b e s.

Seja x_i a fração do total de recursos que será alocado para o ativo i, $x \geq 0$, $\mathbf{1}'x = 1$. O retorno do portfólio, r_P, se escreve como

$$r_p = r'x = (a+e)'x + (a_0 + e_0)b'x$$

definindo $r_P = r'x$, $a_P = a'x$, $b_P = b'x$, a esperânça e variância do portfólio serão:

$$E(r_p) = a_P + a_0 b_P \ , \ \ Var(r_p) = s_0^2 b_P^2 + \sum_1^N (s_i x_i)^2.$$

Definimos a função objetivo a ser maximizada, como o retorno esperado do portfólio poderado pela tolerância ao risco do investidor, η, menos o desvio padrão do retorno:

$$f(x) = \eta E(r_p)) - \text{std}(r_p).$$

Para linearizar o problema, assumimos que a carteira é bem diversificada, i.e. que $x_i \approx 1/n$, e que nenhum ativo tem risco próprio muito maior que o risco de mercado, de modo que

$$s_0^2\, b_P^2 >> \sum_1^N (s_i\, x_i)^2$$

Recaimos assim em um problema de programação linear, conhecido como o modelo de Sharpe:

$$max f(x) = \eta(a + a_0 b)'x \ - s_0 b'x \ , \ x \geq 0 \mid \mathbf{1}'x = 1$$

a restrição do problema é conhecida como "restrição normal", que simplesmente assegura a interpretação do vetor x como a fração investida em cada ativo. Outras restrições lineares podem ser acrescentadas para modelar condições impostas pelo investidor. Por exemplo, a condição de diversificação pode ser escrita como $Ix \leq (\kappa/n)\mathbf{1}$, $\kappa << n$.

1.8 Fluxos em Redes

Muitos problemas importantes podem ser formulados como o problema de encontrar o fluxo de minimo custo que satisfaz as equações de oferta e demanda em uma rede. A Rede (ou Grafo) é descrita pelos seus Vértices (ou Nós), numerados de 1 a n, e pela Matriz de Incidência de seus Arcos, A. Cada arco (i, j), corresponde a uma coluna de A, A^{ij}, tal que $A_i^{ij} = -1$, $A_j^{ij} = +1$ e $A_l^{ij} = 0$, $l \neq i, j$. A demanda da mercadoria (ou commoditie) é representada pelo Vetor de demanda, b. Uma demanda negativa, $b_i < 0$, representa uma oferta, ou disponibilidade da mercadoria no vertice i. Imporemos a condição $\mathbf{1}'b = 0$ i.e., a oferta total iguala a demanda total. O fluxo de (mercadoria de) i para j é x_{ij}. O custo de enviar uma unidade (de mercadoria) de i para j é dado pelo Vetor de Custos, c^{ij}. Podemos ainda impor Restrições de Capacidade sobre os arcos, $l \leq x \leq k$. Finalmente, o problema de Fluxo Capacitado de Mínimo Custo pode ser formulado como:

$$\min cx \ , \ l \leq x \leq k \ \mid \ Ax = b$$

Note que como a demanda total é nula, e o fluxo é conservativo, poderiamos eliminar uma qualquer das equações do sistema.

Um fluxo com $b = 0$ denomina-se uma Circulação. É facil transformar o problema de fluxo em um problema de circulação capacitado. Construimos a Rede de Circulação acrescendo à rede original:

- Dois vértices especiais: a Entrada (ou Fonte), 1, e a Saída (ou Sorvedouro, ou Dreno), n;

- Arcos da entrada aos vétices com oferta, e dos vértices com demanda à saída, com capacidades iguais à quantidade absoluta de oferta ou demanda:
  $(1, j)$, $l_{1j} = 0$, $k_{1j} = b_j^+$, $c^{1j} = 0$,
  (j, n), $l_{jn} = 0$, $k_{jn} = (-b_j)^+$, $c^{jn} = 0$;

- O arco conectando a saída à entrada:
  $(n, 1)$, $l_{n1} = \beta = \sum b_j^+$, $k_{n1} = +\infty$, $c^{n1} = 0$.

Antes de estudar a resolução do problema de circulação capacitada de mínimo custo, enunciaremos um lema auxiliar:

Teorma de Hoffman: Uma rede de circulação capacitada tem uma circulação viável sse $l(V, \bar{V}) \leq k(\bar{V}, V)$, $\forall\, V \subset N$. Como o fluxo é conservativo, a condição é necessária. Verificaremos que a condição é suficiente, ao provar a corretude do algoritmao de calibragem.

Formulemos agora o problema de circulação capacitada de custo mínimo, como um problema de programação linear.

Usando variáveis de folga primais dadas pela distância às restrições de capacidade, o problema, seu dual, e a condição de folgas complementares, podem ser escritos na forma padrão:

$$\begin{aligned}
(P)\ \min \bar{c}\,\bar{x} \qquad & \bar{x} \geq 0 \ \mid\ \bar{A}\bar{x} = \bar{b} \\
\bar{x} &= [x; \dot{x}; \ddot{x}] \\
\bar{c} &= [c, 0, 0] \\
\bar{b} &= [0; l; k] \\
\bar{A} &= \begin{bmatrix} A & 0 & 0 \\ I & -I & 0 \\ I & 0 & I \end{bmatrix} \\
(D)\ \max \bar{y}\,\bar{b} \qquad & \bar{y}\bar{A} \leq \bar{c} \\
\bar{y} &= [y, \dot{y}, \ddot{y}] \\
\text{Folg. Comp.} \qquad & \bar{x}_p\,(\bar{c} - \bar{y}\,\bar{A})^p = 0 \ \Leftrightarrow
\end{aligned}$$

$$x_{ij}(c^{ij} - (-y^i + y^j + \dot{y}^{ij} + \ddot{y}^{ij})) = 0\ ,\ \dot{x}_{ij}\dot{y}^{ij} = 0\ ,\ \ddot{x}_{ij}\ddot{y}^{ij} = 0; \Leftrightarrow$$
$$x_{ij}(d^{ij} - \dot{y}^{ij} - \ddot{y}^{ij}) = 0\ ,\ (x_{ij} - l_{ij})\dot{y}^{ij} = 0\ ,\ (k_{ij} - x_{ij})\ddot{y}^{ij} = 0.$$

As variáveis de folga para as restrições de capacidade são $\dot{x}$ e $\ddot{x}$. As variáveis y^j são denominadas potencial, e interpretadas como o preço de mercado da commoditie na praça j. Definimos o differencial de custo, ou diferença de potencial, $d^{ij} = c^{ij} + y^i - y^j$. Um

diferencial de custo negativo representa uma possibilidade de arbitragem (lucro): basta comprar o produto na praça i, transporta-lo, e vende-lo na praça j. Esta interpretação torna natural que o fluxo $l_{ij} \leq x_{ij} \leq k_{ij}$, $0 < l_{ij} < k_{ij}$, em um arco obedecendo às condições de viabilidade primal, viabilidade dual e folgas complementares deva estar em uma das três codições seguintes:

L: $x_{ij} = l_{ij}$, $\ddot{y}^{ij} = 0$, $\dot{y}^{ij} = d^{ij} > 0$.

M: $l_{ij} \leq x_{ij} \leq k_{ij}$, $\dot{y}^{ij} = \ddot{y}^{ij} = d^{ij} = 0$.

K: $x_{ij} = k_{ij}$, $\dot{y}^{ij} = 0$, $\ddot{y}^{ij} = d^{ij} < 0$.

ou

L: $x_{ij} = l_{ij}$ e $d^{ij} \geq 0$.

M: $l_{ij} < x_{ij} < k_{ij}$ e $d^{ij} = 0$.

K: $x_{ij} = k_{ij}$ e $d^{ij} \leq 0$.

$$\dot{y}^{ij} = \max(0, d^{ij}) \ , \ \ddot{y}^{ij} = \min(0, d^{ij}) \ .$$

O algoritmo de calibragem (out-of-kilter) resolve o problema de circulação capacitada de custo mínimo com uma abordagem primal-dual. Nesta abordagem, iniciamos com uma circulação e potenciais quaisquer, e, a cada passo do algoritmo, procuraremos aproximar a circulação da viabilidade e condições de folgas complementares. Assumiremos que os dados do problema, c, l e k, são inteiros.

A seguir tabelamos as condições possíveis de uma arco descalibrado em uma circulação, e algumas quantidades associadas:

Cond.	d_{ij}	x_{ij}	desvio	Cap.Arco
L'	> 0	$< l_{ij}$	$l_{ij} - x_{ij}$	$h_{ij} = l_{ij} - x_{ij}$
L"	> 0	$> l_{ij}$	$d^{ij}(x_{ij} - l_{ij})$	$h_{ji} = x_{ij} - l_{ij}$
M'	$= 0$	$< l_{ij}$	$l_{ij} - x_{ij}$	$h_{ij} = l_{ij} - x_{ij}$
M"	$= 0$	$> k_{ij}$	$x_{ij} - k_{ij}$	$h_{ji} = x_{ij} - k_{ij}$
K'	< 0	$< k_{ij}$	$d^{ij}(x_{ij} - k_{ij})$	$h_{ij} = k_{ij} - x_{ij}$
K"	< 0	$> k_{ij}$	$x_{ij} - k_{ij}$	$h_{ji} = x_{ij} - k_{ij}$

O desvio total, em relação a "situação econômica de equilibrio", é a somatoria termos em arcos de dois tipos, correspondento a:
(1) Volume de fluxo violando restrições de capacidade, nas condições L', M', M" e K".
(2) Custos de fluxos anti-econômicos em fluxos acima do limite inferior com diferencial

de custo positivo, na condção L", e perdas de oportunidade em fluxos abaixo do limite superior com deferencial de custo negativo, na condição K'.

Dada uma circulação inviável ou não ótima, encontraremos uma nova circulação que mantenha as condições de folgas complementares, e que esteja mais proxima da viabilidade. A nova circulação é obtida pela adição de uma circulação em um circuito no grafo auxiliar, H.

Construimos H com arcos descalibrados, tipos (1) e (2) acima, com capacidades máximas h_{ij} na tabela. Acrescentamos ainda a H arcos correspondendo ao tipo
(3) Arcos calibrados na condição M, com capacidades $h_{ij} = k_{ij} - x_{ij}$ e $h_{ji} = x_{ij} - l_{ij}$.

Escolhido um arco descalibrado, (i, j), procuramos, por busca em profundidade em H, um caminho de j a i. Se um caminho for encontrado, a circulação com volume igual ao arco de mínima capacidade h_{ij} no circuito, pode ser somada à circulação inicial, reduzindo seu desvio total.

Caso contrario, seja J o conjunto dos vérices alcançados na busca a partir de j. Seja

$$\begin{aligned} S &= \{(v,w) \in G \mid v \in J, w \in \bar{J}, d^{vw} > 0, l_{vw} \leq x_{vw} < k_{vw}\} \\ S' &= \{(v,w) \in G \mid v \in \bar{J}, w \in J, d^{vw} < 0, l_{vw} < x_{vw} \leq k_{vw}\} \\ \delta &= \min(|d^{vw}|),\ (v,w) \in S \cup S' \end{aligned}$$

Se $\delta < \infty$, podemos aumentar o potencial dos vértices em $\bar{J}$ de δ, atualizar H, e continuar a busca, repetindo ajustes de potencial até alcançar i. Caso contrário, encontramos um conjunto que viola a condição do teorema de Hoffman.

A cada passo do algoritmo de calibragem reduzimos o desvio total da circulação corrente. Assim, após um número finito de passos, ou verificaremos a inviabilidade do problema, exibindo um sub-conjunto de v'ertices que viola a condição de Hoffman, ou a circulação final será viavel, obedecendo também a condição de folgas complementares, i.e., a circulação final será ótima.

1.9 Métodos de Decomposição

Suponha termos um PPL na forma $\min cx \geq 0, Ax = b$ onde a matriz $A = \begin{bmatrix} \dot{A} \\ \ddot{A} \end{bmatrix}$, e o poliédro descrito por $\ddot{A}x = \ddot{b}$ tem uma estrutura "simples", enquanto $\dot{A}x = \dot{b}$ implica em umas "poucas" restrições adicionais que, infelizmente, complicam em muito o problema. Por exemplo: $\ddot{A}x = \ddot{b}$ é um problema de transporte, enquanto $\dot{A}x = \dot{b}$ impõe restrições de capacidades coletivas sobre conjuntos de arcos, ou $\ddot{A}x = \ddot{b}$ um conjunto de PPL's para grupos de variáveis separadas, enquanto $\dot{A}x = \dot{b}$ impõe acoplamentos entre os PPL's (estrutura diagonal blocada por linha).

Estudaremos agora o método de Dantzig-Wolf, que nos permite resolver o PPL original, atravez de iterações entre um Problema Mestre pequeno, e um Subproblema simples. Suporemos que o poliédro simples é limitado, sendo igual a combinação convexa de seus vértices.

$$\ddot{X} = \{x \geq 0 \mid \ddot{A}x = \ddot{b}\} = ch(V) = Vl \ , \ l \geq 0 \mid \mathbf{1}'l = 1$$

O PPL original equivale ao problema mestre:

$$M: \ \min cVl \ , \ l \geq 0 \mid \begin{bmatrix} \dot{A}V \\ \mathbf{1}' \end{bmatrix} l = \begin{bmatrix} \dot{b} \\ 1 \end{bmatrix}$$

obviamente esta é uma representação que tem interesse teórico, não sendo prático encontrar os muitos vértices de V. Uma dada base B é ótima sse

$$-z = [cV]_R - ([cV]_B B^{-1})R \equiv [cV]_R - [y, \gamma]R \geq 0$$

esta condição equivale a termos, para todo indice residual, j,

$$cV^j - [y, \gamma] \begin{bmatrix} \dot{A}V^j \\ 1 \end{bmatrix} \geq 0 \ , \text{ ou}$$

$$\gamma \leq cV^j - y\dot{A}V^j = (c - y\dot{A})V^j \ , \text{ ou}$$
$$\gamma \leq \min(c - y\dot{A})v \ , \ v \in \ddot{X}$$

Definimos assim o sub-problema

$$S: \ \min(c - y\dot{A})v \ , \ v \geq 0 \ \mid \ \ddot{A}v = \ddot{b}$$

Se a solução ótima de S, v^* tem valor ótimo $(c - y\dot{A})v^* \geq \gamma$, a base B é ótima para M. Caso contrario, v^* nos a próxima coluna para entrar na base: $\begin{bmatrix} \dot{A}v^* \\ 1 \end{bmatrix}$.

A solução ótima do problema auxiliar nos fornece ainda um limite inferior para o problema original. Seja x uma solução viável qualquer do problema original, i.e. $x \in \ddot{X} \mid \dot{A}x = \dot{b}$. Sendo x mais restrito, $(c - y\dot{A})x \geq (c - y\dot{A})v^*$, portanto, $cx \geq y\dot{b} + (c - y\dot{A})v^*$. Note também que $y\dot{b}$ é o limite superior corrente. Note também que não é necessario ter um crescimento monotonico do limite inferior, sendo portanto nescessário guardar o melhor limite inferior já obtido.

Como vimos, a decomposição de Dantzig-Wolf adapta-se muito bem a problemas com certo tipo de estrutura, como a estrutura diagonal bolcada por linhas. Caso tivessemos uma estutura diagonal blocada por colunas, poderiamos decompor o problema dual. Este é, essencialmente, o método de decomposição de Benders, cuja implementação torna-se muito mais eficiente usando o algoritmo Simplex Dual, que estudaremos a seguir. Métodos de decomposição são muito eficientes para obter boas (ainda que não precisamente ótimas)

soluções para problemas estruturados. Algumas estruturas comumnmente encontradas na prática de Pesquisa Operacional incluem as formas Angular Blocada por Linhas (ABL), Angular Blocada por Colunas (ABC), Angular Blocada por Linhas e Colunas (ABLC), Blocada Escalonada (BEs), etc. Em algumas aplicações, como Programação Estocástica Multiperiodo, encontramos a forma recursiva (ou aninhada) destas estruturas. A figura Fig.x ilustra algumas destas formas. Para uma introdução intuitiva, formal e unificada aos diversos métodos de decomposição recomendamos os artigos de A.M.Geoffrion referidos na literatura. Para uma visão alternativa sobre decomposição de problemas de grande porte, focada em métodos computacionais de algebra linear para matrizes esparsas e estruturadas, veja Stern (1994).

1.10 Simplex Dual

O algoritmo simplex dual é análogo ao simplex, só que trabalha com uma base corrente dual viável, até atingir viabilidade primal. O simplex dual é muito útil em diversas situações em que resolvemos um PPL, e em seguida alteramos as restrições.

Trabalharemos com o PPL na forma padrão e seu dual:

$$P:\ \min cx\ ,\ x \geq 0\ Ax = d \quad \text{e} \quad D:\ \max y'd\ ,\ y'A \leq c$$

Em uma base dual viável, $y = c^b B^{-1}$ é uma solução dual, i.e.

$$\begin{aligned}
&\left[\begin{array}{cc} c^b & c^r \end{array}\right] - y' \left[\begin{array}{cc} B & R \end{array}\right] \\
&= \left[\begin{array}{cc} c^b & c^r \end{array}\right] - c^b B^{-1} \left[\begin{array}{cc} B & R \end{array}\right] \\
&= \left[\begin{array}{cc} c^b & c^r \end{array}\right] - c^b \left[\begin{array}{cc} I & \tilde{R} \end{array}\right] \\
&= \left[\begin{array}{cc} c^b & c^r \end{array}\right] - \left[\begin{array}{cc} c^b & c^b\tilde{R} \end{array}\right] \\
&= \left[\begin{array}{cc} 0 & -z \end{array}\right] \geq 0
\end{aligned}$$

Queremos agora reescrever o dual em uma forma analoga à forma padrão, acrescentando variáveis de folga, e usando a partição $A = [B, R]$, como segue:

$$\begin{aligned}
&\max d'y \quad A'y \leq c' \\
&\max d'y \quad \left[\begin{array}{c} B' \\ R' \end{array}\right] y \leq \left[\begin{array}{c} c^{b\prime} \\ c^{r\prime} \end{array}\right] \\
&\max d'y \quad \left[\begin{array}{ccc} B' & I & 0 \\ R' & 0 & I \end{array}\right] \left[\begin{array}{c} y \\ w_b \\ w_r \end{array}\right] = \left[\begin{array}{c} c^{b\prime} \\ c^{r\prime} \end{array}\right]\ ,\ w \geq 0
\end{aligned}$$

Nesta forma, a base dual, sua inversa e a correspondente solução básica são dadas por:

$$\left[\begin{array}{cc} B' & 0 \\ R' & I \end{array}\right]\ ,\ \left[\begin{array}{cc} B^{-t} & 0 \\ -R'B^{-t} & I \end{array}\right]$$

$$\begin{bmatrix} y \\ w_r \end{bmatrix} = \begin{bmatrix} B^{-t} & 0 \\ -R'B^{-t} & I \end{bmatrix} \begin{bmatrix} c^{b'} \\ c^{r'} \end{bmatrix} - \begin{bmatrix} B^{-t} & 0 \\ -R'B^{-t} & I \end{bmatrix} \begin{bmatrix} I \\ 0 \end{bmatrix} w_b \text{ i.e.}$$

$$y = B^{-t}c^{b'} - B^{-t}w_b \quad e \quad w_r = c^{r'} - R'B^{-t}c^{b'} + R'B^{-t}w_b$$

Note que os índices em b e r correspondem a índices básicos e residuais no primal, sendo a situação no dual complementar. Analogamente ao simplex na forma padrão, podemos aumentar um elemento nulo do vetor residual, para melhorar o valor da solução dual,

$$d'y = d'B^{-t}(c^{b'} - w_b) = \text{const} - \tilde{d}'w_b$$

Se $\tilde{d} \geq 0$ a solução básica primal é viável, e temos a solução ótima dual e primal. Se houver um elemento $\tilde{d}_i < 0$, podemos aumentar o valor da solução dual incrementando $w_{b(i)}$. Podemos incrementar $w_{b(i)} = \nu$, sem perder viabilidade dual, enquanto mantivermos

$$w_r = c^{r'} - R'B^{-t}c^{b'} + \nu R'B^{-t}I^i \geq 0 \text{ transpondo}$$

$$c^r - \tilde{c^b}R + \nu B_i^{-1}R \geq 0 \text{ i.e.}$$

$$-z + \nu\tilde{R}_i \geq 0$$

Fazendo $j = \arg\min\{\nu(j) = z^j/\tilde{R}_i^j \ , \ j \mid \tilde{R}_i^j < 0\}$, temos o índice que sai da base dual.

Podemos pois, na nova lista de índices básicos do primal, b, excluir $b(i)$, incluir $r(j)$, atualizar a base, e prosseguir iteragindo até a otimalidade dual, i.e. viabilidade primal.

1.11 Programação Inteira

Consiremos o PPL com restrições de caixa, onde algumas variáveis demem ser inteiras:
$\min cx \ , \ x \in N^p \times R^q \ , \ l \leq x \leq u \mid Ax = d.$
Se todas as variáveis são inteiras o problema é dito um PPLI Puro, caso contrário é dito um PPLI Mixto. A condição de integralidade destroi a convexidade do problema, dai sua grande dificuldade. Estudaremos o método de Ramificação e Poda (Brach and Bound, Cut, Prune), que pode ser aplicado a problemas relativamente grandes, mas com poucas restrições de integralidade.

Observemos inicialmente que, se $j < p$, sua solução ótima será a solução de um dos dois problemas seguintes, iguais ao PPLI original, a menos da restrição de caixa sobre a j-ésima variaável: 1- $l_j \leq x_j \leq h$, 2- $h+1 \leq x_j \leq u_j$, $h \in N$, $l_j \leq h \leq u_j - 1$. Desta forma formamos uma árvore de busca por sucessiva divisão (bifurcação, ramificação) do PPLI original em dois PPLIs menores. Em cada nó resolvemos o problema relaxado (PPL comum, sem as condiçẽes de integralidade). As follhas da árvore são as soluções que obedecem às condições de integridade. Note que pode ocorrer uma folha prematura, i.e. a solução ótima de um problema relaxado pode obedecer a todas as restrições de

integridade (antes destas serem impostas por cortes explicitos). A melhor folha da árvore (solução inteira) já encontrada, x^*, fornece um limite superior para a solução ótima do PPLI, cx^*.

Sejam $PPL(v)$, $x(v)$, e $cx(v)$ o PPL relaxado no nó v, sua solução ótima, e o valor ótimo. $cx(v)$ fornece um limite inferior para as solução que podem ser obtidas por futuras ramificações deste nó. Um nó inviável recebe valor $cx(v) = +\infty$. Sempre que $cx(v) \geq cx^*$, podemos eliminar (do processo de busca) todas as ramificações deste nó, i.e. podemos podar a árvore.

Se possível, terminamos o processo de ramificação quando encontramos uma solução garantidamente suficientemente próxima do ótimo. Note que mesmo após encontrar a solução ótima, x^* é possível que tenhamos que crescer muito a árvore até poder afirmar que x^* é de fato a solução ótima do PPLI. Note ainda que o Simplex dual pode ser de grande ajuda para resolver os PPLs criados por ramificação. A eficiência do método de ramificação e poda depende da heurítica de construção da árvore, i.e. do processo das sucessivas escolhas de nós, variáveis e pontos de corte, $PPL(v)$, x_j e h.

Seja uma variável básica, $x_j = \lfloor x_j \rfloor + f_j$, onde $\lfloor x_j \rfloor$ e f_j indicam as partes inteira e fracionária de x_j. Ao ramificar o nó v, na variaável x_j, seus dois filhos tem a restrição de caixa $l_j \leq x_j \leq u_j$, substituida por $l_j \leq x_j \leq h$ ou $h+1 \leq x_j \leq u_j$, onde o ponto de corte é $h = \lfloor x_j(v) \rfloor = x_j(v) - f_j$, ou $h + 1 = x_j(v) + (1 - f_j)$.

Uma heurística de escolha de variável é ramificar sempre a variável fracionária "mais importante". As prioridades poderiam vir de um vetor de pesos p, dados pela experiência com o problema. Poderiamos talves usar $p = |c|$. Definimos as penalidade associadas aos cortes "acima" e "abaixo" de $x_j(v)$: $P_j^l = f_j p_j$ e $P_j^u = (1 - f_j)p_j$. Estas penalidades pretendem prever o impacto dos cortes nos valores ótimos dos filhos de v, e nos permitem definir duas heurísticas para escolha da variável de corte:
H1: Ramifique em uma variável em $\arg\min_j \min\{P_j^l, P_j^u\}$. Esta heurística visa nos levar a uma "boa" solução inteira.
H2: Ramifique em uma variável em $\arg\max_j \max\{P_j^l, P_j^u\}$. Esta heurística visa criar um nó que será rapidamente podado, evitando um crescimento exagerado da árvore.

Analogamente, definimos duas heurísticas para escolha do nó a ser ramificado:
BFS: Breadth First Serach, ou BEL, Busca Em Largura. Dentre os nós existentes na árvore, ramifique o com melhor valor relaxado, $cx(v)$.
DFS: Depth First Search, ou BEP, Busca Em Profundidade. Dos nós, ainda não podados, criados na mais recente bifurcação, ramifique o com melhor valor relaxado, $cx(v)$.

1.12 Exercícios

1. Geometria e lemas simples:

   a- Desenhe o simplex, S_n, e o cubo, C_n de dimensão 2 e 3. $S = \{x \geq 0 \mid \mathbf{1}'x \leq 1\}$, $C = \{x \geq 0 \mid Ix \leq \mathbf{1}\}$.

   b- Reescreva S_2, S_3, C_2 e C_3 como poliedros padrão em R^n, onde $n = 3, 4, 4, 6$, respectivamente.

   c- Prove que um poliedro (na forma padrão) é convexo.

   d- Prove os lemas 1 e 2 de dualidade.

   e- Prove que um Poliédro limitado é igual ao conjunto das combinações convexas de seus vértices.

2. Escreva um programa para resolver um PPL na forma padrão por enumeração exaustiva de todos os vértices. Sugestão: Use o programa combina.m para enumerar todos os grupos de m colunas da matriz A, m por n, $n > n$. Em seguida forme B, a matriz quadrada com estas m colunas, verifique se B é uma base (é inversível), e se a solução básica é um vértice, $\tilde{d} = B^{-1}d > 0$. Calcule o valor das soluções básicas viáveis, e retorne a base ótima.

3. Adapte e implemente o Simplex para usar a fatoração QR, e atualizar, ao invés de reinverter, a base a cada passo do algoritmo. Incua uma forma de monitorar a qualidade da fatoração para, periodicamente reinverter a base, evitando a degradação da inversa pelo acúmulo de erro nas operações aritméticas.

4. Considere a solução ótima do PPL padrão com vetor de custo parametrizado: $c(\eta) = f + \eta g$.

   a- Quanto podemos aumentar (ou diminuir) o parâmetro η antes que a base que nos fornece a solução ótima mude ?

   b- Ao passar por um ponto de mudança de base (em função de aumento de η), i.e. ao passar por um η crítico, qual a próxima base ótima ?

   c- Conclua, usando o argumento de que existe um número finito de bases, que existe um número finito de etas críticos.

   d- Como poderíamos utilizar o dual do problema parametrizado no vetor de restrições para determinar a seqüência de seus etas (η) críticos ?

   e- Argumente que o valor ótimo deste PPL paramétrico, em função do parâmetro, é uma curva continua e côncava, linear por trechos i.e. composta de segmentos de reta.

   f- Adapte e implemente o Simplex para listar os etas e as soluções críticas, e implemente um programa auxiliar para calcular $x(\eta)$ por interpolação a partir da lista.

5. Adapte e implemente o Simplex para restrições de caixa, i.e.,
   $\min cx\ ,\ l \leq x \leq u\ \mid\ Ax = d.$

   Dica: Considere dada uma base viavel, B, e uma partição $\left[\begin{array}{ccc} B & R & S \end{array}\right]$, onde
   $l_b < x_b < u_b$, $x_r = l_r$, $x_s = u_s$, de modo que,
   $x_b = B^{-1}d - B^{-1}Rx_r - B^{-1}Sx_s$
   $cx = c^bB^{-1}d + (c^r - c^bB^{-1}R)x_r + (c^s - c^bB^{-1}S)x_s = \varphi + z^rx_r + z^sx_s$

   Se $z^{r(k)} < 0$, podemos melhorar a solução corrente aumentando a variavel residual no limite inferior, $x_{r(k)} = l_{r(k)} + \delta_{r(k)}$, $\quad x_b = B^{-1}d - B^{-1}Rl_r - B^{-1}Su_s - \delta_{r(k)}B^{-1}R^k$.
   Todavia, $\delta_{r(k)}$ deve respeitar os seguintes limites:
   1- $x_{r(k)} = l_{r(k)} + \delta_{r(k)} \leq u_{r(k)}$, $\quad$ 2- $x_b \geq l_b$, $\quad$ 3- $x_b \leq u_b$.
   Analogamente, se $z^{s(k)} > 0$, podemos melhorar a solução corrente diminuindo a variavel residual no limite superior, $x_{s(k)} = u_{s(k)} - \delta_{s(k)}$.

6. Faça um estudo empírico da complexidade esperada (flops) do Simplex para resolver um problema em função de: 1) Forma de atualização. 2) Critério de entrada.

6. Implemente o algoritmo de ajuste. Compare seu desempenho com o Simplex em problemas de circulação capacitada.

7. Adapte e implemente o Simplex Dual para problemas com restrições de caixa.

8. Implemente o algoritmo de poda e ramificação. Compare empiricamente diferentes heurísticas de seleção para o processo de ramificação.

9. Implemente o método de decomposição de Dantzig-Wolf para problema na forma diagonal blocada.

10. Implemente o método de decomposição de Benders para o problema de programação estocástica em dois estágios.

11. Prepare um seminário apresentando um modelo da biblioteca de exemplos do GAMS.

Capítulo 2

Programação Não Linear

2.1 GRG: Gradiente Reduzido Generalizado

Consideremos o problema de programação não linear com restrições não lineares de igualdade, alem de restrições de caixa sobre as variáveis,

$$\text{PPNL:} \quad \min f(x) \quad , \quad f : \mathcal{R}^n \mapsto \mathcal{R}$$

$$l \leq x \leq u \ \mid \ h(x) = 0 \quad , \quad h : \mathcal{R}^n \mapsto \mathcal{R}^m$$

O Método do Gradiente Reduzido Generalizado, ou GRG, imita os passos do Simplex, para uma linearização local do PPNL. Seja x um ponto viável inicial.

Como no Simplex, assumimos a hipótese de Não Degenerecência, i.e., assumimos que no máximo $(n - m)$ das restrições de caixa sejam ativas em um ponto viável. Assim, podemos tomar m das variáveis com as restrições de caixa folgadas como sendo as variáveis Básicas (ou dependentes), e as $n - m$ variáveis restantes como variáveis Residuais (ou independentes). Como no Simplex, particionamos todas as entidades vetoriais e matriciais reordenando as variáveis de modo a agrupar as variáveis básicas, x_b, e residuais, x_r,

$$x = \begin{bmatrix} x_b \\ x_r \end{bmatrix} \ , \ \ l = \begin{bmatrix} l_b \\ l_r \end{bmatrix} \ , \ \ u = \begin{bmatrix} u_b \\ u_r \end{bmatrix} \ , \ \ \nabla f(x) = \begin{bmatrix} \nabla^b f(x) & \nabla^r f(x) \end{bmatrix}$$

$$J(x) \ = \ \begin{bmatrix} J^B(x) & J^R(x) \end{bmatrix} \ = \ \begin{bmatrix} \nabla^b h_1(x) & \nabla^r h_1(x) \\ \nabla^b h_2(x) & \nabla^r h_2(x) \\ \vdots & \vdots \\ \nabla^b h_m(x) & \nabla^r h_m(x) \end{bmatrix}$$

Consideremos o efeito de uma pequena alteração no ponto viável corrente, $x + \delta$, assumindo que as funções f e h são contínuas e diferenciáveis. A correspondente alteração

no valor da solução será

$$\Delta f \;=\; f(x+\delta) - f(x) \;\approx\; \nabla f(x)\;\delta \;=\; \left[\begin{array}{cc} \nabla^b f(x) & \nabla^r f(x) \end{array}\right] \left[\begin{array}{c} \delta_b \\ \delta_r \end{array}\right]$$

Queremos também que o ponto alterado $x+\delta$ permaneça (aproximadamente) viável, i.e.,

$$\Delta h \;=\; h(x+\delta) - h(x) \;\approx\; J(x)\;\delta \;=\; \left[\begin{array}{cc} J^b(x) & J^r(x) \end{array}\right] \left[\begin{array}{c} \delta_b \\ \delta_r \end{array}\right] \;=\; 0$$

Isolando δ_b, e assumindo que a báse $J^b(x)$ é inversível,

$$\begin{array}{rcl} \delta_b & = & -\left(J^b(x)\right)^{-1}\; J^r(x)\;\delta_r \\ \Delta f & \approx & \nabla^b f(x)\;\delta_b + \nabla^r f(x)\;\delta_r \\ & = & \left(\nabla^r f(x) - \nabla^b f(x)\left(J^b(x)\right)^{-1}\; J^r(x)\right)\;\delta_r \;=\; z(x)\;\delta_r \end{array}$$

Como o problema não é linear, não podemos garantir que no ponto ótimo todas as variáveis residuais tem uma restrição de caixa justa, o análogo de um vértice no PPL padrão. Assim, não há interesse em restringir δ_r a ter apenas uma componente não nula, como no Simplex. Estas considerações sugerem mover o ponto viável corrente na direção (no espaço das variáveis residuais) do vetor v_r, oposta ao gradiente reduzido, sempre que a correspondente restrição de caixa for folgada, i.e.,

$$v_{r(i)} = \left\{\begin{array}{ll} -z^i & \text{se } z^i > 0 \text{ e } x_{r(i)} > l_{r(i)} \\ -z^i & \text{se } z^i < 0 \text{ e } x_{r(i)} < u_{r(i)} \\ 0 & \text{caso contrario} \end{array}\right.$$

Na próxima secção estudaremos condições gerais de convergência para algoritmos de PNL, e veremos que a descontinuidade do vetor v_r em função da folga nas restrições de caixa é indesejável. Assim, usaremos uma forma contínua do vetor de direção, por exemplo,

$$v_{r(i)} = \left\{\begin{array}{ll} -(x_{r(i)} - l_{r(i)})z^i & \text{se } z^i > 0 \text{ e } x_{r(i)} > l_{r(i)} \\ -(u_{r(i)} - x_{r(i)})z^i & \text{se } z^i < 0 \text{ e } x_{r(i)} < u_{r(i)} \\ 0 & \text{caso contrario} \end{array}\right.$$

A ideia do método GRG é a de, a cada iteração do algoritmo, movimentar o ponto viável, dando um passo $x+\delta$ com $\delta = \eta v$, onde $v_b = -\left(J^b(x)\right)^{-1}\; J^r(x)\; v_r$, i.e., um passo de "tamanho" η na direção (no espaço das variáveis residuais) v_r. Para determinar o tamanho do passo, η, é necessário fazer uma busca linear, respeitando as restrições de caixa. Note ainda que v_b, a direção no espaço das variáveis básicas, foi escolhido de modo a que $x+\eta v$, permaneça um ponto aproximadamente viável, pois estamos nos movendo no hiperplano tangente a superfície (o espaço tangente à variedade) determinada por $h(x) = 0$.

Um novo ponto x deverá ser sempre acompanhado de uma "correção" Δx para re-obter viabilidade exata nas restrições não lineares, $h(x+\Delta x) = 0$. O ponto x pode ser utilizado como ponto inicial de um método recursivo para encontrar um ponto exatamente viável. O método de Newton-Raphson consiste de considerar Jacobiano $J^b(x)$ para calcular a correção,

$$\Delta x_b = -\left(J^b(x)\right)^{-1}\ h(x_b, x_r)$$

2.2 Busca Linear e Convergência Local

Consideremos o problema de minimizar uma função, unidimensional, $f(x)$. Primeiramente, consideremos, como um problema auxiliar, o problema de encontrar a raiz (zero) de uma função derivavel, que aproximamos por sua série de Taylor de primeira ordem, $g(x) \approx q(x^k) + g'(x^k)(x - x^k)$. Esta aproximação implica que $g(x^{k+1}) \approx 0$, onde

$$x^{k+1} = x^k - g'(x^k)^{-1} g(x^k)$$

Este é o Método de Newton para encontrar a raiz de uma função unidimensional.

Mas se $f(x)$ é diferenciavel, encontrar seu ponto de mínimo implica encontrar um ponto onde se anula a derivada primeira, assim, escrevemos o Método de Newton para minimização de um função unidimensional,

$$x^{k+1} = x^k - f''(x^k)^{-1} f'(x^k)$$

Examinemos quão rapidamente a seqüência gerada pelo método de Newton se aproxima do ponto de mínimo, x^*, se a seqüência já estiver próxima do mesmo. Assumindo diferenciabilidade até terceira ordem, podemos escrever

$$0 = f'(x^*) = f'(x^k) + f''(x^k)(x^* - x^k) + (1/2) f'''(y^k)(x^* - x^k)^2\ ,\ \text{ou}$$

$$x^* = x^k - f''(x^k)^{-1} f'(x^k) - (1/2) f''(x^k)^{-1} f'''(y^k)(x^* - x^k)^2$$

Subtraindo da equação que define o método de Newton, temos

$$(x^{k+1} - x^*) = (1/2) f''(x^k)^{-1} f'''(y^k)\,(x^k - x^*)^2$$

Como veremos adiante, este resultado implica que o método de Newton converge rapidamente (quadraticamente) quando já próximo do ponto de ótimo. Todavia, o método de Newton necessita muita informação diferencial sobre a função, algo que pode ser difícil de obter. Ademais, longe do ponto de ótimo não temos nenhuma garantia sobre sua convergência. A seguir, estudaremos métodos que visam superar estas dificuldades.

Examinemos agora o método da Busca pela Razão Áurea para minimizar uma função unidimensional e unimodal, $f(x)$, em um intervalo, $[x^1, x^4]$. Se soubermos o valor da

função em quatro pontos, os estremos do intervalo e mais dois pontos interiores, $x^1 < x^2 < x^3 < x^4$, a unimodalidade da função garante podermos determinar que o ponto de mínimo, x^*, está em um subintervalo, i.e.,

$$f(x^2) \leq f(x^3) \Rightarrow x^* \in [x^1, x^3] \quad , \quad f(x^2) > f(x^3) \Rightarrow x^* \in [x^2, x^4] \quad .$$

Sem perda de generalidade consideremos a forma de dividir o intervalo $[0, 1]$. Uma razão r define uma divisão simétrica da forma $0 < 1-r < r < 1$. Dividindo o subintervalo $[0, r]$, pela mesma razão obtemos $0 < r(1-r) < r^2 < r$. Para que apenas uma nova avaliação da função seja necessária na próxima iteração, devem coincidir os pontos r^2 e $1-r$, i.e. $r^2 + r - 1 = 0$, de modo que $r = (\sqrt{5}-1)/2$, a razão áurea $r \approx 0.6180340$.

O método da razão áurea é robusto, funcionando para qualquer função unimodal, utilizando apenas o valor da função nos pontos de avaliação. No entanto, os extremos do intervalo de busca se aproximam apenas linearmente com o número de iterações. Métodos polinomiais, estudados a seguir, tentam conciliar as melhores características dos métodos já apresentados.

Métodos polinomiais para minimizar uma função unidimensional, $\min f(x+\eta)$, sobre $\eta \geq 0$, baseiam-se no ajuste de um polinômio que aproxima localmente $f(x)$, e a subseqüente minimização analítica do polinômio ajustado. O mais simples destes métodos é o do ajuste quadrático. Consideremos conhecidos três pontos, η_1, η_2, η_3, com o respectivo valor da função, $f_i = f(x+\eta_i)$. Considerando as equações do o polinômio interpolador

$$q(\eta) = a\eta^2 + b\eta + c \quad , \quad q(\eta_i) = f_i$$

obtemos as constantes do polinomio,

$$\begin{aligned}
a &= \frac{f_1(\eta_2-\eta_3) + f_2(\eta_3-\eta_1) + f_3(\eta_1-\eta_2)}{-(\eta_2-\eta_1)(\eta_3-\eta_2)(\eta_3-\eta_1)} \\
b &= \frac{f_1(\eta_3^2-\eta_2^2) + f_2(\eta_1^2-\eta_3^2) + f_3(\eta_2^2-\eta_1^2)}{-(\eta_2-\eta_1)(\eta_3-\eta_2)(\eta_3-\eta_1)} \\
c &= \frac{f_1(\eta_2^2\eta_3-\eta_3^2\eta_2) + f_2(\eta_3^2\eta_1-\eta_1^2\eta_3) + f_3(\eta_1^2\eta_2-\eta_2^2\eta_1)}{-(\eta_2-\eta_1)(\eta_3-\eta_2)(\eta_3-\eta_1)}
\end{aligned}$$

Anulando a derivada do polinômio interpolador, $q'(\eta_4) = 2a\eta + b$, obtemos seu ponto de mínimo, $\eta_4 = a/2b$, ou diretamente,

$$\eta_4 = \frac{1}{2}\frac{f_1(\eta_3^2-\eta_2^2) + f_2(\eta_1^2-\eta_3^2) + f_3(\eta_2^2-\eta_1^2)}{f_1(\eta_3-\eta_2) + f_2(\eta_1-\eta_3) + f_3(\eta_2-\eta_1)}$$

Procuraremos sempre utilizar os pontos iniciais no "padrão de interpolação", $\eta_1 < \eta_2 < \eta_3$ e $f_1 \geq f_2 \leq f_3$, i.e., três pontos onde o ponto intermediário tem o menor valor da função. Assim, garantimos que o mínimo do polinômio estará dentro do intervalo inicial de busca, $\eta_4 \in [\eta_1, \eta_3]$, ou seja, que estamos interpolando e não extrapolando a solução.

Escolhendo η_4 e mais dois dentre os três pontos iniciais, temos uma nova trinca no padrão desejado, e estamos prontos para uma nova iteração. Note ainda que, em geral, não podemos garantir que o quarto ponto será o melhor da nova trinca. Todavia, o quarto ponto sempre substitui um ponto da trinca antiga que tinha valor maior, de modo que a soma $z = f_1 + f_2 + f_3$ é monotonicamente decrescente no processo. Como veremos a seguir, estas características garantem a convergência global do algoritmo de busca linear por ajuste quadrático.

Consideremos agora os erros em relação ao ponto de mínimo, $\epsilon_i = x^* - x_i$. Podemos expressar $\epsilon_4 = g(\epsilon_1, \epsilon_2, \epsilon_3)$. A função g deve ser um polinômio de segundo grau, pois o ajuste feito é quadrático, e simétrico em seus argumentos, pois a ordem dos pontos é irrelevante. Ademais, não é difícil verificar que se ϵ_4 se anula se dois dos três erros iniciais se anulam. Portanto, na proximidade do ponto de mínimo, x^*, temos a seguinte aproximação do quarto erro:

$$\epsilon_4 = C\left(\epsilon_1\epsilon_2 + \epsilon_1\epsilon_3 + \epsilon_2\epsilon_3\right)$$

Assumindo a convergencia do processo, o k-ésimo erro será approximadamente $\epsilon_{k+4} = C\epsilon_{k+1}\epsilon_{k+2}$. Tomando $l_k = \log(C^{1/2}\epsilon_k)$, temos $l_{k+3} = l_{k+1} + l_k$, cuja equação característica é $\lambda^3 - \lambda - 1 = 0$. A maior raiz desta equação é $\lambda \approx 1.3$. Esta é a ordem de convergência local do processo, conforme definiremos a seguir.

Dizemos que uma seqüência de números reais $r^k \to r^*$ converge ao menos em ordem $p > 0$ se

$$0 \leq \lim_{k\to\infty} \frac{|r^{k+1} - r^*|}{|r^k - r^*|^p} = \beta < \infty$$

A ordem de convergência da seqüência é o supremo das constantes $p > 0$ nestas condições. Se $p = 1$ e $\beta < 1$, dizemos que a seqüência tem Convergência Linear com Taxa β. Se $\beta = 0$, dizemos que a seqüência tem Convergência Super-Linear.

Assim, para $c \geq 1$, c é a ordem de convergência da seqüência $a^{(c^k)}$. Vemos também que $1/k$ converge em ordem 1, embora não seja linearmente convergente, pois $r^{k+1}/r^k \to 1$. Finalmente, $(1/k)^k$ converge em ordem 1, pois para qualquer $p > 1$, $r^{k+1}/(r^k)^p \to \infty$, Todavia a convergência é super-liner, pois $r^{k+1}/r^k \to 0$.

2.3 Partan

Estudaremos agora o método da Tangentes Paralelas, ou Partan, para resolver o problema de minimizar uma função convexa irrestrita. Este método, como tantos outros, utiliza a idéia de modelar a função a ser minimizada por uma função quadrática que, sem perda de generalidade, suporemos centrada na origem, $f(x) = (1/2)x'Qx$.

Pela transformação linear $y = Ux$, onde U é o fator de Cholesky $U'U = Q$, podemos escrever f com simetria esférica,

$$f(y) = (1/2)(U^{-1}y)'Q(U^{-1}y) = (1/2)y'(U^{-t}(U'U)U^{-1})y = (1/2)y'Iy$$

Lembremos as definições de paralelismo e ortogonalidade para planos de dimensões arbitrarias: Uma reta é paralela a um plano se for paralela a uma reta deste plano. Uma reta é ortogonal a um plano se for ortogonal a toda reta deste plano. Um plano é paralelo a outro plano se toda reta deste plano for paralela ao outro plano. Um plano é ortogonal a outro plano se toda reta deste plano for ortogonal ao outro plano. Note que estas definições estão de acordo com as definições usuais de espaços vetoriais, mas são diferentes das definições usadas em geometria tridimensional.

Uma propriedade fundamental de uma transformação linear linear qualquer, $y = Ux$, é a de preservar as relações de Coplanaridade e Paralelismo, i.e.:

- Se os pontos $x^1 \dots x^k$ pertencem a um mesmo plano (reta), assim o é para os pontos $y^1 \dots y^k$, e vice-versa.

- Se os planos $\pi_{1,\dots k}$ e $\pi_{k+1,\dots k+h}$, determinados, no antigo sistema de coordenadas pelos pontos $x^1 \dots x^k$ e $x^{k+1} \dots x^{k+h}$, são paralelos, então estes mesmos planos, determinados no novo sistema de coordenadas pelos pontos $y^1 \dots y^k$ e $y^{k+1} \dots y^{k+h}$, também são paralelos.

Assim, se definirmos um bom algoritmo para o caso particular de funções esféricas, usando apenas relações de coplanaridade e paralelismo, este mesmo algoritmo permanecerá válido no caso geral. Este é o argumento básico no desenvolvimento do algoritmo Partan.

Lembremos ainda que a relação de ortogonalidade, $v'w = 0$, Não é preservada por uma transformação linear, pois se $v = Ux$ e $w = Uy$, então $v'w = (Ux)'(Uy) = x'(U'U)y = x'Qy$. Assim, utilizaremos a relação de Q-Conjugação, $x'Qy = 0$, ou simplesmente conjugação, se a forma quadrática Q estiver subentendida, como uma generalização do conceito de ortogonalidade.

Utilizaremos a seguinte notação: p^k para um ponto qualquer; q^k para o gradiente de f em p^k, i.e., $q^k = Qp^k$; π_k para o plano tangente a curva de nível de f passando por p^k, i.e., ao plano passando por p^k e ortogonal a q^k; e $\pi_{1,\dots k}$ para o (hiper) plano de dimensão $k-1$ determinado pelos pontos $p_1 \dots p_k$. Assim, $\pi_{1,2}$ é uma reta, $\pi_{1,2,3}$ é um plano bidimensional, etc. Finalmente, definimos os coeficientes de conjugação, $c_{k,j} \equiv (p^k)'Qp^j = (p^k)'q^j = c_{j,k}$.

Examinemos inicialmente o problema bidimensional, do ponto de vista geométrico.

Dado uma função quadrática bidimensional, $f(x) = (1/2)x'Qx$, que sem perda de generalidade suporemos centrada na origem, temos o seguinte método geométrico para encontrar o ponto de mínimo, p^*, veja figura 2.1a.

Algoritmo das Cordas Paralelas:

1) Ao longo de duas retas paralelas, r_0 e r_3, encontre os pontos de mínimo p^0 e p^3.

2) Ao longo da reta $\pi_{0,3}$, encontre o ponto de mínimo, p^4.

Pela simetria do problema esférico, fica claro que o centro p^* pertence a reta $\pi_{0,3}$, de modo que p^4 o ponto de mínimo em $\pi_{0,3}$, é o ponto de mínimo no plano, p^*.

Como p^0 e p^3 são pontos de mínimo de $f(x)$ nas retas r_0 e r_3, estas retas são também tangentes às respectivas curvas de nível de f passando por p^0 e p^3, or seja, são p^0 e p^3 determinam "tangentes paralelas". O algoritmo seguinte aproveita esta idéia. Uma reta passando por um ponto p^k é dita Degenerada se pertence ao plano π_k.

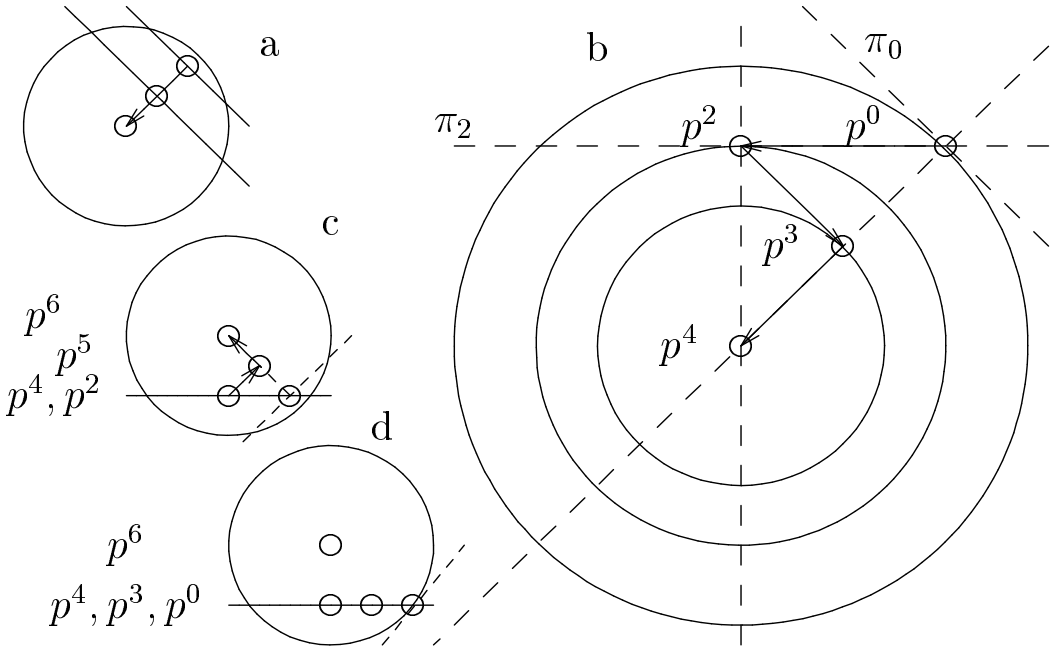


Figura 2.1 Partan na Esfera

Algoritmo Partan Bidimensional:

1) Tome um ponto inicial qualquer, p^0.

2) Tome p^2 o ponto de mínimo sobre uma reta por p^0 não degenerada.

3) Tome p^3 o ponto de mínimo na reta por p^2, paralela a π_0.

4) Tome p^4 o ponto de mínimo na reta por $p_{0,3}$.

Na figura 2.1b vemos que, para função esférica, este algoritmo nada mais é que uma forma particular de implementar o algoritmo das cordas paralelas, sendo portanto correto. Como as relações de coplanaridade e paralelismo (usadas para especificar o algoritmo) são invariantes por transformações lineares, vemos também que o algoritmo é correto para funções quadráticas quaisquer. Repare que os vetores $w^4 = p^4 - p^2$ e $w^2 = p^2 - p^0$ são ortogonais. Vamos agora estender este algoritmo para o espaço tridimensional.

Algoritmo Partan Tridimensional:

1) Tome um ponto inicial qualquer, p^0.

2) Tome p^2 o ponto de mínimo sobre uma reta por p^0 não degenerada.

3) Tome p^3 o ponto de mínimo em uma reta por p^2, paralela a π_0 e não degenerada.

4) Tome p^4 o ponto de mínimo na reta por $p_{0,3}$.

5) Tome p^5 o ponto de mínimo na reta por p^4 paralela a intersecção de π_0 e π_2.

6) Tome p^6 o ponto de mínimo na reta $\pi_{2,5}$.

Vejamos que p^6 é realmente o ponto de ótimo, olhando o problema no sistema de coordenadas com simetria esférica, veja a figura 2.1. Para tanto basta verificar que:
1) $p^5 - p^4 \;//\; \pi_0 \cap \pi_2$, 2) $p^* - p^0 \perp \pi_0$ e $p^* - p^2 \perp \pi_2$, 3) $p^5 - p^4 \perp \pi_{0,2,*}$,
4) $p^5 - p^4 \perp \pi_{0,2}$, 5) $p^5 - p^4 \in \pi_{2,4,*}$, 6) $p^6 = p^*$.

As afirmações acima são válidas: 1) por construão, 2) por símetria, 3) por 1 e 2, 4) pois $\pi_{0,2} \in \pi_{0,2,*}$, 5) por simetria, e 6) segue da corretude do Partan 2D.

Podemos ainda fazer algumas observações a respeito da geometria do Partan 3D com simetria esférica. Primeiro, vemos de p^{2k} é o ponto de mínimo do plano $\pi_{0,2,\ldots 2k}$. Segundo vemos de o vetor $w^{2k} \equiv p^{2k} - p^{2k-2}$ é perpendicular a este mesmo plano. Assim, os vetores w^{2k} são mutuamente ortogonais.

Estas observações são a báse para a generalização do Partan n-dimensional, e também para a prova de sua corretude.

Algoritmo Partan N-dimensional:

1) Tome um ponto inicial qualquer, p^0.

2) Tome p^2 o ponto de mínimo sobre uma reta por p^0 não degenerada.

3) Para $k = 1, 2, \ldots n-1$:

3a) Tome p^{2k+1} o ponto de mínimo em uma reta por p^{2k} não degenerada e paralela aos planos $\pi_0, \pi_2, \ldots \pi_{2k-2}$.

3b) Tome p^{2k+2} o ponto de mínimo na reta por p^{2k-2} e p^{2k+1}.

Teorema de Corretude do Partan n Dimensional: O algoritmo Partan n dimensional

encontra o ponto de mínimo $p^* = p^{2n}$.

Para provar o teorema de corretude, provemos o seguinte teorema,

Teorema de Conjugação do Partan: No algoritmo Partan n dimensional, os vetores $w^{2k} \equiv p^{2k} - p^{2k-2}$, $k = 1\!:\!n$, são mutuamente conjugados.

O teorema de conjugação do Partan implica imediatamente no teorema de corretudo do Partan, pois em n dimensões, não pode haver mais que n vetores não nulos Q-conjugados, com Q não singular. O teorema de conjugação decorre imediatamente do seguinte lema, devido a B.V.Shah, J.Buehler e O.Kempthorne.

Lema SBK: Para $k = 1\!:\!n, \quad c_{2k,0} = c_{2k,2} = \ldots = c_{2k,2k}$.

Para provar o Lema SBK, notemos inicialmente que as equações de construção dos pontos p^k do Partan implicam em várias identidades entre os coeficientes de conjugação:

(a) Pela condição de paralelismo, $(p^{2k+1} - p^{2k}) \perp q^{2j}$ ou, equivalentemente, para $k = 1\!:\!n-1$, $j = 0\!:\!k-1, \quad c_{2k+1,2j} = c_{2k,2j}$.

(b) Pela condição de convexidade, $p^{2k+2} = \lambda_{k+1} p^{2k+1} + \overline{\lambda}_{k+1} p^{2k-2}$, onde $\overline{\lambda} = (1-\lambda)$, e portanto, para $k = 1\!:\!n-1$, $j = 0\!:\!k-1, \quad c_{2k+2,j} = \lambda_{k+1} c_{2k+1,j} + \overline{\lambda}_{k+1} c_{2k-2,j}$.

(c) Pela condição de minimização, $(p^2 - p^0) \perp q^2 = 0$ e $(p^k - p^{k-1}) \perp q^k = 0$, ou para $k = 3\!:\!2n, \quad c_{2,2} = c_{2,0}$ e $c_{k,k} = c_{k,k-1}$.

(d) Pela condição de colinearidade de $p^{2k+2} - p^{2k+1}$ e $p^{2k+1} - p^{2k-2}$, para $k = 1\!:\!n-1$, $c_{2k+2,2k+2} = c_{2k+1,2k+2} = c_{2k-2,2k+2}$.

Por (c) o Lema SBK é verdadeiro para $k = 1$. Como hipótese indução, assumamos que o Lema SBK é verdadeiro até k, e provemos que é verdadeiro para $k+1$, i.e., que $c_{2k+2,0} = c_{2k+2,2} = \ldots = c_{2k+2,2k-2} = c_{2k+2,2k} = c_{2k+2,2k+2}$.

Usando (c), (b), (a), e novamente (c), temos a identidade (e):

$$c_{2k+2,2k+2} - c_{2k+2,2k} = c_{2k+2,2k+1} - c_{2k+2,2k} =$$

$$\lambda_{k+1}(c_{2k+1,2k+1} - c_{2k+1,2k}) + \overline{\lambda}_{k+1}(c_{2k-2,2k+1} - c_{2k-2,2k}) = 0$$

e, usando (d), temos a identidade (f):

$$c_{2k+2,2k+2} = c_{2k-2,2k+2}$$

De (e) e (f), temos as tres últimas igualdades do Lema SBK. As demais igualdades são obtidas tomando $j \leq k-1$, e escrevendo a identidade (g):

$$c_{2k+2,2k-2} - c_{2k+2,2j} = \lambda_{k+1}(c_{2k+1,2k-2} - c_{2k+1,2j}) + \overline{\lambda}_{k+1}(c_{2k-2,2k-2} - c_{2k-2,2j})$$

O fator da combinação convexa $\lambda_{k+1} = 1$, pela hipótese de indução. Usando (a) em ambos os termos com fator λ_{k+1} em (g), e usando novamente a hipótese de indução, temos

$$c_{2k+1,2k-2} - c_{2k+1,2j} = c_{2k,2k-2} - c_{2k,2j} = 0$$

de forma que (g) é zero, e o lema SBK é satisfeito para $2k+2$, Q.E.D.

Uma das observações que nos levaram ao Partan era que p^{2k} é o ponto de mínimo do plano $\pi_{0,2,\ldots 2k}$. Esta mesma observação nos fornece uma forma simples e eficiente de escolher a direção de $p^{2k+1} - p^{2k}$ respeitando a condição de paralelismo com os planos $\pi_0, \pi_2, \ldots \pi_{2k-2}$. Basta tomar (sempre) a direção $p^{2k+1} - p^{2k}$ perpendicular a π_{2k}, isto é, na direção do gradiente. Isto acontece pois são paralelos os três planos, π_{2k}, $\pi_{0,2,\ldots 2k}$ e $\pi'_{0,2,\ldots 2k}$, onde o último plano é o paralelo ao penúltimo passando pela origem. Todos os gradientes $q^0, q^2, \ldots q^{2k-2}$ estão em $\pi'_{0,2,\ldots 2k}$, de modo que tomar a direção do gradiente q^{2k} garante a perpendicularidade com os gradientes anteriores, $q^0, q^2, \ldots q^{2k-2}$ ou, equivalentemente, a condição de paralelismo. Este é o algoritmo Partan Gradiente.

Como Partan Gradiente é um caso particular do Partan, ele encontra a solução ótima de uma função quadrática em um numero finito $(2n)$ de passos. Todavia, mesmo se o modelo quadrático para a função objetivo for pobre, os passos impares são nada mais que minimizações na direção de máximo declive. Isto explica porque o Partan Gradiente é um algoritmo de otimização irrestrita com as melhores características de dois mundos, tendo a robustes do método de Cauchy quando longe do ponto de ótimo, e convergencia quadrática quando perto da solução ótima.

O Partan necessita de duas buscas lineares (passo impar e par) para cada dimensão do problema. Longe do ponto de ótimo o Método de Cauchy usaria apenas uma busca. Perto da solução ótima poderiamos usar também apenas uma busca linear por dimensão, caso soubessemos gerar facilmente as n direções conjugadas. Esta é a idéia básica do(s) algoritmo(s) tipo Gradientes Conjugados. Todavia este corte pela metade o número de buscas linears tem um custo implícito: A implementação de um mecanismo de monitoramento do comportamento do algoritmo, para decidir em que momento fazer a tranzição de Cauchy para Gradientes Conjugados.

2.4 Convergência Global

Estudaremos nesta seção as condições de convergência de um algoritmo para a solução ótima de um problema de otimização não linear. Seguiremos de perto a apresentação desenvolvida por W.I.Zangwill.

Definimos Algoritmo como um processo iterativo gerando uma seqüência de pontos, $x^0, x^1, x^2 \ldots$ que obedece a uma equação de recursão da forma $x^{k+1} \in A_k(x^k)$, onde o Mapa Ponto a Conjunto $A_k(x^k)$ define os possíveis sucessores de x^k na seqüência.

A idéia de usar um mapa ponto-a-congunto, ao invés de uma função, ou mapa ponto-a-ponto, nos permite estudar de forma unificada classes de algoritmos, incluindo diversas implementações de vários detalhes, cálculos aproximados ou inexatos, variáveis randomizadas, etc. A propriedade básica que queremos dos mapas definindo os algoritmos é a

propriedade de Fechamento, definida a seguir.

Um mapa ponto-a-conjunto do espaço X no espaço Y, é fechado em x na seguinte condição: Se a seqüência x^k converge para x, e a seqüência y^k converge para y, onde $y^k \in A(x)$, então o ponto limite na imagem, y, pertence a imagem por A do ponto limite no domínio, x, i.e.

$$x^k \to x \ , \ y^k \to y \ , \ y^k \in A(x^k) \ \Rightarrow \ y \in A(x) \ .$$

O mapa é fechado em $C \subseteq X$ se for fechado em qualquer ponto de C. Note que se trocarmos na definição de fechamento a relação de pertinencia pela relação de igualdade, obtemos a definição de continuidade para mapas ponto-a-ponto. Assim, a propriedade de fechamento generaliza a propriedade de continuidade. Com efeito, uma função contínua é fechada, embora o inverso não seja necessariamente verdadeiro.

A idéia básica do teorema de convergência global de Zangwill é procurar alguma característica que "melhore" continuamente" a cada iteração do algoritmo. Esta característica é representada pelo conceito de função de descendência.

Seja A um algoritmo em X para resolver o problema P, e seja $S \subset X$ o conjunto de soluções de P. Uma fução $Z(x)$ é uma função de descendência para (X, A, S) se a composição de Z e A sempre decresce fora do conjunto solução, e não aumenta dentro dele, i.e.:
$x \notin S \wedge y \in A(x) \Rightarrow Z(y) < Z(x) \quad \text{e} \quad x \in S \wedge y \in A(x) \Rightarrow Z(y) \leq Z(x) \ .$

Em problemas de otimização, muitas vezes uma boa função de descendência é a própria função objetivo. Outras vezes, funções de descendência mais complexas tem de ser utilizadas, como por exemplo a soma da função objetivo com termos auxiliares, como penalidades para a violação de restrições, ou uma função decrescente de um contador de iterações sem melhora no objetivo para certos tipos de problemas degenerados.

Antes de enunciar o Teorema de Zangwill, recordemos dois conceitos de elementares de topologia de conjuntos: Um Ponto De Acumulação de uma seqüência é um ponto limite para uma se suas sub-seqüências. Um conjunto é Compacto sse qualquer seqüência (infinita) tem um ponto de acumulação dentro do conjunto. Em R^n, um conjunto é compacto sse é fechado e limitado.

Teorema de Convergência Global (Zangwill):

Seja Z uma função de descendência para o algoritmo A definido em X com conjunto de soluções S, e $x^0, x^1, x^2, \ldots$ uma seqüência gerada pelo algoritmo tal que:
A) O mapa A é fechado em qualquer ponto fora do conjunto solução,
B) Todos os pontos da seqüência permanecem dentro de um compacto, $C \subseteq X$, e
C) Z é contínua.
Então qualquer ponto de acumulação da seqüência estará no conjunto solução.

Pela compacidade de C, a seqüência gerada pelo algoritmo tem um ponto de acu-

mulação, $x \in C \subseteq X$, para uma subseqüência de acumulação, $x^{h(k)}$. Pela continuidade de Z em X, o limite do valor de Z na subseqüência de acumulação coincide com o valor de Z no ponto de acumulação, i.e., $Z(x^{h(k)}) \rightarrow Z(x)$. Mas a seqüência completa, $Z(x^k)$ é monotonicamente não crescente, de modo que se $h(k) \leq j \leq h(k+1)$ então $Z(x^{h(k)}) \geq Z(x^j) \geq Z(x^{h(k+1)})$, de modo que o valor de Z na seqüência completa também converge para o valor de Z no ponto de acumulação i.e., $Z(x^k) \rightarrow Z(x)$.

Consideremos agora, por absurdo, que x não seja uma solução, de modo que $Z(A(x)) < Z(x)$. Consideremos a subseqüência dos sucessores dos pontos na primeira subseqüência de acumulação, $x^{h(k)+1}$, seqüência esta que, por compacidade, tem tamém um ponto de acumulação, x'. Mas pelo resultado no parágrafo anterior, o valor da função de descendência em ambas as subseqüências converge para o valor limite da seqüência completa, i.e., $\lim Z(x^{h(k)+1}) = \lim Z(x^k) = \lim Z(x^{h(k)})$. Fica demonstrada assim a impossibilidade de que x não seja uma solução.

A formulação de muitos algoritmos é feita pela composição de vários passos. Assim, o mapa descrevendo o algoritmo completo é a composição de vários mapas, um para cada passo. Um exemplo típico é termos um passo correspondente a escolha de uma direção de busca, e o passo seguinte correspondente a uma busca linear. Os lemas apresentados a seguir são úteis na construção de mapas compostos.

Primeiro Lema de Composição:
Seja A de X em Y, e B de Y em Z, mapas ponto a conjunto, A fechado em $x \in X$, B fechado em $A(x)$. Se para qualquer seqüência x^k convergindo para x, $y^k \in A(x^k)$ tem um ponto de acumulação y, então o mapa composto $B \circ A$ é fechado em x.

Demonstração:
Como A é fechado em x, $y \in A(x)$. Como $y^h(k) \rightarrow y$ e B é fechado em y, uma seqüência $z^{h(k)} \in B(y^{h(k)})$ converge para um ponto $z \in B(y) \subseteq B(A(x))$.

Segundo Lema de Composição:
Seja A de X em Y, e B de Y em Z, mapas ponto a conjunto, A fechado em $x \in X$, B fechado em $A(x)$. Se Y for compacto, então o mapa composto $B \circ A$ é fechado em x.

Terceiro Lema de Composição:
Seja A um mapa ponto a ponto de X em Y, e B um mapa ponto a conjunto de Y em Z. Se A for contínuo em x, e B for fechado em $A(x)$. então o mapa composto $B \circ A$ é fechado em x.

Capítulo 3

Programação Quadrática

No capítulo anterior vimos um modelo relativamente simples de composição de carteiras que recaiu em um problema com função objetivo quadrática. Para esta categoria de problemas uma alternativa é, realizar algum tipo de linearização que nos permita empregar o algoritmo simplex. Todavia, é possível explorar a estrutura da função objetivo no caso em que as restrições do problema são lineares, conseguindo construir algoritmos para resolver o problema quadrático.

Estudaremos aqui este caso, assumindo que a função sendo minimizada é quadrática, isto é, da forma $x^{'}Qx + \eta p^{'}x$ e que as restrições são lineares. Para resolvê-lo iremos construir um algoritmo similar ao simplex, baseado nas condições de otimalidade de Lagrange.

3.1 Multiplicadores de Lagrange

Um problema de programação não linear tem a forma:

$$\min f(x),\ x \mid g(x) \leq 0 \wedge h(x) = 0\ ,\ f : \mathcal{R}^n \mapsto \mathcal{R}\ ,\ g : \mathcal{R}^n \mapsto \mathcal{R}^m\ ,\ h : \mathcal{R}^n \mapsto \mathcal{R}^k\ .$$

Podemos imaginar f como um *potencial*, ou como a "altura" de uma superfície. *Equipotenciais*, ou *curvas de nível*, são curvas nas quais $f(x)$ mantém um valor constante. O gradiente,

$$\nabla f \equiv \partial f/\partial x = \left[\ \partial f/\partial x_1,\ \ \partial f/\partial x_2,\ \ \ldots\ \ \partial f/\partial x_n\ \right]$$

nos dá a direção de "maior inclinação" da superfície $f(x)$. Temos também que o gradiente de f num ponto x, é ortogonal (ou perpendicular) à curva de nível de f que passa por x.

Daremos a seguir uma explicação intuitiva para uma condição necessária de otimalidade. Podemos imaginar uma partícula puxada "para baixo" por uma "força" $-\nabla f(x)$. Um ponto de mínimo será portanto um *ponto de equilíbrio* para a partícula, i.e. um ponto

onde a força que puxa a partícula para baixo se anula, ou então é equilibrada por "forças de reação" exercidas pelas restrições. A força de reação de uma restrição de desigualdade, $g_i(x) \leq 0$, deve ser:

a) perpendicular à curva de nível desta restrição (pois a partícula pode mover-se livremente ao longo da curva de nível),

b) uma força de reação "para dentro" da região viável (i.e. impedindo que a partícula saia para fora da região viável).

c) Ademais esta restrição de desigualdade só pode exercer uma força de reação quando estiver ativa, algo como: "a partícula não pode apoiar-se numa parede (restrição) em que não esteja encostada".

Uma restrição de igualdade, $h_i(x) = 0$, pode ser vista como um par de restrições de desigualdade, $h_i(x) \leq 0$ e $h_i(x) \geq 0$; Ademais, dentro da região viável, este par de restrições será sempre ativo.

A nossa discussão intuitiva pode ser resumida analiticamente nas *Condições de Lagrange*: Se $\hat{x} \in \mathcal{R}^n$ é um ponto ótimo então:

$$\exists u \in \mathcal{R}^m \ , \ v \in \mathcal{R}^k \ \mid u\nabla g + v\nabla h - \nabla f = 0 \ , \ \text{onde} \ \ u \leq 0 \wedge ug = 0$$

A condição $u \leq 0$ implica que a reação das restrições de desigualdade aponta para dentro da região viável, enquanto a *condição de complementaridade*, $ug = 0$, implica que apenas restrições ativas podem exercer forças de reação. Os vetores u e v são conhecidos como *multiplicadores de Lagrange*.

3.2 Condição de Otimalidade

O Problema de Programação Quadrática (QP) é:

$$minf(x) \equiv (1/2)x'Qx - \eta p'x \mid x \geq 0 \wedge Te * x = te \wedge Tl * x \leq tl$$

onde as dimensões da matrizes são: $Te \ me \times n$, $me < n$, $Tl \ ml \times n$, $ml < n$, $Ml = 1, 2, \ldots ml$, $Me = 1, 2, \ldots me$, $N = 1, 2, \ldots n$. e, por hipótese, a forma quadrática é simétrica e positiva definida, isto é $Q = Q'$, $Q > 0$.

No QP, o gradiente da função objetivo é:

$$\nabla f = x'Q - \eta p' \ ,$$

e os gradientes das restrições são:

$$g_i(x) = T_i x \leq t_i \Rightarrow \nabla g_i = T_i \ ,$$

gerando as condições de equilíbrio de Lagrange, ou condições de otimalidade:

$$x \in R^n_+, s \in R^n_+, l \in R^{ml}_+, e \in \mathcal{R}^{me}, \ \mid \ -(x'Q - \eta p') + s' - l'Tl + e'Te = 0$$

$$\wedge \forall i \in N \ , \ x_i s_i = 0 \ \wedge \ \forall k \in Ml \ , \ (Tl * x - tl)_k l_k = 0$$

ou

$$x \in R^n_+, s \in R^n_+, l \in R^{ml}_+, e \in \mathcal{R}^{me}, y \in R^{ml}_+ \ \mid \ Qx - s' + Tl' * l + Te'e = \eta p$$

$$\wedge \forall i \in N \ , \ x_i s_i = 0 \ \wedge \ \forall k \in Ml \ , \ yl_k l_k = 0 \text{ onde } yl = (tl - Tl * x)$$

As condições de complementaridade (CC), $x's = 0$ e $yl'l = 0$, indicam que só restrições justas podem equilibrar componentes negativas do gradiente da função objetivo. Com a mudança de variáveis $e = ep - em$, $ep, \ em \geq 0$, o ponto ótimo dado pelas "Equações de Viabilidade e Otimalidade" ou EVO:

$$\begin{bmatrix} x \\ l \\ ep \\ en \\ s \\ yl \end{bmatrix} \geq 0 \mid \begin{bmatrix} Tl & 0 & 0 & 0 & 0 & I \\ Te & 0 & 0 & 0 & 0 & 0 \\ Q & Tl' & Te' & -Te' & -I & 0 \end{bmatrix} \begin{bmatrix} x \\ l \\ ep \\ en \\ s \\ yl \end{bmatrix} = \begin{bmatrix} tl \\ te \\ \eta p \end{bmatrix}$$

$$x's = 0, \ yl'l = 0.$$

3.3 Complementaridade Linear

Em EVO as CC implicam que, na solução, são nulos: pelo menos, n elementos dentre x e s; e pelo menos ml elementos dentre yl e l. ep e en são, respectivamente, a parte positiva e negativa do vetor irrestrito e, de modo que está implícita na formulação do problema que, na solução, são nulos pelo menos me elementos dentre ep e en. Assim, na solução, não há mais que $ml + me + n$ variáveis não nulas, que podem ser escritas como uma uma solução básica do sistema EVO. Isto sugere usarmos o algoritmo Simplex para resolver o QP. Assumiremos agora, por mera conveniência para as aplicações que se seguem, que $tl \geq 0$ (alterar o material que se segue de modo a prescindir desta hipótese é um exercício trivial). Formularemos EVO como um PPL na forma padrão, mais as CC.

Este é o problema de complementaridade linear, PCL:

$$\min \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} x \\ l \\ ep \\ en \\ s \\ yl \\ ye \\ yq \end{bmatrix} \quad \begin{bmatrix} x \\ l \\ ep \\ en \\ s \\ yl \\ ye \\ yq \end{bmatrix} \geq 0 \mid$$

$$\begin{bmatrix} Tl & 0 & 0 & 0 & 0 & I & 0 & 0 \\ Te & 0 & 0 & 0 & 0 & 0 & De & 0 \\ Q & Tl' & Te' & -Te' & -I & 0 & 0 & Dq \end{bmatrix} \begin{bmatrix} x \\ l \\ ep \\ en \\ s \\ yl \\ ye \\ yq \end{bmatrix} = \begin{bmatrix} tl \\ te \\ \eta p \end{bmatrix}$$

onde temos as CC, os novos blocos, e o vértice inicial, dados por, respectivamente

$$x's = 0,\ yl'l = 0, \quad Dq = diag(sign(\eta p)),\ De = diag(sign(te)), \quad tl \geq 0.$$

$$\begin{bmatrix} x \\ l \\ ep \\ en \\ s \\ yl \\ ye \\ yq \end{bmatrix}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ tl \\ \mid te \mid \\ \mid p \mid \end{bmatrix}.$$

Na solução inicial as CC são satisfeitas, pois $x's = 0'0 = 0$, e $yl'l = tl'0 = 0$. Para assegurar que as CC continuem sendo satisfeitas, usaremos uma regra adicional no simplex: Não permitiremos que

- x_i ou se torne básica se s_i for uma variável básica, e vice-versa,
- yl_i ou se torne básica se l_i for uma variável básica, e vice-versa.

Regras proibitivas como esta são denominadas regras 'Tabu'. Precisamos agora demonstrar que, mesmo com esta regra tabu, o algoritmo termina num vértice ótimo do PCL de valor 0, i.e., com $ye = 0$, $yq = 0$ e x a solução ótima do QP.

Assumiremos que $Tex = te$. Para tanto minimizaremos primeiro a soma das variáveis artificiais (ye) correspondentes a estas restrições, i.e. levaremos as variáveis artificiais ye para fora da base. Em seguida, mantendo ye fora da base, tentaremos levar as variáveis artificiais yq para fora da base.

Consideremos a possibilidade do algoritmo terminar num vértice de valor positivo, sem nenhuma variável residual de custo reduzido negativo cuja variável complementar já não seja básica. Derivaremos desta possibilidade uma contradição, provando assim a corretude do algoritmo PCL para resolução do QP. Particionemos cada um dos vetores x, s, yl e l em três blocos, de modo que as variáveis:

- em x_1 sejam básicas, e portanto suas complementares em s_1 sejam residuais,
- em s_2 sejam básicas, e portanto suas complementares em x_2 sejam residuais,
- em yl_1 sejam básicas, e portanto suas complementares em l_1 sejam residuais,
- em l_2 sejam básicas, e portanto suas complementares em yl_2 sejam residuais.

Esta solução é a solução ótima do seguinte PPL (as variáveis omitidas são as proibidas de entrar na base pela regra de complementariedade), PAUX:

$$\min \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_3 \\ l_2 \\ l_3 \\ e \\ s_2 \\ s_3 \\ yl_1 \\ yl_3 \\ yq \end{bmatrix}, \quad \begin{array}{lcl} x_1 & \geq & 0 \\ x_3 & \geq & 0 \\ l_2 & \geq & 0 \\ l_3 & \geq & 0 \\ e & * & \\ u_2 & \geq & 0 \\ u_3 & \geq & 0 \\ yl_1 & \geq & 0 \\ yl_3 & \geq & 0 \\ yq & \geq & 0 \end{array} \quad |$$

$$\begin{bmatrix} Tl_1^1 & Tl_1^3 & 0 & 0 & 0 & 0 & 0 & I & 0 & 0 \\ Tl_2^1 & Tl_2^3 & 0 & 0 & 0 & 0 & 0 & 0 & I & 0 \\ Te^1 & Te^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Q_1^1 & Q_1^3 & Tl_2^{1'} & Tl_3^{1'} & Te^{1'} & 0 & 0 & 0 & 0 & Dq_1 \\ Q_2^1 & Q_2^3 & Tl_2^{2'} & Tl_3^{2'} & Te^{2'} & -I & 0 & 0 & 0 & Dq_2 \\ Q_3^1 & Q_3^3 & Tl_2^{3'} & Tl_3^{3'} & Te^{3'} & 0 & -I & 0 & 0 & Dq_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_3 \\ l_2 \\ l_3 \\ e \\ s_2 \\ s_3 \\ yl_1 \\ yl_3 \\ yq \end{bmatrix} = \begin{bmatrix} tl_1 \\ tl_2 \\ te \\ \eta p_1 \\ \eta p_2 \\ \eta p_3 \end{bmatrix}$$

e consideremos também seu dual, DAUX:

$$\max \left[\begin{array}{cccccc} tl_1' & tl_2' & te' & \eta p_1{}' & \eta p_2{}' & \eta p_3{}' \end{array} \right] \left[\begin{array}{c} ul_1 \\ ul_2 \\ ue \\ uq_1 \\ uq_2 \\ uq_3 \end{array} \right]$$

$$\left[\begin{array}{cccccc} Tl_1^{1'} & Tl_2^{1'} & Te^{1'} & Q_1^{1'} & Q_2^{1'} & Q_3^{1'} \\ Tl_1^{3'} & Tl_2^{3'} & Te^{3'} & Q_1^{3'} & Q_2^{3'} & Q_3^{3'} \\ 0 & 0 & 0 & Tl_2^1 & Tl_2^2 & Tl_2^3 \\ 0 & 0 & 0 & Tl_3^1 & Tl_3^2 & Tl_3^3 \\ 0 & 0 & 0 & Te^1 & Te^2 & Te^3 \\ 0 & 0 & 0 & 0 & -I & 0 \\ 0 & 0 & 0 & 0 & 0 & -I \\ I & 0 & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & Dq_1' & Dq_2' & Dq_3' \end{array} \right] \left[\begin{array}{c} ul_1 \\ ul_2 \\ ue \\ uq_1 \\ uq_2 \\ uq_3 \end{array} \right] \begin{array}{ccll} \geq & 0 & 1\text{a, justa, } x_1 > 0 \\ \geq & 0 & 2\text{a, folgada,} \\ \geq & 0 & 3\text{a, justa, } l_2 > 0 \\ \geq & 0 & 4\text{a, folgada,} \\ = & 0 & 5\text{a, } e \text{ irrestrito} \\ \geq & 0 & 6\text{a, justa, } s_2 > 0 \\ \geq & 0 & 7\text{a, folgada,} \\ \geq & 0 & 8\text{a, justa, } yl_1 > 0 \\ \geq & 0 & 9\text{a, folgada,} \\ \geq & 1 & 10\text{a} \end{array}$$

No dual apresentamos a solução ótima, particionada conforme o primal. Nas restrições do DAUX temos igualdade na 5a linha devido a termos e sem restrição de sinal, e igualdades nas 1a, 3a, 6a, e 8a linhas pelo teorema de folgas complementares, pois as variáveis respectivamente indicadas são básicas.

Da 6a linha blocada das restrições do DAUX vemos que $uq_2 = 0$, e da 8a linha vemos que $ul_1 = 0$. Podemos pois escrever a equação (inequação justa) da 1a linha descartando os termos em ul_1 e uq_2. Multiplicando esta equação por uq_1' temos

$$uq_1' \left[\begin{array}{cccc} Tl_2^{1'} & Te^{1'} & Q_1^{1'} & Q_3^{1'} \end{array} \right] \left[\begin{array}{c} ul_2 \\ u_e \\ uq_1 \\ uq_3 \end{array} \right] = uq_1' \mathbf{0} = 0$$

analogamente, multiplicando a inequação da 2a linha por uq_3', e notando que pela 7a linha $uq_3' \leq 0$, temos

$$uq_3' \left[\begin{array}{cccc} Tl_2^{3'} & Te^{3'} & Q_1^{3'} & Q_3^{3'} \end{array} \right] \left[\begin{array}{c} ul_2 \\ u_e \\ uq_1 \\ uq_3' \end{array} \right] \leq 0$$

Somando e transpondo as duas últimas desigualdades temos

$$\begin{aligned}
&\begin{bmatrix} ul_2' & u_e' & uq_1' & uq_3' \end{bmatrix} \begin{bmatrix} Tl_2^1 \\ Te^1 \\ Q_1^1 \\ Q_3^1 \end{bmatrix} uq_1 \\
+ &\begin{bmatrix} ul_2' & u_e' & uq_1' & uq_3' \end{bmatrix} \begin{bmatrix} Tl_2^3 \\ Te^3 \\ Q_1^3 \\ Q_3^3 \end{bmatrix} uq_3 \quad \leq 0
\end{aligned}$$

ou

$$\begin{aligned}
&ul_2' \begin{bmatrix} Tl_2^1 & Tl_2^3 \end{bmatrix} \begin{bmatrix} uq_1 \\ uq_3 \end{bmatrix} \\
+ &\, ue' \begin{bmatrix} Te^1 & Te^3 \end{bmatrix} \begin{bmatrix} uq_1 \\ uq_3 \end{bmatrix} \\
+ &\begin{bmatrix} uq_1' & uq_3' \end{bmatrix} \begin{bmatrix} Q_1^1 & Q_1^3 \\ Q_3^1 & Q_3^3 \end{bmatrix} \begin{bmatrix} uq_1 \\ uq_3 \end{bmatrix} \quad \leq 0
\end{aligned}$$

Notemos agora que os dois primeiros termos desta última inequação são não negativos:

- Pela 9a linha das restrições do DAUX vemos que $ul_2 \geq 0$, e pela 6a linha que $uq_2 = 0$, donde, usando a 3a linha, concluimos que o primeiro termo é não negativo.

- Pela 5a linha o segundo termo é nulo.

Obtivemos assim uma contradição com a positividade de Q.

3.4 Problema Quadrático Paramétrico

Consideremos agora o Problema Quadrático Paramétrico (QPP) que consiste de encontrar todas as soluções ótimas do QP em função do parâmetro η: estas são as "soluções eficientes" do QPP. Da mesma forma que reduzimos o problema de programação quadrática (QP) a um problema de programação linear (PPL), o problema de complementariedade linear (PCL), reduziremos o problema quadrático paramétrico (QPP) a um problema linear paramétrico (PLP).

Consideremos no QP, uma solução básica das EVO,

$$\begin{bmatrix} x \\ l \\ ep \\ en \\ s \\ yl \end{bmatrix} \geq 0 \mid \begin{bmatrix} Tl & 0 & 0 & 0 & 0 & I \\ Te & 0 & 0 & 0 & 0 & 0 \\ Q & Tl' & Te' & -Te' & -I & 0 \end{bmatrix} \begin{bmatrix} x \\ l \\ ep \\ en \\ s \\ yl \end{bmatrix} = \begin{bmatrix} tl \\ te \\ \eta p \end{bmatrix}$$

$$x's = 0 \ , \ yl'l = 0.$$

após determinar um primeiro parâmetro crítico, podemos reescrever o problema paramétrico introduzindo o termo parametrizado do vetor do lado direito como a última coluna da matriz de coeficientes de um novo PPL:

$$\min \pm\eta$$

$$\begin{bmatrix} x \\ l \\ ep \\ en \\ s \\ yl \\ \eta \end{bmatrix} \geq 0 \mid \begin{bmatrix} Tl & 0 & 0 & 0 & 0 & I & 0 \\ Te & 0 & 0 & 0 & 0 & 0 & 0 \\ Q & Tl' & Te' & -Te' & -I & 0 & -p \end{bmatrix} \begin{bmatrix} x \\ l \\ ep \\ en \\ s \\ yl \\ \eta \end{bmatrix} = \begin{bmatrix} tl \\ te \\ 0 \end{bmatrix}$$

$$x's = 0 \ , \ yl'l = 0$$

Como no PLP, $x(\eta)$, a curva das *soluções eficientes* do QPP é linear entre um par de parâmetros críticos consecutivos. Vemos pois que o valor ótimo da solução do problema quadrático com o correspondente parâmetro $\eta = (1-\lambda)\eta_k + \lambda\eta_{k+1}$, $vopt(\eta)$, é uma função quadrática em η:

$$\begin{aligned} vopt(\eta) \ &= \ (1-\lambda)p'x(\eta_k) + \lambda p'x(\eta_{k+1}) \\ &\quad +(1-\lambda)^2 x(\eta_k)'Qx(\eta_k) + \lambda^2 x(\eta_{k+1})'Qx(\eta_{k+1}) \\ &\quad +2\lambda(1-\lambda)x(\eta_k)'Qx(\eta_{k+1}) \ . \end{aligned}$$

Portanto o gráfico de da parte linear versus a parte quadrática de $vopt(\eta)$, para $-\infty < \eta < \infty$, será uma curva contínua composta de arcos de quadráticos.

3.5 Implementação Computacional

A implementação dos algoritmos para QP e QPP deve contemplar vários aspectos computacionais, entre eles:

a. ser numericamente estável,

b. preservar a estrutura blocada nas sucessivas bases do simplex,

c. manter a esparsidade das sub-matrizes das bases,

d. utilizar estruturas de dados eficientes para matrizes esparsas,

e. controlar dinamicamente os erros com operações de ponto flutuante.

Estes aspectos, comuns a algoritmos de otimização envolvendo álgebra linear computacional, fogem ao escopo introdutório deste trabalho, sendo tratados em algumas das referências bibliográficas citadas.

É importante ressaltar que, mesmo usando as mais avançadas técnicas de estabilização, os algoritmos dependem fortemente de termos:

1. A matriz de covariância positiva definida.

2. As restrições linearmente independentes.

Ao tratarmos um problema "malposto", i.e. um problema próximo de violar qualquer das duas condições acima, o algoritmo torna-se instável, produzindo resultados incorretos. Para tentar sanar estes problemas procura-se tomar alguns cuidados no momento de modelar o problema. No capítulo 3 apresentaremos algumas sugestões nesta direção para o problema de seleção de carteiras.

Capítulo 4

Modelo de Markowitz

4.1 Análise de Média e Variância

O modelo de Markowitz é um modelo para formação de portfólios, que visa maximizar a utilidade de um investidor, que deve escolher um conjunto de ativos para compor uma carteira, obedecendo a restrições de disponibilidade de recursos ou outra natureza. Consideram-se n ativos, com taxas de retorno $r_1, \ldots, r_n$, sendo que um portfólio é especificado pela quantidade investida em cada ativo, $x_1, \ldots, x_n$. As taxas de retorno são consideradas variáveis aleatórias. Os dados do modelo (que supõem-se conhecidos) são o vetor das taxas de retorno esperadas, $E(r)$, e a matriz de covariância das taxas de retorno, $Cov(r)$. É um pressuposto do modelo termos a matriz de covariância positiva definida.

O modelo de Markowitz usa uma função utilidade do tipo média-e-variância vide apêndice B), e define como portfólio ótimo um portfólio x que maximiza a função utilidade estando o mesmo sujeito a restrições lineares de igualdade, desigualdade e sinal:

$$\max_x \; U(x) = \eta x' E(r) - x' Cov(r) x \;, \quad x \geq 0 \;\mid\; Te * x = te \;\wedge\; Tl * x \leq tl$$

O parâmetro $1/\eta$ é a aversão ao risco do investidor.

Nos capítulos anteriores, estudamos algoritmos para resolver este problema de otimização: dado valor particular de η este era o QP - Problema de Programação Quadrática. Encontrar a curva das soluções ótimas em função do parâmetro η, $x(\eta)$, (as soluções eficientes) era o QPP - Problema Quadrático Paramétrico.

Para não sobrecarregar a notação, ao falarmos dos portfólios x e y, denotaremos $x'E(r)$ ou $Cov(x'r, y'r)$ por, respectivamente, $e(x)$ ou e_x, e $\sigma(x, y)$ ou $\sigma_{x,y}$. No caso de uma família de portfólios parametrizada por um escalar, $x(\lambda),\ \lambda \in \mathcal{R}$ denotamos $x'(\lambda)E(r)$ por $e(\lambda)$ ou e_λ, e $Cov(x(\alpha)'r, x(\beta)'r)$ por $\sigma(\alpha, \beta)$ ou $\sigma_{\alpha,\beta}$.

4.2 Distribuição das Taxas de Retorno

Dados, em dois instantes de tempo 0 e t, o valor ou preço de um ativo, v_0 e v_t, existem várias definições de "taxa de retorno" do ativo neste período. As mais comuns são:

- A *taxa de retorno simples*, $rs \mid v_t = (1 + rs)v_0$ donde $rs = (v_t/v_0) - 1$
- A *taxa de retorno t vezes composta*, $rc \mid v_t = (1 + rc)^t v_0$ donde $rc = (v_t/v_0)^{(1/t)} - 1$
- A *taxa de retorno continuamente composta*, $v_t = v_0 \exp(r * t)$, isto é, dada por $r = (1/t)\ln(v_t/v_0) = (1/t)(\ln(v_t) - \ln(v_0))$ A taxa de retorno continuamente composta pode ser derivada como o limite de Nepper, i.e. o limite da taxa de retorno t vezes composta quando $t \to \infty$, ou através da equação diferencial $r \mid \frac{\partial v(t))}{\partial t} = rv(t)$, com condição de contorno $v(0) = v_0$ e $v(t) = v_t$.

Para manter a coerência de algumas interpretações, e para facilitar o uso de métodos estatísticos de estimação, gostaríamos que a distribuição da taxa de retorno utilizada tivesse algumas propriedades: minimamente gostaríamos de ter uma distribuição simétrica, idealmente, queremos uma distribuição Normal. Na maioria das situações, a taxa de retorno continuamente composta é a que melhor se adapta a estes quesitos.

4.3 Fronteira Eficiente

Consideremos a curva das soluções eficientes do QPP, $x(\eta), 0 \leq \eta \leq \infty$. O gráfico do valor da parte linear versus a parte quadrática do valor das soluções eficiente,

$$e(\eta) \equiv E(r)'x(\eta) \;\; \text{contra} \;\; v(\eta) \equiv x'(\eta)Cov(r)x(\eta) \;,$$

é denominado *fronteira eficiente*. Podemos interpretar o parâmetro η como uma medida de quanto maximizar o retorno esperado, $e(\eta)$, é preferível a minimizar o risco do investimento, $x'Sx$. No extremo $\eta = 0$ queremos apenas minimizar o risco; no extremo oposto, $\eta = \infty$, queremos apenas maximizar o retorno esperado.

Provemos que a fronteira eficiente é uma curva côncava. Usaremos para tanto o fato (demonstrado no capítulo 2) de que a solução eficiente do QPP entre dois parâmetros críticos, digamos η_0 e η_1, $\eta_1 > \eta_0$, é a correspondente combinação convexa das soluções eficientes nestes dois parâmetros críticos, i.e., escrevendo um parâmetro intermediário como

$$\eta(\lambda) = (1 - \lambda)\eta_0 + \lambda\eta_1$$

temos a solução eficiente:

$$x(\eta(\lambda)) = (1 - \lambda)x(\eta_0) + \lambda x(\eta_1)$$

de modo que

$$e(x(\eta(\lambda))) = (1-\lambda)E(r)'x(\eta_0) + \lambda E(r)'x(\eta_1)$$

ou

$$e(\eta(\lambda)) = (1-\lambda)e(\eta_0) + \lambda e(\eta_1)$$

Numa notação mais compacta,

$$e_\lambda = (1-\lambda)e_0 + \lambda e_1$$

Analogamente,

$$\begin{aligned} v(x(\eta(\lambda))) = \ & (1-\lambda)^2 Cov(r'x(\eta_0), r'x(\eta_0)) \\ & +2\lambda(1-\lambda)Cov(r'x(\eta_0), r'x(\eta_1)) + \lambda^2 Cov(r'x(\eta_1), r'x(\eta_1)) \end{aligned}$$

ou

$$v(\eta(\lambda)) = (1-\lambda)^2\sigma(\eta_0,\eta_0) + 2\lambda(1-\lambda)\sigma(\eta_0,\eta_1) + \lambda^2\sigma(\eta_1,\eta_1)$$

ou ainda

$$v_\lambda = (1-\lambda)^2\sigma_{0,0} + 2\lambda(1-\lambda)\sigma_{0,1} + \lambda^2\sigma_{1,1}$$

portanto

$$\frac{\partial e_\lambda}{\partial \lambda} = e_1 - e_0$$

$$\frac{\partial v_\lambda}{\partial \lambda} = -2(1-\lambda)\sigma_{0,0} + 2(1-2\lambda)\sigma_{0,1} + 2\lambda\sigma_{1,1}$$

$$\frac{\partial^2 v_\lambda}{\partial \lambda^2} = 2(\ \sigma_{0,0} - 2\sigma_{0,1} + \sigma_{1,1}\ )$$

Como assumimos que a matriz de covariância é positiva definida, não podemos ter $\rho_{0,1} = 1$. Assim, sabemos que (vide apêndice) a última expressão é positiva, e portanto, entre dois pontos críticos, a fronteira eficiente é côncava (note que definimos a fronteira eficiente como o gráfico $e(\eta) \times v(\eta)$, e não $v(\eta) \times e(\eta)$ que seria uma curva convexa).

Nos pontos críticos a derivada $\partial\sigma/\partial e$ é descontínua. Para provar que a fronteira eficiente, considerada integralmente, é côncava, basta provar que, num ponto crítico, esta derivada não pode aumentar. Consideremos a possibilidade de que tal ocorresse: Neste caso teríamos uma corda entre portfólios x e y, situados numa vizinhança suficientemente próxima do ponto crítico, cuja utilidade domina a fronteira eficiente. Mas como a correlação entre estes dois portfólios, $\rho_{x,y} < 1$ esta corda é por sua vez dominada pela utilidade das combinações convexas destes portfólios, $(1-\lambda)x + \lambda y$; contradizendo a hipótese de uma fronteira como a considerada ser realmente eficiente.

No capítulo precedente discutimos brevemente a questão de problemas mal postos, nos quais a matriz de covariância podia deixar de ser positiva definida ou as restrições poderiam ser LD. Nos problemas de composição de carteiras as seeguintes sugestões são úteis para evitar problemas mal postos:

As seguintes sugestões são úteis para evitar problemas malpostos:

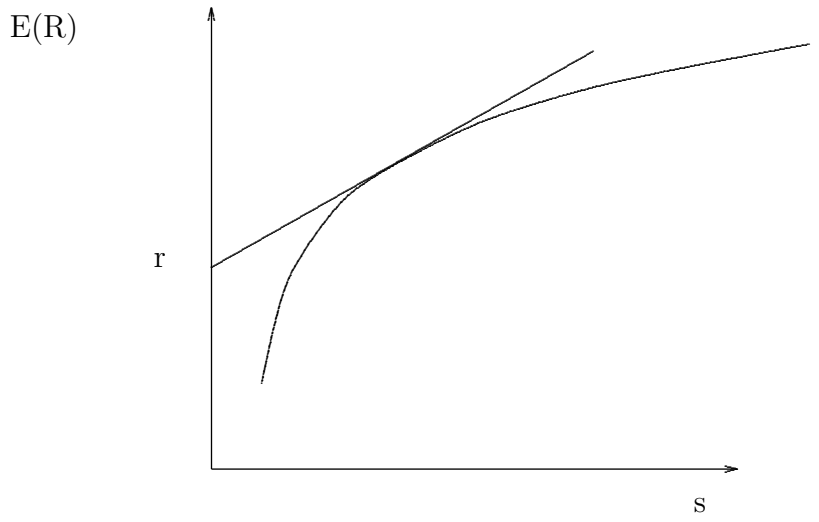


Figura 4.1: O gráfico risco X retorno

1. Adição gradual de ativos ao portfólio:

   (a) Ao invés de considerar de partida todos os ativos disponíveis no mercado, comece com um subgrupo básico de tamanho mais modesto, como por exemplo os ativos já presentes no portfólio, ou os mais líquidos no mercado.

   (b) A seguir acrescente novos subgrupos ao modelo, agrupando sempre ativos semelhantes, i.e. altamente correlacionados. Critérios úteis de semelhança incluem o país sede da empresa, seu setor de atividade, etc.

   (c) Mantenha na modelagem apenas os ativos em cada subgrupo adicional que efetivamente entram na composição de carteiras eficientes.

2. Adição gradual de restrições:

   (a) Ao invés de considerar de partida todas as restrições desejadas, comece com um subgrupo de tamanho mais modesto, como as restrições de igualdade. A seguir acrescente, dentre as condições de desigualdade desejadas, apenas as que se mostrarem violadas.

   (b) Tome cuidado para não usar um restrição de igualdade quando o que se deseja no modelo é apenas uma restrição de desigualdade.

   (c) Tome muito cuidado para não escrever restrições redundantes. Os exemplos abaixo mostram alguns exemplos de restrições redundantes:

      i.
      $$x_1 + x_2 \leq 1 \ , \ 0 \leq x_1 \leq 0.5 \ , \ 0 \leq x_2 \leq 0.5 \ .$$

      ii.
      $$x_1 + x_2 = 1 \ , \ 0 \leq x_1 \leq 0.5 \ , \ 0 \leq x_2 \leq 0.5 \ .$$

4.4 Modelos de Tobin e Brennan

O modelo de Tobin considera, além dos ativos de risco, um ativo sem risco de taxa de retorno e_0, com desvio padrão $\sigma_0 = 0$. Consideremos uma combinação convexa entre o ativo sem risco e um portfólio eficiente, $x(\eta)$, o portfólio

$$x(\lambda) = (1 - \lambda)x_0 + \lambda x(\eta)$$

A covariância de uma variável aleatória com uma constante é sempre zero, portanto o retorno esperado e o desvio padrão deste portfólio são dados por:

$$e(\lambda) = (1 - \lambda)e_0 + \lambda e(\eta)$$

$$\sigma(\lambda) = \lambda\sigma(\eta)$$

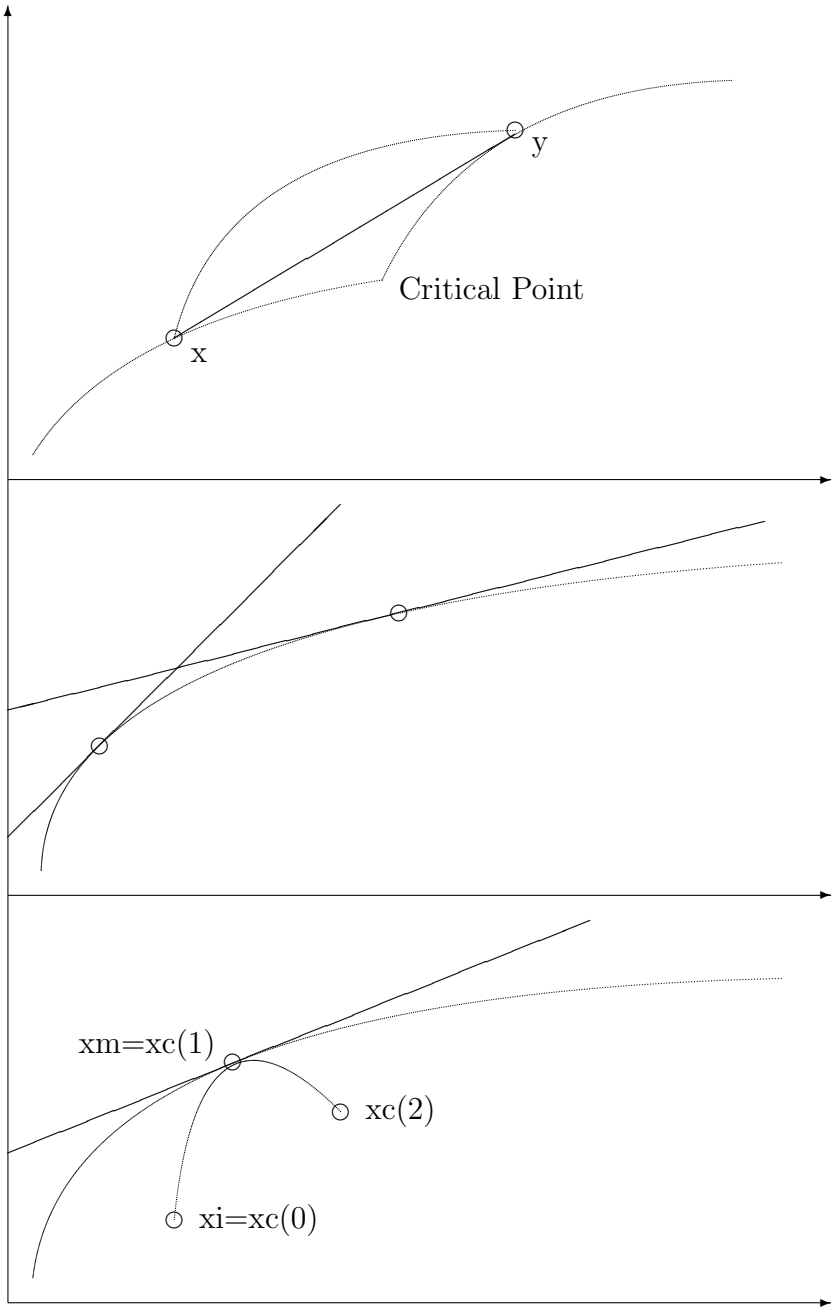
Estes portfólios estão representados, pela semi-reta secante à fronteira eficiente. Vemos todavia que a utilidade de qualquer ponto na reta secante é dominada pela utilidade de um ponto na semi-reta tangente a fronteira eficiente (em virtude da concavidade da fronteira eficiente, esta semi-reta é única).

Parâmetros $\lambda > 1$ podem ser interpretados como empréstimo (possuir uma quantidade negativa) do ativo sem risco aplicados ao portfólio tangente, $x(\eta)$. O parâmetro correspondente ao ponto de tangência, $\hat{\eta}$, é a raiz de

$$h(\eta) = \frac{\partial e(\eta)}{\partial \eta} \; / \; \frac{\partial \sigma(\eta)}{\partial \eta} \quad - \; (e(\eta) - e_0) \; / \; \sigma(\eta)$$

Se o ponto de tangência não é crítico, $h(\eta) = 0$ traduz a igualdade entre as inclinações (derivadas) da fronteira eficiente e da reta secante. A função $h(\eta)$ é monótona, mas descontínua nos pontos críticos. Se $h(\eta)$ muda de sinal num ponto de descontinuidade, então o parâmetro de tangência coincide com um parâmetro crítico.

O modelo de Brennan assume dois ativos sem risco, a interpretação deste modelo é que geralmente pagamos por um empréstimo uma taxa maior do que a que recebemos por um depósito no ativo sem risco. Nesta situação temos dois pontos de tangência, e uma fronteira eficiente que está na primeira semi-reta para $0 \leq \lambda < 1$, coincide com a fronteira eficiente do modelo de Markowitz para $\lambda = 1$, e esta sobre a segunda semi-reta para $\lambda > 1$.



4.5 Modelos de Índices

O modelo de Cohen e Pogue considera *taxas de crescimento setoriais*, c_k, para os setores da economia $k = 1, \ldots K$. A taxa de retorno de cada ativo, r_i $i = 1 \ldots N$ é a soma de uma *taxa própria*, a_i, mais as taxas de crescimento setoriais ponderadas por *fatores de sensibilidade*, $B_{i,k}$. Neste tipo de modelo temos sempre que $k << n$. Ademais, o modelo supõe que as taxas próprias são variáveis aleatórias independentes entre si, e também

independentes das taxas setoriais:

$$r_i = a_i + \sum_k B_{i,k} c_k \ , \quad Cov(a_i, a_j) = 0, \ Cov(a_i, c_k) = 0$$

Usando as leis de transformação, podemos calcular a esperança e covariância do vetor dos retornos:

$$E(r) = E(a) + B\,E(c) \ , \quad Cov(r) = diag([Var(a_1) \ldots Var(a_n)]) \ + B\,Cov(c)\,B'$$

Para usar este modelo temos que estimar $E(a_i)$, $Var(a_i)$, $B_{i,k}$, $E(c)$ e $Cov(c)$, i.e. da ordem de $2n + k * n + k^2$, ao invés de n^2 parâmetros. Para fazer estas estimativas são usadas uma série de técnicas que integram estatística e otimização. A partir de séries históricas de valores de r_i e $c_{i,k}$, costumamos estimar o vetor a e a matriz B, por:

1. Modelos Lineares, como por exemplo:
   (a) Análise de regressão,
   (b) Filtro de Kalman; ou
2. Modelos não Lineares, como por exemplo:
   (a) Regressão não Linear,
   (b) Redes Neurais,
   (c) Filtros Adaptativos.

A análise e previsão de índices é feita utilizando todos os instrumentos listados acima, e ainda técnicas econométricas como Análise de Clusters, Análise de Box-Jenkins, Análise Espectral, Análise de Insumo-Produto, etc.

Modelos Diagonais

Um modelo de índices está em forma *diagonal*, se a matriz de covariância dos índices é uma matriz diagonal,

$$Cov(c) = \text{diag}([\sigma_1^2, \ldots \sigma_n^2]) = \begin{bmatrix} \sigma_{1,1} & & 0 \\ & \ddots & \\ 0 & & \sigma_{k,k} \end{bmatrix}$$

Modelos diagonais permitem uma interpretação mais simples, pois $i \neq j \Rightarrow Cov(c_i, c_j) = 0$. Na verdade, qualquer modelo de índices pode ser posto na forma diagonal, como veremos a seguir.

É um teorema básico de Análise Numérica que, dada uma matriz simétrica e positiva definida, S, podemos computar sua *fatoração de Cholesky*, i.e. podemos encontrar uma matriz triangular inferior, L, tal que $S = LL'$.

Usaremos agora a fatoração de Cholesky para transformar um modelo de índices geral, cuja matriz de covariância dos índices não é diagonal, num modelo diagonal equivalente. Consideremos o novo vetor de índices

$$d = L^{-1}c \ , \text{ onde } \ Cov(c) = S = LL' \ .$$

A matriz de covariância dos novos índices é diagonal, pois

$$Cov(d) = L^{-1}Cov(c)L^{-t} = L^{-1}(LL')L^{-t} = (L^{-1}L)(L'L^{-t}) = I$$

O modelo originalmente escrito em função dos índices c, pode ser facilmente reescrito em função dos novos índices, d:

$$r = a + Bc \Rightarrow r = a + BLL^{-1}c = a + \tilde{B}d$$

4.6 Modelos de Equilíbrio

Apresentaremos agora alguns modelos de equilíbrio. Estes modelos tem uma natureza essencialmente diferente dos modelos que estudamos até agora. Os modelos que vimos até agora são modelos de decisão: eles supõem que um dado investidor tem certas preferências, e que este investidor foi a campo e "mediu" algumas características dos ativos no mercado. A partir daí os modelos de decisão fornecem algoritmos que encontram as decisões "ótimas" (ou racionais) para este investidor.

Modelos de equilíbrio assumem que todos os investidores no mercado se comportam racionalmente usando um mesmo modelo de decisão. Geralmente estes modelos também assumem que todos eles estão de *acordo* quanto às entradas do modelo (suas medidas e previsões sobre as características relevantes dos ativos no mercado são idênticas). A partir destas hipóteses estes modelos calculam propriedades de *pontos de equilíbrio* (ou pontos estacionários) do mercado.

Modelo CAPM

O modelo CAPM assume que todos os investidores usam o modelo de Markowitz para tomada de decisão, que existe um ativo sem risco acessível a todos os investidores, que podem tomá-lo emprestado ou nele investir com uma mesma taxa de retorno r_0, e que todos os investidores estão de acordo quanto ao retorno esperado e a matriz de covariância dos retornos dos ativos de risco no mercado.

A primeira conclusão a que chegamos neste cenário é que todos os investidores concordam sobre a fronteira eficiente do mercado. Ademais, como no modelo de Tobin, todos os investidores formam seu portfólio sobre a *linha de mercado*, a partir do ativo sem risco e de um mesmo portfólio tangente, o *portfólio de mercado*, xm.

Sob condições de equilíbrio, todas os ativos deveriam estar presentes no portfólio de mercado (pois ninguém teria interesse num ativo fora de xm). Examinemos agora a combinação convexa de xm com o portfólio xi constituído apenas do i-ésimo ativo, x_i:

$$xc(\lambda) = (1-\lambda)xi + \lambda xm \ , \quad \text{com } e(\lambda) \equiv e(xc(\lambda)'r) \text{ e } \sigma_{\lambda,\lambda} \equiv Var(xc(\lambda)'r)$$

$xc(1)$ coincide com xm, e para todos os demais valores de λ, a solução $xc(\lambda)$ é dominada pela fronteira eficiente; i.e., a curva $e(xc(\lambda)) \times \sigma(xc(\lambda))$ tangencia a fronteira eficiente.

Mas

$$\begin{aligned} e(\lambda) &= (1-\lambda)e(xi) + \lambda e(xm) \\ \frac{\partial e(\lambda)}{\partial \lambda} &= e(xm) - e(xi) \\ \sigma_{\lambda,\lambda} &= (1-\lambda)^2\sigma_{xi,xi} + 2\lambda(1-\lambda)\sigma_{xi,xm} + \lambda^2\sigma_{xm,xm} \\ \frac{\partial \sigma_\lambda}{\partial \lambda} &= \frac{-(1-\lambda)\sigma_{xi,xi} + (1-2\lambda)\sigma_{xi,xm} + \lambda\sigma_{xm,xm}}{\sigma_\lambda} \end{aligned}$$

Igualando, em $\lambda = 1$ a derivada, $\partial e(xc)/\partial\sigma(xc)$ à inclinação da linha de mercado temos

$$\frac{(e(xm) - e(xi))\sigma_{xm}}{\sigma_{xm,xm} - \sigma_{xi,xm}} = \frac{e(xm) - r_0}{\sigma_{xm}}$$

simplificando,

$$e(xi) = r_0 + (e(xm) - r_0)\sigma_{xi,xm}/\sigma_{xm,xm}$$

No modelo de índices, com um único índice, tinhamos

$$r_i = \alpha_i + \beta_i c$$

Nesta situação as condições de equilíbrio se traduzem como

$$\alpha_i = r_0 \ , \quad c = e(xm) - r_0 \ , \quad \beta_i = \frac{\sigma_{xi,xm}}{\sigma_{xm,xm}} = \frac{e(xi)}{e(xm) - r_0}$$

Vemos portanto que o modelo CAPM espera de cada ativo um retorno igual ao do ativo sem risco, mais um retorno proporcional a taxa de crecimento da economia; este fator de proporcionalidade é por sua vez proporcional a covariância do ativo com o portfólio de mercado.

Modelo APT

O modelo APT é um modelo de equilíbrio baseado no modelo de índices, que supõem ser possível construir "portfólios de arbitragem" com as seguintes características:

1. $x'\mathbf{1} = 0$,
2. $x'B = 0$,
3. $Var(x'a) \approx 0$.

Isto é, portfólios de valor zero, *imunes* a todas as taxas de crescimento setorial. e quase livres de risco próprio.

A condição de equilíbrio do modelo é: "As condições acima implicam que a taxa esperada de retorno de um portfólio de arbitragem deve ser nula, $x'E(a) = 0$". Esta condição de equilíbrio é motivada pela possibilidade de ganhar dinheiro sem risco, caso a condição não seja satisfeita. Matematicamente, para qualquer portfólio de arbitragem, a condição de equilíbrio se expressa como:

$$x' \left[\begin{array}{cc} 1 & B \end{array} \right] = 0 \Rightarrow x'E(r) = 0$$

Mas a implicação acima é equivalente a termos

$$E(r) = \left[\begin{array}{cc} 1 & B \end{array} \right] \left[\begin{array}{ccc} l_0 & l_1 & \dots l_k \end{array} \right]'$$

Assim o modelo APT conclui que, em condições de equilíbrio, o retorno esperado de um dado ativo, $E(r_i)$, é uma constante, l_0, mais termos proporcionais aos *prêmios de risco* de cada setor, l_k ponderados pelo fator de sensibilidade, $B_{i,k}$. Finalmente, usando novamente um argumento de arbitragem, concluímos que a constante l_0 deve ser a taxa de retorno do ativo sem risco, r_0.

Capítulo 5

Programação Dinâmica

Neste capítulo procuraremos introduzir conceitos básicos de programação dinâmica e apresentar exemplos de resolução de problemas. Programação dinâmica é uma metodologia que pode ser empregada para resolver uma extensa classe de problemas. Mais que uma técnica, é uma abordagem conceitual que procura subdividir o problema de interesse em subproblemas de mais simples resolução, estabelecendo uma relação de dependência entre os diversos subproblemas. Essa relação possibilita que a solução ótima do problema original seja obtida recursivamente: em cada estágio de resolução busca-se a solução ótima de um subproblema que depende apenas da solução ótima do subproblema anteriormente resolvido.

O objetivo de um problema de programação dinâmica ou controle ótimo é otimizar as ações de um operador capaz de influenciar o sistema em estudo. Procura-se determinar uma regra de decisão ótima, ou seja, que minimize o custo total associado à tomada de decisão. Discutiremos a questão assumindo dois agentes: o operador que tem à sua disposição um conjunto de controles, ou ações, para atuar sobre o sistema, e o administrador do sistema a quem cabe estabelecer uma política que determine o controle a ser usado pelo operador em função do estado do sistema.

A evolução do sistema ao longo do tempo é descrita por uma equação relacionando o estado em que o mesmo se encontrará no instante t+1 e seu histórico passado e presente; também nesta descrição considera-se o conjunto de ações tomadas pelo operador dentro do horizonte de tempo em questão, i.e., os controles exercidos nos instantes presente e passado.

5.1 Conceitos Básicos

Os conceitos apresentados a seguir serão utilizados neste capítulo e sintetizam alguns dos principais aspectos de programação dinâmica.

1 **Tempo, estágio, período, época, instante**: é um parâmetro de evolução do sistema. Muitas vezes representa tempo, mas nem sempre esta interpretação é a mais adequada. Pode ser inerente à natureza do problema mas com freqüência é introduzido artificialmente para subdividir o problema original em subproblemas.

2 **Horizonte:** um limitante superior para o índice de tempo t. Neste trabalho, o domínio do tempo t será sempre discreto, uma consequência da tônica algorítmica e computacional adotada. Geralmente tem a forma $t \in \{0, 1, 2, \ldots, h\}$ ou $t \in \{1, 2, \ldots, h\}$, onde h é o horizonte (de planejamento). Podemos ter $h = +\infty$.

3 **Estado:** descrição das condições do sistema no instante t, denotada por $x(t)$. Quando não houver ambigüidade, o instante t em discussão será omitido.

4 **Ação, decisão, controle:** $u(t)$. Descreve a influência do operador sobre o sistema. Otimizar as ações do operador é o objetivo de um problema de programação dinâmica ou de controle ótimo.

Neste livro veremos exemplos de espaços de estado e controle discretos e contínuos, ainda que utilizemos somente parâmetro de tempo discreto.

5 **Equação de Evolução Dinâmica:** É a equação que nos dá o próximo estado do sistema em função dos estados e controles presentes e passados, $x(t+1) = g(x(t), u(t), x(t-1), u(t-1), \ldots), x(1), u(1)$.

6 **Trajetória:** uma possível seqüência de estados e controles, do instante inicial ao horizonte, $x(1), u(1), x(2), u(2), \ldots x(h)$.

7 **Evolução sem memória, Sistema Markoviano:** $x(t+1) = g(x(t), u(t))$ é função apenas do estado e controles presentes. Muito conveniente em programação dinâmica, sendo que muitas vezes adotamos uma descrição redundante de estados apenas para obter markovianidade. A grande maioria dos problemas de programação dinâmica são markovianos ou são tais que a função $g(.)$ depende de um número finito de estados passados. Neste último caso $x(t+1) = g(x(t), u(t), x(t-1), u(t-1), \ldots, x(t-n), u(t-n))$, isto é, a equação de evolução é uma equação de diferenças finitas de ordem n+1.

8 **Restrições:** condições especificando as trajetórias viáveis (admissíveis) para o sistema, geralmente expressas na forma $R(x(0), u(0), \ldots, x(h), u(h)) \geq 0$. Geralmente estudaremos restrições independentes do passado, $R(t, x(t), u(t)) \geq 0$.

9 **Custo instantâneo:** $c_t = c(t, x(t), u(t))$, o custo do controle $u(t)$ estando o sistema no estado $x(t)$.

10 **Custo aditivo, custo descontado, custo médio:** a soma, a soma descontada, ou a média dos custos instantâneos ao longo de uma trajetória (caminho

de evolução) do sistema. $S = \sum_{t=1}^{h} c_t$, $S = \sum_{t=1}^{h} \beta^t * c_t$, ou $S = (1/h)\sum_{t=1}^{h} c_t$, onde $c_t = c(t, x(t), u(t))$.

11 **Função de custo:** uma função $f(x(0), u(0), x(1), u(1), \ldots, x(h), u(h))$ que pretendemos minimizar. Geralmente $f()$ é uma função simples do custo aditivo ou descontado. Muitas vezes utilizamos truncamentos da função de custo (funções de custo futuro), $f(x(t), u(t), x(t+1), u(t+1), \ldots, x(h), u(h))$, para sub-dividir o problema em problemas auxiliares.

12 **Política:** Uma regra para tomada de decisão. Se o sistema se encontra em um estado $x(t)$, a adoção de uma política Π gera uma ação ou controle $u(t)$, dado por $u(t) = \Pi(t, x(t), u(t-1), x(t-1), \ldots, u(0), x(0))$. Geralmente estudaremos políticas que dependem apenas do estado presente, $u(t) = \Pi(t, x(t))$. Uma política é viável se respeita as restrições do sistema.

13 **Política ótima:** a política viável ótima é aquela que minimiza a função de custo

14 **Princípio de otimalidade:** uma prova (argumento) de que a política ótima é função apenas do estado presente. Note que a markovianidade do sistema não implica num princípio de otimalidade, sendo fácil apresentar contra-exemplos com funções de custo não aditivas.

15 **Condição de contorno:** base para a aplicação recursiva da equação de Bellman. Estabelece o valor da função custo no horizonte de planejamento.

16 **Equação de Bellman ou equação de otimalidade:** uma afirmação sobre a política ótima, baseada no princípio de otimalidade e que nos permite calcular a solução ótima do problema recursivamente. Muitas vezes a equação de Bellman não é montada diretamente sobre a política ótima, mas sobre o ótimo da função custo numa série de problemas auxiliares.

17 **Recuperação da política ótima ou backtracking**: recuperação da política ótima a partir da tabela de valores calculados com a equação de Bellman.

18 **Equação de evolução em sistemas estocásticos**: a evolução de um sistema estocástico markoviano é dada pela probabilidade (distribuição de probabilidade) de transição do sistema para o estado $x(t+1)$, a partir do estado $x(t)$ com controle $u(t)$, descrita como $pr(t, x(t), u(t), x(t+1))$.

19 **Função de custo em sistemas estocásticos**: nestes sistemas adota-se ao invés da função custo, a esperança das mesmas. Assim, as funções mais utilizadas em sistemas estocásticos são:

1. O valor esperado do custo aditivo (ou descontado): $f_\Pi = E(S = \sum_{t=0}^{h} c(t, x, u))$.

2. Uma combinação linear do valor esperado e da variância do custo aditivo: $f_{\Pi} = E(S) - Var(S)$.
3. A esperança de uma função do custo aditivo ou descontado, como por exemplo no controle quadrático Gaussiano sensível a risco (ELQG):

$$\begin{aligned} f_{\Pi} &= (-2/\theta) \log E(\exp(-\theta S/2)) \\ &= E(S) + (\theta/4) Var(S) + O(\theta^2) \end{aligned}$$

5.2 Distância Mínima em um Grafo

Um problema clássico em teoria dos grafos, denominado problema do caminho mínimo, possui diversas aplicações tanto em engenharia como em finanças. Iremos resolvê-lo dentro do contexto de programação dinâmica.

Primeiramente apesentaremos algumas definições elementares:

Um **grafo**, $G = (X, U)$, é uma estrutura composta de um conjunto de **vértices**, $X = \{1, \ldots, n\}$, e um conjunto de arestas $U \subseteq X \times X$. Imagine vértices como sendo cidades, e uma aresta $u = (x, y)$ como uma estrada de mão única indo de x para y.

Um grafo é **completo** se contém todas as n^2 arestas possíveis.

Um **caminho** é uma sequência de arestas consecutivas,

$$p = ((v_0, v_1), (v_1, v_2), (v_2, v_3), \ldots, (v_{t-1}, v_t)).$$

Um **ciclo** é um caminho que começa e termina no mesmo vértice, $v_0 = v_t$, sem repetir nenhum outro vértice.

O **tamanho** de um caminho, $|p|$, é o número de arestas que o compõem. Denotamos por $p(s) = v_s$ o s-ésimo vértice em p.

O **custo** é uma função $c(u)$, $U \mapsto \mathcal{R}$. O custo aditivo de um caminho é a soma dos custos das arestas que o compõem, i.e. $c(p) = \sum_1^t c((v_{t-1}, v_t))$.

O problema que queremos resolver é determinar $p^*(1, y)$, o caminho de mínimo custo de 1 até y. Assumiremos válidas as seguintes hipóteses:

- O custo de uma aresta nunca é negativo.
- O grafo é completo.

A primeira hipótese simplifica grandemente o problema. Já a segunda é apenas uma conveniência de modelagem e pode ser assumida sem perda de generalidade uma vez que arestas inexistentes podem ser modeladas por arestas de custo muito alto, ou $+\infty$.

Para resolver o problema do caminho mínimo iremos utilizar o princípio de programação dinâmica, construindo problemas auxiliares. No t-ésimo subproblema queremos determinar um caminho de no máximo t arestas que minimize a distância entre o vértice (estado) origem, 1, e o destino y. O índice de "tempo" t neste caso indicará os subproblemas construidos e estará associado ao tamanho dos caminhos construidos. A decisão a ser tomada em cada estado (cada vértice do grafo) consiste em determinar a próxima aresta que irá compor o caminho mínimo.

Vamos subidividir o problema considerando vários subproblemas nos quais iremos determinar um caminho de custo mínimo entre o vértice (estado) 1 e o vértice y com a restrição de conter no máximo t arestas.

Ou seja, para $t \in \{0, 1, 2, \ldots\}$ consideramos:

Problema t: Encontre o mínimo custo aditivo, para um caminho de 1 até y de tamanho menor ou igual a t, i.e. determine a função de custo

$$f^*(t, 1, y) = \min_{p \in \bar{P}(t,1,y)} c(p)$$

onde

$$P(t, 1, y) = \{p| \ |p| = t, p(0) = 1, p(t) = y\} \ \text{ e } \ \bar{P}(t, 1, y) = \cup_{s=0}^{t} P(s, 1, y)$$

Caso não exista uma caminho de 1 a y de tamanho menor ou igual a t temos, pela definição de min, $f^*(t, 1, y) = +\infty$.

O primeiro problema auxiliar tem solução trivial:

$$f^*(t = 0, 1, 1) = 0 \ \ e \ \ f^*(t = 0, 1, y) = +\infty \ , \ \ \forall y \neq 1$$

.

Temos também a relação de recorrência:

$$f^*(t + 1, 1, y) = \min_{u=(x,y)}\{f^*(t, 1, y), (f^*(t, 1, x) + c(x, y))\}.$$

Esta recorrência se explica da seguinte forma: Qualquer caminho $p \in P(t, 1, x)$ seguido da aresta $u = (x, y)$ resulta num caminho em $P(t + 1, 1, y)$, e qualquer caminho em $P(t + 1, 1, y)$ pode ser obtido exatamente desta forma (por que?).

Estabelecida a recorrência podemos usar a solução do problema auxiliar com $t = 0$ e a relação de recorrência para determinar os ótimos da função de custo, $f^*(t, 1, y)$, para $t \in \{1, 2, \ldots\}$. Em particular, a primeira aplicação da relação de recorrência nos fornece:

$$f^*(1, 1, y) = c((1, y)), \forall y \in X \ .$$

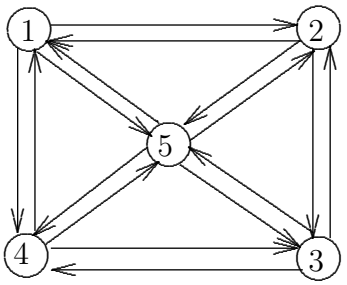


Figura 5.1: Um grafo orientado

Na figura 4.1 temos um grafo com 5 vértices. Seus custos são apresentados na Tabela I abaixo:

Tabela I

$C(i,j)$	1	2	3	4	5
1	0	2	∞	1	2
2	2	0	2	∞	0
3	∞	2	0	3	4
4	1	∞	3	0	0
5	2	0	4	0	0

Na tabela abaixo temos o exemplo de aplicação da recorrência para $t \in \{0, 1, \ldots, n-1\}$ no grafo da figura 4.1.

$t \backslash y$	1	2	3	4	5
0	**0**	∞	∞	∞	∞
1	0	2	∞	**1**	2
2	0	2	4	1	**1**
3	0	**1**	4	1	1
4	0	1	**3**	1	1

Tabela II

Lema 1: $\forall t \geq n, f^*(t, x, y) = f^*(n-1, x, y)$

Prova: Considere p_t^* o caminho de mínimo custo de x a y com tamanho $t \geq n$. Provemos que existe um caminho de tamanho $t-1$ com custo menor ou igual ao custo de p_t^*. Como $t \geq n$, e só ha n vértices no grafo, pt contêm ao menos um ciclo. Como o custo deste ciclo é não negativo, basta removê-lo para obter um caminho como o desejado. Disto segue trivialmente o lema (por que?).

QED

Pelo **Lema 1** sabemos que, $f^*(n-1,1,y)$, calculado na última linha da Tabela II, é $f^*(1,y)$, o custo da solução ótima do problema original! De posse da Tabela I, e de uma calculadora de mão, é fácil **recuperar** (backtrack)

$$p^* = \text{argmin}_{p\in\bar{P}(n,1,y)} c(p),$$

o caminho que realiza este mínimo. Na Tabela I temos o caminho ótimo para um dado destino traçado nos valôres ótimos do custo futuro, f^*, em **negrito**.

Explique exatamente como foi possível traçar este caminho na tabela pronta. Seria possível ir traçando o caminho ao longo da construção da tabela, i.e. após o cálculo de cada linha? Como este traçado na tabela efetivamente nos dá o caminho ótimo, assinalado na figura-1?

O programa mindist.m implementa em Matlab algoritmo de mínima distância. Estude-o e escreva o algoritmo de backtraking.

Algoritmo de Dijkstra

O programa dijk.m implementa em MATLAB o cálculo da tabela de custos ótimos. O algoritmo de Dijkstra é uma formulação mais eficiente, requerendo apenas da ordem de n^2 operações aritméticas e espaço de memória, contra respectivamente n^3 e n^2 no primeiro programa.

Tente explicar o algoritmo de Dijkstra em dijk.m e justificar sua corretude. Dica: Ao término da fase $t = 0, 1, \ldots, n-1$ temos ao menos $t+1$ vértices para os quais o custo do caminho ótimo já foi determinado corretamente. Estes são os índices não mais presentes no conjunto *inder*.

5.3 Cadeias de Markov e Custo Esperado

A natureza dos problemas de finanças nos leva a considerar modelos estocásticos: sejam preços, taxas, volumes ou outras as variáveis de decisão, devemos levar em conta a incerteza inerente a elas. Há várias maneiras de abordar esta questão: uma delas é interpretar algumas variáveis de decisão como variáveis aleatórias, estudando sua distribuição de probabilidade. Outra forma de inserir aleatoriedade nos modelos é considerar que há uma probabilidade conhecida de mudança de estado no sistema em questão. Essa segunda abordagem nos permite utilizar os princípios de programação dinâmica apresentados anteriormente.

Muitos problemas em finanças procuram determinar uma sequência de decisões ótimas ao longo de um horizonte de planejamento. Em geral a tomada de decisão em um dado instante depende do estado em que o sistema se encontra naquele particular momento e da ação, ou controle, que será exercido no sistema a partir de então. Os modelos de programação dinâmica são facilmente empregáveis nestes casos e possibilitam que seja introduzida aleatoriedade no problema.

Neste capítulo apresentaremos uma classe de modelos denominada cadeias de Markov, procurando inserí-los no contexto de programação dinâmica. Como no capítulo precedente iremos procurar construir uma expressão recursiva que forneça a ação ótima a ser tomada em cada período, considerando que há um custo associado a cada uma das possíveis ações. Nos exemplos que serão apresentados, teremos como meta minimizar o custo esperado.

Decisão por custo esperado

Consideremos um sistema com estados $x \in X = \{1, 2, \ldots\}$, evoluindo num parâmetro discreto $t \in T = \{1, 2, \ldots, h\}$, ou seja, $x(t) \in X$. A cada instante t tomamos uma ação $u(t) \in U = \{0, 1, 2, \ldots\}$. A ação tomada é função da política Π adotada, isto é, de uma regra de decisão previamente estabelecida.

A cada controle $u(t)$, associamos um custo determinístico $c_x^u(t)$, que é função do estado, do controle e do instante de tempo considerado. Consequentemente podemos atribuir a cada trajetória $(x(1), u(1), x(2), u(2), \ldots, x(h), u(h))$ um custo aditivo dado por $\sum_{t=1}^{h} c_x^u(t)$

A aleatoriedade no sistema é introduzida através de probabilidades de mudança de estado. Assim, definimos $P_{x,y}^u(t)$ como a probabilidade de transição do estado $x(t)$ para o estado $y = x(t+1)$, e obviamente tomamos $\sum_y P_{x,y}^u(t) = 1, \forall x, u, t$. Com isto podemos falar no custo esperado associado a uma política, pois fixar uma política determina a probabilidade de cada trajetória possível.

Assim, definimos o custo associado a uma política Π, como sendo o valor esperado do custo aditivo, a partir de um estado inicial $x(t)$:

$$f_\Pi(1, x(1)) = E(\sum_{t=1}^{h} c_x^u(t))$$

Igualmente podemos construir, para uma fixada política e um dado instante t, a função de custo futuro,

$$f_\Pi(t, x(t)) = E(\sum_{s=t}^{h} c_x^u(s)),$$

o custo esperado futuro a partir do estado $x(t)$. Definimos também o ótimo da função de custo futuro como $f^*(t, x) = \min_\Pi f_\Pi(t, x)$. Queremos determinar a política ótima Π^* que minimize a função de custo.

Uma expressão recursiva para o cálculo do ótimo da função é facilmente obtida:

- Condição de contorno: para t no horizonte h, temos $f^*(h,x) = \min_u c_x^u(h)$.
- Para $t \in \{h-1, h-2, \ldots 1\}$ vale a relação de recorrência

$$f^*(t,x) = \min_u \{c_x^u(t) + \sum_{y \in X} P_{x,y}^u(t) * f^*(t+1,y)\}$$

Como no problema de mínima distância num grafo, a partir da condição de contorno (neste caso no horizonte h) e da equação de Bellman (recorrência sobre a função f()), é possível resolver todos os problemas auxiliares, e a partir destes recuperar a solução ótima do problema original.

5.4 Hedging

Ilustremos a técnica de controle em cadeias de markov em um problema concreto. O problema em questão é anedótico, mas ilustra o uso de estoques reguladores como forma de Hedging, ou proteção contra as ocilações de preço ou demanda no mercado.

O Bigode, que vende cachorro-quente na porta da faculdade de economia, tem uma demanda constante de uma caixa de salsicha (uma unidade) por dia. O Bigode vende esta unidade a um preço constante de US$ 120, todo dia. Ele compra esta unidade no mercado a um preço que varia aleatoriamente, sendo que o mercado abre cada dia no estado m, onde a unidade custa $v(m)$, com probabilidade $p(m)$ conforme descrito na tabela abaixo.

m	1	2	3
v(m)	90	100	110
p(m)	0.2	0.7	0.1

Além da unidade sendo vendida no dia, o Bigode tem em sua casa um estoque de $l(t)$ unidades. O semestre letivo tem $h+1$ dias consecutivos e, exceto no último dia $t = h+1$, o Bigode tem que decidir quantas unidades comprar, $u(t)$. Deste modo, $l(t+1) = l(t) + u(t) - 1$. A decisão sobre quanto comprar é tomada estando ciente do estado $x(t) = (l(t), m(t))$.

Como o Bigode não pode faltar com seus clientes, nem guardar este produto perecível por mais de 4 dias, impomos as restrições $0 \leq l(t) \leq 3, \forall 1 \leq t \leq h$. Para maximizar seus lucros, o Bigode quer uma política (viável) $u(t) = \Pi^*(t, l(t), m(t))$ que minimize a esperança de seu custo operacional durante o semestre.

Com a política ótima, Π^*, o custo operacional esperado, do dia t até o final do semestre, é (equação de Bellman):

$$f^*(t, l(t), m(t)) = \min_{u | 0 \leq l+u-1 \leq 3} (u * v(m) + \sum_{n=1}^{3} p(n) * f^*(t+1, l+u-1, n))$$

O resultado da política ótima no último dia de compra do semestre, $t = h$, é (condição de contorno):

$$f^*(h, l, m) = \quad 0, \quad \text{se} \quad 1 \leq l \leq 3; \quad v(m), \quad \text{se} \quad l = 0.$$

5.5 Custo Descontado

Em muitas situações reais existem oportunidades de investimento em cada período com uma taxa de juro $r(t)$. Na presença desta oportunidade, para pagar o custo $c_x^u(t)$ do controle $u(t)$ no instante t bastaria tomar no instante $t-1$ o valor presente deste custo, ou seja, $\delta(t-1)c_x^u(t)$, onde $\delta(t-1) = \frac{1}{1+r(t-1)}$

Nestas condições gostaríamos de substituir, como critério de avaliação de uma política, o custo esperado pelo custo descontado esperado:

$$f_{\Pi}(1, x(1)) = E(\sum_{t=1}^{h} \left[\prod_{s=1}^{(t-1)} \delta(s) \right] c_x^u(t))$$

A equação de Bellman (relação de recorrência) é facilmente adaptada para incorporar o fator de desconto:

$$f^*(t, x) = \min_u \{c_x^u(t) + \delta(t) \sum_{y \in X} P_{x,y}^u(t) * f^*(t+1, y)\}$$

No caso de uma taxa de desconto constante estas equações se reduzem respectivamente a:

$$f_{\Pi}(1, x(1)) = E(\sum_{t=1}^{h} \delta^{(t-1)} c_x^u(t))$$

$$f^*(t, x) = \min_u \{c_x^u(t) + \delta \sum_{y \in X} P_{x,y}^u(t) * f^*(t+1, y)\}$$

Como o custo presente dos controles no horizonte de planejamento diminuem exponencialmente com h, a política ótima inicial, $\Pi(1, x)$ $u(1)$, tende a convergir para um controle $u(1)$ a medida que o horizonte se distancia (i.e. que h aumenta). Pela mesma razão, para horizontes longínquos, os controles ótimos iniciais tendem a ser pouco sensíveis às condições de contorno. Ambas as características tornam geralmente mais robustas as modelagens pelo critério de custo descontado.

5.6 Precificação de Contratos Derivativos

Em um contrato de opção de compra (venda) o contratante adquire do contratado o direito de comprar (vender) uma mercadoria do (ao) contratado por um preço de exercício pré-estabelecido, K. Note que o contratante adquire o direito de fazer uma compra (venda), sem ter a obrigação de faze-la (ao contrário de um simples contrato futuro de compra ou venda), daí o nome contrato de opção.

Existem duas modalidades básicas de contrato de opção: em um contrato de tipo Americano o contratante tem o direito de exercer sua opção a qualquer instante t desde a assinatura do contrato até até sua data de vencimento, $0 \leq t \leq h$. No contrato de tipo Europeu a opção pode somente ser exercida na data de vencimento do contrato. Um contrato opção de compra (venda) é tambem denominado um call (put). O preço pelo qual o contratante adquire o call (put) é o preço do contrato, C (P).

A mercadoria especificada no contrato é chamada ativo fundamental do contrato. O contratado recebe uma remuneração para absorver os riscos decorrente das variações de preço do ativo fundamental, e é por isto denominado especulador. O contratante paga para se proteger contra os mesmos riscos, e é por isto denominado hedger. Contratos de opção são denominados ativos derivativos, pois os ganhos e perdas de ambas as partes resultam (derivam) da evolução do preço do ativo fundamental.

O call ou put deve especificar a mercadoria em termos de quantidade, local de entrega (praça) e qualidade. Ambas as partes estabelecem um intermediador idôneo, por exemplo uma bolsa de mercadorias, para dirimir quaisquer duvidas. No exercício de uma opção teriamos então a entrega física da mercadoria na praça especificada com verificação, por parte da bolsa, da quantidade e qualidade.

A mercadoria especificada no contrato é muitas vezes uma mercadoria amplamente comercializada, não importando o fornecedor (uma commoditie), e cujo preço de mercado, $S(t)$, é tambem divulgado pela bolsa. Nestas condições, é geralmente mais cômodo para ambas as partes substituir a entrega física por um acerto de contas em função da diferença entre o preço de exercício e o preço de mercado da comoditie na data do exercício. Por exemplo, num call, o vendedor da opção (contratado ou especulador) simplesmente pagaria ao comprador da opção (contratante ou hedger) o valor $max\{0, (S(t) - K)\}$. No caso de uma put este valor seria $max\{0, (K - S(t))\}$.

Para colocar os contratos de opções no contexto de programação dinâmica em cadeias de programação dinâmica é preciso modelar a evolução do preço $S(t)$ por um processo estocástico discreto. No modelo de passeio aleatório trinomial temos que o preço poderia assumir apenas três valores: $Pr(S(t+1) = fu(t) * S(t)) = pu(t)$, $Pr(S(t+1) = fd(t) * S(t)) = pd(t)$, $Pr(S(t+1) = S(t)) = 1 - pu(t) - pd(t)$, onde $fu(t)$, $fd(t)$, $pu(t)$ e $pd(t)$ são, respectivamente, os fatores de subida e descida do preço fundamental entre os instantes t e $t+1$, e suas probabilidades. No modelo binomial temos $pu(t) + pd(t) = 1$. No modelo

geométrico há uma simetria entre os fatores de subida e descida, $fd(t) = 1/fu(t)$. Esta hipótese de simetria, que é motivada por certas considerações econômicas e empíricas, em muito facilita a representação do espaço de estados. No modelo estacionário os fatores e suas probabilidades são constantes no tempo. Assim no modelo binomial geométrico temos apenas os parâmetros $fu(t)$ e $pu(t)$. Mesmo num modelo tão simples podemos retratar um processo onde a taxa
$r(t) = ln(S(t+1)/S(t))$ tem valor esperado e desvio padrão (tendência e volatilidade)

$$\mu(t) = pu(t)*fu(t)+(1-pu(t))/fu(t) \text{ e } \sigma(t) = \sqrt{pu(t)*(1-pu(t))*(fu(t)-1/fu(t))^2}$$

Estes conceitos podem ser colocados dentro da linguagem de programação dinâmica, como pode ser visto nos exercícios abaixo:

1. Escreva um programa para ajudar o bigode a gerenciar sua empresa.

2. Call Americano de salchicha: Modifique o programa do Bigode para um mercado de salchicha onde o preço segue um passeio aleatório trinomial geométrico simétrico e estacionário, definido pela tendência e volatilidade diárias ms e ss. Admita que um especulador ofereça ao Bigode no primeiro dia do semestre a oportunidade de comprar por C um (1) call americano para Q caixas, com preço de exercício K, expirando no final do semestre. Considere que as caixas a serem entregues no eventual exercício da opção são do tipo longa-vida, que podem ser armazenadas até o final do semestre, independentemente do estoque regular. O espaço de estados do modelo deve incluir, cotidianamente, o preço do mercado, o estoque regular, o estoque longa-vida, e a posse ou não do call não exercido. O espaço de controle deve incuir, cotidianamente, a possibilidade do Bigode se abastecer no mercado até o limite do estoque regular, e a possibilidade de exercitar o call caso o Bigode o possua. No primeiro dia o Bigode deve decidir também pela compra ou não do call. Queremos como saída deste programa de programação dinâmica a decisão ótima no primeiro dia pelo critério do custo descontado com taxa de desconto diária δ, e também a rendabilidade esperada do negócio do Bigode.

3. Escreva um programa de precificação, que utilizando o método da bisecção e invocando o programa de programação dinâmica, determine valor C a partir do qual a call deixa de ser interessante para o Bigode.

4. Adapte o programa de programação dinâmica, para uma taxa de desconto $\delta(t)$, que tem valor inicial $\delta(1)$ e segue um passeio aleatório trinomial geométrico estacionário, definido pela tendência e volatilidade diárias md e sd.

   O programa Bigopt.m, listado em apêndice, resolve parcialmente o exercício 1.

5.7 Políticas de Scarf (s,S)

Diferentes características ou objetivos parciais de um sistema produtivo típico sugerem tendências divergentes para a política de estoques. Por exemplo: grandes tempos de set-up (tempos de que decorrem da necessidade de ajustar ou limpar o equipamento antes iniciar ou depois de terminar a prudução de um lote de um dado produto), ou perda de vendas e mercado por demora no atendimento ao cliente favorecem a formação de grandes estoques; enquanto altos custos financeiros e de manutenção favorecem o trabalho com pequenos estoques.

Para encontrar soluções ótimas nesta situação paradoxal, que geralmente são soluções intermediárias entre as políticas extremas, é necessário equacionar e otimizar um modelo quantitativo. Discutiremos agora uma classe de modelos para otimizar políticas de planejamento de produção e estoques em uma empresa industrial ou comercial. Consideraremos a demanda como uma variável aleatória, a ser especificada por uma distribuição de probabilidade a partir das previsões de demanda.

A decisão (ou controle), em cada período n, é a quantidade de produto a ser produzida, $y(n)$. Sob certas condições é possivel demonstrar que o controle ótimo é caracterizado pelo par "ponto de recompra – patamar de recomposição": caso o estoque seja menor ou igual ao ponto de recompra, s_n, devemos recompor o estoque, comprando ou produzindo $y(n)$, até o patamar S_n. Estas são as políticas de Scarf.

A estrutura básica dos modelos (s, S) é a seguinte:

1. Um horizonte finito de planejamento, i.e. N períodos (meses, semanas) de produção, $n \in \{1, 2, \ldots N\}$. Em cada período "herdamos" um estoque $x(n)$ do período $n-1$, produzimos para aumentar o estoque até o nível $y(n)$, e finalmente, após as vendas do período, o estoque baixa para $z(n)$.

2. $d(n)$, a demanda no período n.

3. $b(n)$ a fração dos clientes recusa espera no atendimento.

4. A equação básica de evolução do sistema é a equação de evolução dos estoques, $z(n, y, d)$. Diferentes equações de evolução de estoques modelam, vários aspectos pertinentes; por exemplo:

$$z(n, y, d) = (y - d) + b(n) * (d - y)^+$$

   Para qualquer número real x definimos x^+ como sendo a parte positiva de x, isto é, $x^+ = max\{x, 0\}$. Estoques negativos representam pedidos não atendidos em espera.

5. $k(n)$, o custo constante de set-up para produção.

6. $w(n)$, o custo unitário de compra ou produção.

7. $r(n)$, o preço unitário de venda. O estoque remanecente do último período é "liquidado" por um preço $r(N+1)$.

8. $h(z,n)$ o custo de manter o estoque $z(n)$. Para valores negativos de z, este custo representa uma penalidade pela espera do cliente.

   Assim o custo instantâneo no período n será dado por

$$c(n) = w(n) * (y - x) + k(n) * (y - x)^{>} + h(z(n, y, d)) - r(n) * (y - z)$$

   Para qualquer número real x definimos $x^{>}$ como 1 se $x > 0$ e 0 caso contrário.

9. $a(n)$, o fator de desconto intertemporal computado a partir das taxas de inflação e juros no período, nos define o custo presente de operar o sistema ate o horizonte de planejamento (custo aditivo descontado):

$$f = \sum_{n=1}^{N} \left(c(n) * \prod_{l=1}^{n} a(l) \right) .$$

A partir desta descrição do sistema podemos computar uma política ótima de estoques na variável de decisão, $y(n)$, através de programação dinâmica no contexto de cadeias de Markov.

Ilustramos agora a semântica do modelo de Scarf num contexto anedótico. Consideremos um lojista que vende esquis em Bariloche. Ele pode fazer pedidos à fábrica (em Buenos Aires) semanalmente pagando um custo fixo de k = US$100,00 ao motorista, mais um custo unitário de US$100,00 por esqui pedido. Cada esqui é revendido em Bariloche por r = US$150,00 , mas b = 30% dos clientes se recusa a esperar caso não haja mercadoria em estoque. A taxa de juros semanal na temporada é de 2%. A temporada turística de inverno é de N=10 semanas, finda a qual é feita uma liquidação de estoque ao preço de r(N+1)= US$ 80,00. Assumiremos que não há custo de manutenção de estoques e analisaremos duas demandas: a primeira determinística de $\bar{d} = 10$ unidades por período (semana) e a segunda estocástica com a seguinte distribuição:

d	7	8	9	10	11	12	13
probab	0.05	0.10	0.20	0.30	0.20	0.10	0.05

, $\bar{d} = 10$.

As primeiras duas linhas da tabela seguinte apresentam a ordem de compra ótima para a primeira semana, sem estoque inicial. VP é o valor presente esperado do resultado

operacional do negócio. Δ é o decréscimo esperado deste resultado, caso no primeiro período seja feita a ordem sub-ótima de $\bar{d} = 10$ unidades. Os valores n.Δ nos dão uma estimativa da perda de resultado para ordens iguais à demanda esperada, $\bar{d} = 10$, em processo contínuo. As colunas referentes a s1 e S1 são respectivamente o ponto de recompra e o patamar de recomposição no primeiro período.

Nas linhas seguintes da tabela estudamos como estes resultados são afetados pela alteração de alguns dos parâmetros do modelo. Nas últimas duas linhas estudamos a introdução de um custo de falta de 10% do valor de venda por unidade.

Dentro da notação apresentada temos a seguinte situação inicial:

$N = 10$

$b(n) = 0.3$

$k(n) = \text{US\$ } 100.00$

$r(n) = \text{US\$ } 150.00$

$a(n) = 0.98$

$r(N+1) = \text{US\$ } 80.00$

$w(n) = \text{US\$ } 100.00$

$hn/r = 0$

Da comparação dos resultados no cenários determinístico e estocastico, vemos que é mais facil obter lucros no mundo determinístico (maior valor presente), estando todavia no mundo com incertezas o maior valor de planejamento (maior Δ). Ademais, a análise de sensibilidade aos diversos fatores nos mostra que as seguintes alterações podem induzir a um aumento no nível ótimo de estoques: Aumento do lucro unitário, diminuição da taxa de juros, aumento de liquidez (um mercado com clientes cativos é denominado viscoso, enquanto um mercado com grande disponibilidade de fornecedores é denominado líquido), aumento de custo de set-up, e finalmente, aumento do custo de mal atendimento.

Solução Ótima do Modelo Básico Alterado						
Alteração	Demanda	Política		Ordem	VP	n Δ
		s_1	S_1	Ótima		
Dados	Estocást.	05	31	31	3980	447
Originais	Determ.	05	30	30	4070	389
$r = 200$	Estocást.	07	32	32	8532	494
	Determ.	07	30	30	8643	389
$a = 0.99$	Estocást.	06	52	52	4299	610
	Determ.	07	50	50	4393	590
$b = 0.7$	Estocást.	08	32	32	3960	522
	Determ.	08	30	30	4070	389
$k = 50$	Estocást.	07	21	21	4172	225
	Determ.	07	20	20	4250	195
hn/r = 0.1	Estocást.	07	32	32	3964	476
	Determ.	07	30	30	4070	389

Capítulo 6

Contole e Estimação LQG

Neste ponto do trabalho, o leitor já deve ter indagado quanto à aplicabilidade, sob o ponto de vista computacional, das estruturas recursivas que construimos. É claro que a viabilidade de utilização do método depende fortemente da estrutura da função custo e também da dinâmica relacionando estados e controles.

Estudaremos agora sistemas onde o controle ótimo em cada instante é trivialmente construido a partir de uma transformação linear do estado no qual o sistema se encontra. Trabalharemos com sistemas nos quais os estados e os controles são determinísticos e relacionam-se com seu passado segundo uma expressão linear do tipo $x(t+1) = A(t) * x(t) + B(t) * u(t)$, com matrizes $A(t)$ e $B(t)$ conhecidas, e onde os custos são funções quadráticas dos estados.

As propriedades decorrentes desta estrutura são muito fortes e acarretam um ganho substancial do ponto de vista de eficiência computacional. Deve-se ainda salientar que um grande número de problemas reais admitem uma modelagem desta natureza, e assim estes modelos conjugam eficiência e aplicabilidade.

6.1 Evolução Linear com Custo Quadrático

Num sistema **Linear com custo Quadrático** (LQ) os estados evoluem na forma

$$x(t+1) = A(t) * x(t) + B(t) * u(t)$$

sendo o custo instantâneo uma função quadrática $c(t, x, u)$.

Mais precisamente, o custo quadrático instantâneo é definido em função de uma matriz $Q(t)$, simétrica e positiva definida, ou seja:

$$c(t, x(t), u(t)) = \begin{bmatrix} x(t)' & u'(t) \end{bmatrix} \begin{bmatrix} QX(t) & QC(t) \\ QC(t)' & QU(t) \end{bmatrix} \begin{bmatrix} x(t) \\ u(t) \end{bmatrix}$$

A função de custo adotada é simplesmente o custo aditivo e para cada instante t, definimos as funções auxiliares de custo futuro

$$f_{\Pi}(t,x) = \sum_{s=t}^{h} c(s,x,u)$$

que fornecem o custo a partir do instante t, quando adotamos a política Π.

Para este problema convencionamos que $u(h) = 0$, ou seja, o controle exercido no horizonte de planejamento é nulo.

O seguinte teorema apresenta um resultado muito importante e surpreendente que permite que para esta particular classe de sistemas, o controle ótimo em cada instante seja obtido trivialmente como uma transformação linear do estado em que o sistema se encontra.

Teorema (Princípio de Otimalidade)

Em um sistema linear com custo quadrático vale:

a. A política ótima é uma transformação linear do estado presente, i.e. existe uma matriz $K(t)$ denominada matriz de controle tal que

$$u^*(t) = K(t)x(t)$$

b. O custo futuro ótimo é uma função quadrática do estado presente, i.e. existe, para cada instante t, uma matriz $FO(t)$, tal que $f^*(t) = x(t)'FO(t)x(t)$.

Prova:

Base de indução (condição de contorno)

Como por hipótese $u(t) = 0$, segue trivialmente da definição do custo que

$$f^*(x(h)) = x(h)'QX(h)x(h).$$

Passo de Indução (equação de Bellman):

$$\begin{aligned}
f^*(x(t)) &= \min_{u(t)} c(x(t),u(t)) + x(t+1)'FO(t+1)x(t+1) \quad = \\
&\min_{u(t)} \begin{bmatrix} x(t)' & u'(t) \end{bmatrix} \begin{bmatrix} QX(t) + A(t)'FO(t+1)A(t) & QC(t) + A(t)'FO(t+1)B(t) \\ QC(t)' + B(t)'FO(t+1)A(t) & QU(t) + B(t)'FO(t+1)B(t) \end{bmatrix} \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} \\
&= \min_{u(t)} \begin{bmatrix} x(t)' & u'(t) \end{bmatrix} \begin{bmatrix} FX(t) & FC(t) \\ FC(t)' & FU(t) \end{bmatrix} \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} \\
&= \min_{u(t)} f(x(t),u(t))
\end{aligned}$$

A determinação do controle ótimo é obtida através de

$$\nabla_u(f(x,u)) = 2FU(t)u(t) + 2FC(t)'x(t) = 0$$

Mas

$$\nabla_u(f(x,u^*)) = 0 \Rightarrow u^*(t) = K(t)x(t)$$

onde $K(t) = -FU(t)^{-1}FC(t)'x(t)$ (Note que $FU(t)$ é inversível pois $FU(t) > 0$, o que segue da hipótese de positividade de $Q(t)$).

Assim temos:

$$\begin{aligned} K(t)x(t) &\equiv -FU(t)^{-1}FC(t)'x(t) \\ &= -\left[(QU(t) + B(t)'FO(t+1)B(t))^{-1}(QC(t)' + B(t)'FO(t+1)A(t))\right] \end{aligned}$$

Tomando $u(t) = K(t)x(t)$, temos

$$f^*(x(t)) = x(t)'FO(t)x(t) = x(t)'(FX(t) - FC(t)FU(t)^{-1}FC(t)')x(t)$$

Q.E.D.

Observe que tomando o controle ótimo $u(t) = u^*(t)$, conforme apresentado no teorema, a evolução do sistema será dada por

$$x(t+1) = A(t)x(t) + B(t)K(t)x(t) = (A(t) + B(t)K(t))x(t) \equiv G(t)x(t)$$

A matriz $G(t) = A(t) + B(t)K(t)$ é denominada *matriz de ganho do sistema*.

A prova do teorema não só garante a forma linear do controle ótimo e a forma quadrática do ótimo da função de custo futuro, mas fornece explicitamente um sistema de relações de recorrência para o cômputo das matrizes envolvidas.

Mais precisamente, tanto as matrizes que fornecem os controles e custo futuro ótimos ($FO(t)$ e $K(t)$) quanto as matrizes que são utilizadas para obtê-las ($FX(t)$, $FC(t)$, $FU(t)$, $FO(t)$) podem ser escritas em função da matriz $Q(t)$ que define o custo quadrático e da matriz $FO(t+1)$. Ou seja,

$$[FX(t), FC(t), FU(t), FO(t), K(t), G(t)] = R[QX(t), QC(t), QU(t), FO(t+1)].$$

Esta relação de recorrência regredindo no tempo é conhecida como a *Equação de Riccati*. Note que na equação de Riccati a matriz FO é o elo da recorrência, sendo a única entidade aparecendo com índice $t+1$ o que se reflete em um ganho computacional substancial.

6.2 Sistemas Homogêneos no Tempo

Analisemos o que ocorre quando trabalhamos com sistemas invariantes no tempo, ou seja, quando as matrizes de coeficientes são constantes. Chamamos de *Sistemas Homogêneos no Tempo* àqueles que evoluem segundo uma relação do tipo:

$$x(t+1) = Ax(t) + Bu(t).$$

Uma propriedade que seria desejável nestes sistemas é denominada controlabilidade. Um sistema é controlável se e só se podemos levá-lo a um estado previamente fixado a partir de qualquer estado inicial $x(1)$. Formalmente,

Controlabilidade

Um sistema linear $x(t+1) = Ax(t)+Bu(t)$ é *k-controlável* sse, para qualquer estado inicial $x(1)$, existe uma sequência de k controles, $u(1), u(2), \ldots, u(k)$, que levam $x(1)$ a um estado arbitrariamente prefixado, $x(k+1)$.

É fácil verificar que o sistema é k-controlável se e só se a equação linear $x(k+1) - A^k x(1) = \sum_{i=0}^{k-1} A^i B u(k-i)$ for sempre solúvel, ou equivalentemente, se a matriz $M(k) = \left[B|AB|A^2B|\ldots|A^{k-1}B\right]$ tiver posto pleno.

Vimos, ao estudar o Princípio de Otimalidade, que o custo futuro ótimo se escreve como uma função quadrática do estado presente e que sua determinação está diretamente associada à estrutura recursiva obtida para o cálculo das matrizes $FX(t), FC(t), FU(t), K(t)$ e $G(t)$. Uma importante questão torna-se então a existência de um ponto fixo na Equação de Ricatti, i.e. na expressão recursiva. Queremos assegurar, na verdade, que para sistemas com horizonte muito grande a função custo futuro convirja.

Observemos inicialmente que, fixado um horizonte h e um estado x, $f^*(t,x)$ decresce monotonicamente em t. De fato, sendo $x^*(t)$ e $u^*(t)$ a trajetória ótima correspondendo ao custo $f^*(t,x)$, temos que

$$\begin{aligned} f^*(t,x) &= \sum_{k=t}^{h} c(k, x^*(t), u^*(k)) \geq \sum_{k=t}^{h-1} c(k, x^*(t), u^*(k)) \\ &= \sum_{k=t+1}^{h} c(k, x^*(t), u^*(k)) \geq f^*(t+1, x) \end{aligned}$$

Reciprocamente, fixando x e t, $f_h^*(t,x)$ cresce monotonicamente ao aumentarmos o horizonte, h. Se todavia (A,B) for controlável, $f_h^*(t,x)$ é limitada superiormente, e portanto converge para um limite $f_\infty^*(t,x)$. Logo, em virtude do princípio de otimalidade, também FO converge. Como todas as outras matrizes na equação de Riccati são escritas em função de FO, temos limites de horizonte infinito para FX, FU, FC, K e G. No limite

de horizonte infinito FO será um ponto fixo da equação de Riccati. Esta é a *Equação de Equilíbrio de Riccati*:

$$FO = R[QX, QC, QU, FO].$$

O conceito *Estabilidade* indica se um sistema homogêneo converge para a origem.

> **Estabilidade** Um sistema linear $x(t+1) = Ax(t)+Bu(t)$ é dito *estável* se e somente se existir uma matriz de controle, K, tal que $x(t+1) = (A+BK)^t x(0) = G^t x(0) \to 0$

Mas como os autovalores de G^t são as t-ésimas potências dos autovalores de G, a última condição equivale a termos todos os autovalores de G dentro do círculo unitário. Nestas condições, e por abuso de notração a matriz de ganho G é dita estável.

No caso particular onde a matriz B é nula, é imediato que a seqüência $\{x(t)\}$ converge para a origem, a partir de um estado inicial arbitrário, $x(1)$, se e somente se $\lim_{t\to\infty} A^t = 0$. Mas como os autovalores de A^t são as t-ésimas potências dos autovalores de A, a última condição equivale a termos todos os autovalores de A dentro do círculo unitário. Neste caso $||x(t)|| \leq |\lambda|^t ||x(0)||$, onde λ é o autovalor de G de máximo módulo. Nestas condições o sistema, e por abuso de notação a matriz A, é dito **estável**. Estabilidade, assim como controlabilidade é uma condição que assegura limitação de $f_h^*(t, x)$ e consequentemente a convergência de f.

Para controlar sistemas com horizonte muito grande é prático utilizarmos o controle de horizonte infinito para $t < ha, (h - ha) \ll h$, o controle (sub-ótimo) "de cruzeiro", e o controle ótimo para "a aproximação final" em $ha \leq t \leq h$.

6.3 Evolução Linear com Ruído Gaussiano

Estudamos na seção anterior, modelos onde tanto estados quanto controles eram determinísticos. A introdução de ruídos aleatórios em um sistema trás os benefícios de modelos estocásticos, porém exige que empreguemos um ferramental mais sofisticado, especialmente quando se pretende realizar estimações.

Admitiremos que são conhecidas observações de variáveis relacionadas com os estados cujos valores queremos prever. Nestas observações estão presentes ruídos, e será nosso objetivo filtrá-los de forma a conseguir informações sobre os estados. Aqui também será considerado essencialmente o caso linear, onde tanto as observações $y(t)$ e os estados $x(t)$ num dado instante t, quanto os estados em instantes distintos ($x(t)$, $x(t-1)$) estão interligados por relações lineares.

A literatura comumente apresenta a resolução do problema aqui analisado em termos de uma equação recursiva onde as soluções, i.e., as estimativas dos estados em um dado instante são obtidas em função da estimativa do instante imediatamente anterior, porém

com a desvantagem de haver a necessidade de se inverter uma matriz a cada passo. A formulação aqui apresentada, ainda que possa ser colocada dentro deste contexto, possibilita que seja explorada a estrutura dos sistemas lineares nos quais a resolução do problema recai.

Um sistema Linear com Ruído Gaussiano (LG) evolui na forma

$$x(t+1) = A(t)x(t) + B(t)u(t) + v(t)$$

onde são dados o controle $u(t)$ e as matrizes $A(t)$ e $B(t)$, e $v(t)$ e um processo estocástico Gaussiano, i.e. de esperança zero e distribuição normal multi-variada.

No sistema LG a informação que temos a cada instante sobre o estado do sistema, $x(t)$, é a observação :

$$y(t) = C(t)x(t) + w(t)$$

onde a matriz $C(t)$ é dada, e $w(t)$ é um processo Gaussiano.

Para completar a caracterização do sistema LG, é dada a covariância dos ruído, i.e.

$$Cov(\begin{bmatrix} w(t) \\ -v(t) \end{bmatrix}) = \begin{bmatrix} VX(t) & VC(t) \\ VC(t)' & VY(t) \end{bmatrix} = L(t)L(t)'.$$

Os ruídos em tempos distintos são supostos independentes.

Nosso objetivo é encontrar a melhor estimativa para os estados passados, presente, e o estado do próximo instante futuro quando se dispõe de informações até o presente t. Assim, queremos estimar no instante t, os valores, $x(t-k|t)$ ou $x(t|t)$, ou $x(t+1|t)$. Estimar $x(t)$ no instante t, $x(t|t)$, é chamado **filtração** , estimar $x(t+1)$ no instante t, $x(t+1|t)$, é chamado **predição** , enquanto estimar os estados passados, $x(t-k|t)$, é chamado **revisão**.

Numa notação compactada, queremos estimar o vetor $x[t+1] = [x(1)'|\dots|x(t)'|x(t+1)']'$, a partir das equações de evolução e das observações presente e passadas, $y[t] = [y(1)'|\dots|y(t)']'$.

Sendo as matrizes $B(t)$ e os controles $u(t)$ conhecidos, estão disponíveis as diferenças no estado por ação do controle,

$$\begin{aligned} d(t) &= B(t)u(t) = x(t+1) - A(t)x(t) - v(t) \\ d(0) &= x(1) - v(0) \end{aligned}$$

onde $d(0)$ é uma estimativa do estado inicial, $x(1)$, sujeita a um erro Gaussiano $v(0)$ de covariância $L(0)L(0)'$.

Podemos escrever todas as equações relacionando os estados entre si e também as observações e estados ao longo do tempo na forma de um sistema de equações lineares cuja matriz é esparsa e estruturada:

$$\begin{bmatrix} d(0) \\ y(1) \\ d(1) \\ y(2) \\ d(2) \\ \bullet \\ \bullet \\ y(t) \\ d(t) \end{bmatrix} = \begin{bmatrix} I & & & & & & \\ C(1) & & & & & & \\ -A(1) & I & & & & & \\ & C(2) & & & & & \\ & -A(2) & \bullet & & & & \\ & & \bullet & \bullet & & & \\ & & & & C(t) & & \\ & & & & -A(t) & I \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \\ \bullet \\ \bullet \\ x(t) \\ x(t+1) \end{bmatrix} + \begin{bmatrix} -v(0) \\ w(1) \\ -v(1) \\ w(2) \\ -v(2) \\ \bullet \\ \bullet \\ w(t) \\ -v(t) \end{bmatrix}$$

ou, numa notação mais compacta,

$$\begin{cases} d[t] = A[t]x[t+1] + v[t] \\ y[t] = C[t]x[t] + w[t]. \end{cases}$$

Duas questões são decisivas ao tratarmos da resolução do problema: a obtenção de seus parâmetros, i.e., das matrizes de coeficientes $A(t)$, $B(t)$ e $C(t)$ que definem a dinâmica no sistema acima, e a determinação das estimativas para uma particular escolha dos parâmetros. Restringiremos nossa discussão ao caso em que os parâmetros são conhecidos pois a questão de estimação de parÂmetros foge do escopo deste trabalho.

Conhecidos os parâmetros do modelo, a obtenção de estimativas de estados é conseguida através da resolução do sistema acima, explorando-se esta particular estrutura.

Multiplicando a equação acima por

$$L[t]^{-1} = diag(L(0), L(1), \ldots, L(t))^{-1}$$

temos

$$\begin{bmatrix} \bar{d}(0) \\ \bar{y}(1) \\ \bar{d}(1) \\ \bar{y}(2) \\ \bar{d}(2) \\ \bullet \\ \bullet \\ \bar{y}(t) \\ \bar{d}(t) \end{bmatrix} = \begin{bmatrix} \bar{L}(0) & & & & & & \\ \bar{C}(1) & & & & & & \\ \bar{A}(1) & \bar{L}(1) & & & & & \\ & \bar{C}(2) & & & & & \\ & \bar{A}(2) & \bullet & & & & \\ & & \bullet & & & & \\ & & \bullet & \bullet & & & \\ & & & & \bar{C}(t) & & \\ & & & & \bar{A}(t) & \bar{L}(t) \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \\ \bullet \\ \bullet \\ x(t) \\ x(t+1) \end{bmatrix} + \begin{bmatrix} \bar{v}(0) \\ \bar{w}(1) \\ \bar{v}(1) \\ \bar{w}(2) \\ \bar{v}(2) \\ \bullet \\ \bullet \\ \bar{w}(t) \\ \bar{v}(t) \end{bmatrix}$$

ou, numa notação mais compacta,

$$\begin{cases} \bar{d}[t] = \bar{A}[t]x[t+1] + \bar{v}[t] \\ \bar{y}[t] = \bar{C}[t]x[t] + \bar{w}[t] \end{cases}$$

Ao colocar o sistema nesta forma obtemos um novo sistema, ainda bem estruturado, e com a vantagem de possuir ruídos brancos, i.e., com matriz de covariância igual à identidade. Isso decorre da propriedade da covariância: $Cov(Mx) = MCov(x)M'$, e pode ser verificado facilmente. Neste caso, sabemos que coincidem o estimador de máxima verossimilhança , MLE, o estimador de mínimos quadrados, LSE, e também o melhor estimador linear não tendencioso, BLUE. Portando a melhor estimativa de $x[t+1]$, LSE, é obtida minimizando a norma quatrática de resíduo (ruído) no sistema de equações super-determinado acima.

Vamos então resolver o sistema apresentado utilizando transformações ortogonais nos subsistemas do sistema original. Essas transformações deixam inalterada a norma quadrática de um vetor, e nos permitem colocar o sistema de equações numa forma ainda mais conveniente. Consideremos então as seguintes fatorações ortogonais (QR):

$$Q(1)' \begin{bmatrix} \bar{L}(0) & 0 & \bar{d}(0) \\ \bar{C}(1) & 0 & \bar{y}(1) \\ \bar{A}(1) & \bar{L}(1) & \bar{d}(1) \end{bmatrix} = \begin{bmatrix} \tilde{L}(0) & S(0) & \tilde{d}(0) \\ 0 & 0 & \tilde{y}(1) \\ 0 & \hat{L}(1) & \hat{d}(1) \end{bmatrix}$$

$$Q(t)' \begin{bmatrix} \hat{L}(t-1) & 0 & \hat{d}(t-1) \\ \bar{C}(t) & 0 & \bar{y}(t) \\ \bar{A}(t) & \bar{L}(t) & \bar{d}(t) \end{bmatrix} = \begin{bmatrix} \tilde{L}(t-1) & S(t-1) & \tilde{d}(t-1) \\ 0 & 0 & \tilde{y}(t) \\ 0 & \hat{L}(t) & \hat{d}(t) \end{bmatrix}$$

As transformações acima são facilmente construídas como sequências de rotações de Givens que eliminam as colunas de $\bar{C}$ e $\bar{A}$ da direita para a esquerda, eliminando os elementos de cima para baixo.

As transformações ortogonais do sistema de equações são definidas pelas composições das transformações ortogonais aplicadas aos blocos,

$$Q[t]' = Q(t)' \circ Q(t-1)' \circ \ldots \circ Q(1)' \circ Q(0)' \ .$$

Aplicando $Q[t]'$ ao nosso sistema de equações temos

$$\begin{bmatrix} \tilde{d}(0) \\ \tilde{y}(1) \\ \tilde{d}(1) \\ \bullet \\ \bullet \\ \tilde{d}(t-1) \\ \tilde{y}(t) \\ \hat{d}(t) \end{bmatrix} = \begin{bmatrix} \tilde{L}(0) & S(0) & & & & & \\ 0 & 0 & & & & & \\ & \tilde{L}(1) & \bullet & & & & \\ & & \bullet & & & & \\ & & \bullet & \bullet & & & \\ & & & 0 & & & \\ & & & \tilde{L}(t-1) & S(t-1) & & \\ & & & 0 & 0 & & \\ & & & & \hat{L}(t) & & \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \\ \bullet \\ \bullet \\ x(t) \\ x(t+1) \end{bmatrix} + \begin{bmatrix} \tilde{v}(0) \\ \tilde{w}(1) \\ \tilde{v}(1) \\ \tilde{w}(2) \\ \tilde{v}(2) \\ \bullet \\ \bullet \\ \tilde{w}(t) \\ \hat{v}(t) \end{bmatrix}$$

ou, numa notação mais compacta, separando os dois sub-sistemas,

$$\begin{cases} \tilde{d}[t] = \tilde{L}[t]x[t+1] + \tilde{v}[t] \\ \tilde{y}[t] = 0 + \tilde{w}[t] \end{cases}$$

O primeiro subsistema, $\tilde{d}[t] = \tilde{L}[t]x[t+1]$, é triangular e pode ser resolvido exatamente, i.e. com ruído $\tilde{v}[t] = 0$. Todavia no segundo sub-sistema nada pode ser feito para acomodar $\tilde{y}[t] = 0$. Portando o estimador de mínimos quadrados é simplesmente a solução do sistema triangular.

Para fazer apenas a filtração, isto é, estimar $x(t)$ no instante t, não é necessário manter toda a matriz $\tilde{L}[t]$. Para a operação de predição no instante t basta preservar $\hat{L}(t)$ e $\hat{d}(t)$, como fica claro na relação de recorrência:

$$\begin{aligned} x(t+1|t) &= \hat{L}(t)^{-1}\hat{d}(t) \\ x(t|t) &= \tilde{L}(t-1)^{-1}(\tilde{d}(t-1) - S(t-1)x(t+1)) \end{aligned}$$

6.4 Princípio de Equivalência

O filtro acima descrito é uma variante do filtro de Kalman. Estude a implementação deste filtro em Matlab, e use-o para estimar os estados presentes e passados, no sistema do exemplo dado, ao longo de sua evolução temporal. Interprete o filtro que construimos em termos de programação dinâmica. Interprete cada um dos conceitos apresentados no capítulo 4 no contexto do filtro.

Não é difícil demonstrar que a teoria de controle LQ é compatível com a teoria de sistemas LG no seguinte sentido: o controle u^{**} que minimiza o custo de controle esperado, em um sistema LQG (sistema linear com custo quadrático e ruído gaussiano), é igual ao controle u^* que minimiza o custo de controle no sistema LQ determinístico, onde o estado do sistema LQ é o estado do sistema LQG estimado pelo Filtro de Kalman. Este é o Princípio de Equivalência. Neste sentido o problema de controle LQ e o problema de estimação LG são ditos duais.

6.5 Generalizações do Filtro de Kalman

Diversas generalizações do Filtro de Kalman são comumente empregadas. Para os problemas que analisamos em finanças, estudamos sistemas na forma:

$$\begin{cases} x(t+1) = f_t(x(t)) + B(t)u(t) + \epsilon(t) \\ y(t) = g_t(x(t)) + \xi(t) \qquad\qquad t \in N \end{cases} \tag{6.1}$$

O objetivo aqui é encontrar uma estimativa, $\hat{x}(t^* \mid t)$, para algum t^*,

As funções f_t e g_t podem não ser lineares, porém são deriváveis. Também aqui os controles u_t são determinísticos e $y(t)$ são as observações realizadas. A abordagem mais simples consiste em fazer aproximações por Taylor recaindo no caso linear.

6.6 Exercícios

1. Reescreva a teoria de sistemas LQ com custo aditivo descontado.

2. Considere o vetor de estados aumentado, $[x_1, \ldots x_n, 1]'$; e explique como usar esta última "coordenada constante" do vetor de estado para introduzir custos puramente lineares em x ou u.

3. Implemente a equação de Ricatti com custo descontado em Matlab. Monte e controle um exemplo simples.

4. Prove que

   (a) Todo sistema controlável é estabilizável.

   (b) Todo sistema k-controlável é $k+1$-controlável.

   (c) Se $x \in R^n$, todo sistema controlável é $n+1$-controlável.

5. De exemplos de

   (a) Um sistema estabilizável mas não controlável.

   (b) Um sistema, em R^2, 2-controlável mas não 1-controlável.

6. Imagine que a medida que o tempo passa, passemos a descrer cada vez mais de nossas observações e estimativas passadas: Formalmente, estando no instante t tomaremos

$$Cov(\begin{bmatrix} w(t-k) \\ -v(t-k) \end{bmatrix}) = \delta^{t-k} \begin{bmatrix} VX(t-k) & VC(t-k) \\ VC(t-k)' & VY(t-k) \end{bmatrix} .$$

   Este modelo é chamado de *memória evanescente*. Reescreva a teoria de estimação LG com memória evanescente, e interprete o medelo como um problema de programação dinâmica com custo descontado.

7. Implemente a Estimação LG em linguagem C.

   a. Considere memória evanescente.

   b. Considere que os parâmetros de evolução do sistema, A, B e L, bem como o desconto intertemporal δ, são constantes. Considere que a matriz de observação, C é usualmente igual a um padrão, C, mas que podemos especificar, em cada instante, alterações do padrão.

   c. Explique como utilizar alterações em C para modelar observações perdidas, dias sem pregão, etc.

   d. A implementação deve permitir filtração, previsão para até kp passos no futuro (via observações nulas), e revisão para no máximo $kr << h$ instantes anteriores. Explique como esta revisão limitada facilita a implementação do algoritmo.

e. Implemente medidas de erro quadrático médio sobre os resíduos de filtração, k-revisão, e k-predição. Explique como poderíamos usar estas medidas de erro para melhor ajustar o modelo (A, B, C, L, δ).

Capítulo 7

Árvores de Decisão

Este capítulo apresenta o algoritmo REAL de aprendizado automático, para construção de árvores de classificação TDIDT (Top Down Induction Decision Tree) com atributos a valores reais [1], [7], [8].

O Projeto REAL começou como uma aplicação a ser utilizada no mercado de ações, provendo um bom algoritmo para pever a adequação de estratégias de operação. Neste contexto, o sucesso ou fracasso de uma dada operação corresponde a classe do exemplo, enquanto os atributos são "indicadores técnicos" que assumem valores reais. As exigencias dos usuários para a ferramenta de suporte à decisão explicam várias características únicas do algoritmo.

O projeto começou testando vários algoritmos de aprendizado apresentados no projeto ESPRIT-StatLog [5]; o software CAL5 [6], um algoritmo Top-Down para geração de árvores de classificação, mostrou-se especialmente adequado para a aplicação que tinhamos em mente. O algoritmo CAL5 teve uma forte influência no projeto, e foi utilizado como a principal referência para comparação de performance. Na nossa aplicação o algoritmo REAL apresentou algumas vantagens importantes:

1. Geralmente as árvores de classificação tem menor taxa de erro.

2. Uma medida única de convicção mostrou-se mais conveniente que o tradicional par (probabilidade, confiânça).

3. Os procedimentos de ramificação do REAL detem-se naturalmente, dispensando um posterior procedimento de poda.

7.1 Formulação do Problema

Os problemas de classificação são apresentados como uma matriz A, $n \times (m+1)$. Cada linha, $A(i,:)$, representa um examplo, e cada coluna, $A(:,j)$, um atributo. As primeiras m colunas são atributos a valores reais, e a última coluna, $A(i,m+1)$ é a classe do exemplo. Parte destes exemplos, o conjunto de treinamento, é usado pelo algoritmo para gerar a árvore de classificação, que é então testada com os exemplos remanecentes. A taxa de erro de classificação no conjunto de teste é uma maneira simples de avaliar a árvore de classificação.

7.2 Construção da Árvore

Cada iteração principal do algoritmo REAL corresponde à ramificação de um nó terminal (folha) da árvore. Os exemplos naquele nó são classificados de acordo com o valor do atributo selecionado, e novos ramos são gerados para intervalos especificos. A partição do domínio de um atributo em (sub) intervalos adjacentes e não sobrepostos corresponde ao processo de discretização. Cada iteração principal do REAL inclui:

1. A discretização de cada atributo, e sua avaliação por uma função de perda.

2. Seleção do melhor atributo, e ramificação de acordo com sua discretização.

3. Junção de intervalos adjacentes que não alcançaram um limiar mínimo de convicção.

7.3 Convicção e Função de Perda

Dado um nó de classe c con n exemplos, k dos quais incorretamente classificados e $n-k$ corretamente classificados, queremos um parâmetro escalar, cm, para medir tanto a probabilidade de termos uma classificação incorreta como o nível de confiança desta probabilidade. Uma tal medida simplificada de convicção nos foi colocada como uma necessidade dos usuários para operar no mercado de capitais. Seja q a probabilidade de classificação incorreta para um exemplo em um dado nó, $p = (1-q)$ a probabilidade de classificação correta, e assumamos a existência para q de uma distribuição Bayesiana:

$$D(c) = Pr(q \leq c) = Pr(p \geq 1 - c)$$

Definimos a medida de convicção: $100 * (1 - cm)\%$, onde

$$cm = \min c \;\mid\; Pr(q \leq c) \geq 1 - g(c)$$

e $g(\ )$ é uma bijeção monotonicamente crecente de $[0,1]$ sobre si mesmo. Nossa experiência no mercado de capitais nos ensinou a ser muito cautelosos com a certeza de afirmações, assim tomamos $g(\ )$ uma função convexa. Neste artigo $D(c)$ é a distribuição a posteriori de uma amostra tomada de uma distribuição de Bernoulli, com uma distribuição a priori uniforme para q:

$$\begin{aligned} B(n,k,q) &= comb(n,k) * q^k * p^{n-k} \\ D(c,n,k) &= \int_{q=0}^{c} B(n,k,q) \quad / \quad \int_{q=0}^{1} B(n,k,q) \\ &= \text{betainc}(c, k+1, n-k+1) \end{aligned}$$

Também neste artigo focalizamos nossa atenção em:

$$g(c) = g(c,r) = c^r, \quad r \geq 1.0$$

onde r é chamado o parâmetro de convexidade.

Com estas escolhas, a posteriori é uma função beta incompleta, facilmente computavel, e cm é a raiz da função monotonicamente decrecente:

$$\begin{aligned} cm(n,k,r) &= c \mid f(c) = 0 \\ f(c) &= 1 - g(c) - D(c,n,k) \\ &= 1 - c^r - \text{betainc}(c, k+1, n-k+1) \end{aligned}$$

Finalmente queremos uma função de perda para a discretização baseada na medida de convicção. Neste artigo utilizamos a somatória da convicção da classificação de cada exemplo, i.e. a soma, sobre todos os intervalos, da medida de convicção do intervalo vezes o número de exemplos nele contido.

$$loss = \sum_i n_i * cm_i$$

7.4 Procedimento de Discretização

Dado um atributo, o primeiro passo do processo de discretização é ordenar os exemplos no nó pelo valor do atributo, e então agrupar exemplos vizinhos de mesma clase, ou com identico valor do atributo. Assim, ao fim deste primeiro passo temos, para o atributo escolhido, a melhor discretização ordenada onde cada intervalo contem um grupo de exemplos de uma mesma classe, ou com o mesmo valor do atributo.

Nos passos subseqüentes, procuramos juntar intervalos para reduzir o valor da função de perda da discretização. O ganho de juntar J intervalos adjacentes, I_{h+1}, I_{h+2}, $\ldots I_{h+J}$, é o decrecimo relativo da função de perda:

$$gain(h,j) = \sum_j loss(n_j, k_j, r) \quad - loss(n,k,r)$$

onde $n = \sum_j n_j$ e k conta os exemplos de classes minoritárias no novo intervalo.

A cada passo realizamos a operação de agrupamento de máximo ganho. O procedimento de discretização termina ao esgorarem-se as operações de agrupamento com ganho positivo.

Os exemplos seguintes mostram alguns intervalos de classe uniforme que seriam agrupados que poderiam ser agrupados durante o procedimento de discretização. A notação $(n, k, m, r, \pm)$ significa que temos dois grupos de mesma classe, de tamanho n e m, separados por um grupo de tamanho k com exemplos de outra classe. r é o parâmetro de convexidade, e + (−) indica que poderiamos juntar (ou não) os três intervalos.

```
( 2,1, 2,2,+)
( 6,2, 7,2,-) ( 6,2, 8,2,+) ( 6,2,23,2,+) ( 6,2,24,2,-)
( 7,2, 6,2,-) ( 7,2, 7,2,+) ( 7,2,42,2,+) ( 7,2,43,2,-)
(23,3,23,2,-) (23,3,43,2,-) (23,3,44,2,+)
(11,3,13,3,-) (11,3,14,3,+) (11,3,39,3,+) (11,3,40,3,-)
(12,3,12,3,-) (12,3,13,3,+) (12,3,54,3,+) (12,3,55,3,-)
```

Nestes exemplos vemos que é necessário termos nos extremos grupos suficientemente grandes e equilibrados para "absorver" o ruido ou impureza no grupo intermediário. Um parâmerto de convexidade alto implica numa perda maior em grupos pequenos, ajudanto a absorção de impurezas esparsas.

7.5 Ramificação e Reagrupamento

Para cada nó terminal na árvore, devemos:

1. realizar, para cada atributo disponível, o procedimento de discretização.
2. medir a perda da discretização para cada atributo.
3. selecionar o atributo que leva a discretização de mínima perda, e
4. ramificar o nó de acordo com a discretzação correspondente.

Se nenhum atributo induzir uma discretização que decreça a função de perda de pelo menos um limite de precisão numérica $\epsilon > 0$, não ocorre ramificação.

Uma dada discretização, feita em um certo nível da árvore, pode impedir o progresso do processo de ramificação nos níveis abaixo. Por esta razão estabelecemos uma meta mínima de convicção, ct, e após cada ramificação reagupamos os intervalos adjacentes onde $cm < ct$. Para evitar um loop infinito, o valor da função de perda dado ao intervalo reagrupado

é a soma das perdas nos intervalos sendo reagrupados. Nas folhas da árvore final esta operação de reagrupamento é desfeita. A meta de convicção termina naturalmente o processo de ramificação, não havendo necessidade de um procedimento dicional para poda da árvore, como na maioria dos algoritmos TDIDT.

7.6 Implementação Computacional

Para os testes numéricos, a serem detalhados na seção 9, utilizamos uma implementação padrão do algoritmo REAL. Nesta implementação cada problema demora cerca de 2 minutos, incluindo treinamento e teste, em um Pentium 200MHz.

REAL foi implementado como um código *C++* absolutamente portavel, e a aplicação final final recebeu uma interface gráfica para o usuário em Microsoft VB-4.0. Esta implementação padrão gasta grande parte do tempo de processamento computando a função $cm(n, k, r)$. Podemos acelerar substancialmente o algoritmo utilizando tabelas pré-computadas para valore pequenos do argumento n (digamos $n \leq 100$), e tabelas de coeficientes para polinomios interpoladores para valores maiores de n. Outro expediente de aceleração é restringir o espaço de busca para as operações de agrupamento a uma vizinhança de apenas $2 \leq J \leq Jmax$ intervalos: Escolhendo um $Jmax$ conveniente aceleramos o algoritmo sem nenhuma degradação apreciavel dos resultados.

7.7 Estratégias de Operação no Mercado

Uma estratégia de operação no mercado é um conjunto pré-definido de regras determinando as ações de operador no mercado. A estratégia deve conter critérios para classificar uma operação realizada como um suscesso ou um fracasso.

Definamos, como exemplo, a estratégia $buysell(t, d, l, u, c)$:

- No instante t compre o ativo A, a seu preço $p(t)$.
- Venda o ativo A assim que:
  1. $t' = t + d$, ou
  2. $p(t') = p(t) * (1 + u/100)$, ou
  3. $p(t') = p(t) * (1 - l/100)$.
- A estratégia é bem suscedida se $c \leq 100 * p(t')/(p(t) \leq u$

Os parametros u, l, c and d podem ser interpretados como, respectivamente, o retorno desejado e o mínimo aceitavel o custo da operação, e o limite de tempo para

encerrar a operação. A figura 1 ilustra possíveis instâncias de aplicação da estratégia $buysell(t, d, l, u, c)$.

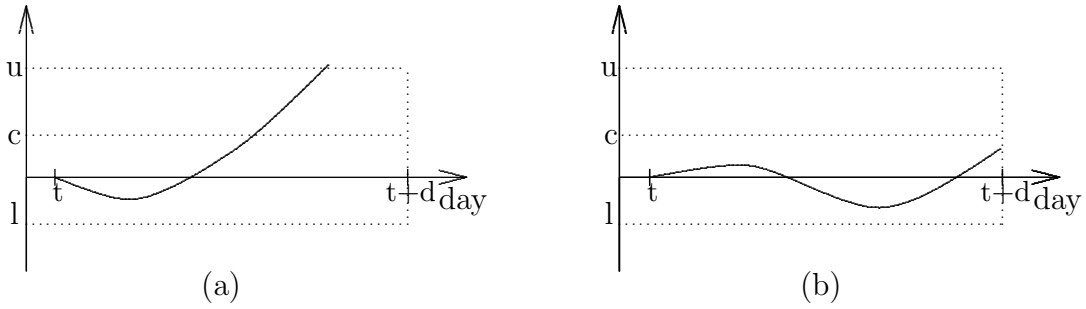


Figura 7.1: Exemplos para a estrategia $buysell(t, d, l, u, c)$.

Indicadores Técnicos

Analistas de Mercado utilizam várias ferramentas para previsão de preço de ativos, incluindo ferramentas baseadas em dados atuariais, análise de insumo-produto ou macroeconônica, etc., todas estas denominadas ferramentas fundamentalistas. Outra família de ferramentas são os indicadores técnicos. Um indicador técnico é uma função de uma ou mais variáveis observáveis no mercado. Vários distribuidores comercias difundem dados diarios e "intraday" para todos os mercados importantes. Na BOVESPA, a Bolsa de Valores de São Paulo, estes dados incluem: $H(t)$, $L(t)$, $O(t)$, $C(t)$, $M(t)$ and $V(t)$, respectivamente os preços máximo, mínimo, de abertura, fechamento e médio, e o volume negociado no dia t. Um exemplo de indicador técnico é o preço de abertura em relação ao máximo alcançado nos r dias anteriores:

$$OH_r(t) = O(t)/\max\{H(t), H(t-1), \ldots H(t-r)\}$$

Indicadores técnicos são ferramentas de análise de mercado tradicionais e de grande aceitação [3]. A grande familiaridade dos operadores de mercado estes indicadores explica a motivação conforto e confiança dos operadores ao utilizar uma ferramenta de suporte a decisão que fornecem regras lógicas de operação baseadas no valor de indicadores técnicos conhecidos (atributos). As regras e os atributos utilizados para uma dada classificação podem ser facilmente compreendidas (e aceitas ou rejeitadas). Esta confiança traduz-se em um uso ágil e eficiente da ferramenta. De fato, o uso dos indicadores como atributos da árvore TDIDT foi um pedido do cliente.

O núcleo do sistema de suporte à decisão para operações de mercado é um sistema de classificação do tipo TDIDT baseado no algoritmo REAL. Consideremos alguns problemas de classificação para a estratégia $buysell(t, d, l, u, c)$. Nestes problemas, um exemplo no instante t tem como atributos valores de vários indicadores técnicos no instante $t-1$, e

como classe o sucesso ou fracasso da aplicação da estratégia no instante t, baseado na informação disponível entre os instantes t e $d + d$. Tomamos os exemplos em segmentos de séries temporais sem superposição.

Função Objetivo

O procedimento de classificação, aplicado a um dado conjunto de exemplos, gera a matriz de classificação, também conhecida como matriz de confusão, como a matriz na tabela 1, onde $n11$ e $n22$ são o número de exemplos corretamente classificados de aplicações bem

Verdad./Atribuida	Sucesso	Falha
Sucesso	$n11$	$n12$
Falha	$n21$	$n22$

Tabela 7.1: Matriz de Classificação (confusão)

sucedidas e fracassadas, $n12$ são aplicações bem sucedidas incorretamente classificadas como falhas, e $n21$ os erros de classificação opostos. O operador de mercado espera que o sistema de suporte à decisão o auxilie a detectar quase todas as boas oportunidades de aplicação da estratégia. Quando aconselhado a aplicar a estatégia, o operador espera que o sistema raramente que o sistema raramente esteja errado. Assim nosso objetivo é o de maximizar a taxa de aproveitamento das oportunidades, ry, e também o de minimizar as falhas de aplicação, rf:

$$ry = n11/(n11 + n12) \quad \text{and} \quad rf = n21/(n11 + n21)$$

Para conciliar estes objetivos múltiplos e antagônicos definimos uma função de mérito que pode ser interpretada como uma estimativa conservadora do ganho com as aplicações da estratégia $buysell(t, d, l, u, c)$:

$$merit = c * n11 - l * n21$$

7.8 Testes Numéricos

Testamos o algoritmo de classificação exposto nas seções anteriores com $buysell(t, d, l, u, c)$, e $d = 5$ $dias$, $l = 1\%$, $c = 1\%$ e $u = 3\%$, em segmentos não superpostos de séries temporais de preços de duas das ações mais liqüidas (negociadas) na BOVESPA: Telebras-PN (TEL4), com aprox. 300 exemples 45% bem sicedidos, e Petrobras-PN (PET4), com aprox. 400 exemplos 40% bem sucedidos. Dividimos cada conjunto de exemplos em 10 subconjuntos. Em cada rodada do algoritmo geramos a árvore usando 9 dos subconjuntos para tereinamento, e testando a árvore no subconjunto remanecente. repetimos este procedimento para os parâmetros do algoritmo em um grid discreto:

REAL: $(r, ct) \in \{1.0, 1.5, \ldots, 4.0\} \times \{0.1, 0.15, 0.2, \ldots, 0.45\}$.

Cal5: $(S, \alpha) \in \{0.05, 0.10, \ldots, 0.90\} \times \{0.05, 0.10, \ldots, 0.90\}$.

NewID: $\phi \in \{0\%, 2\%, 4\%, \ldots, 28\%, 30\%\}$.

Também incluimos como benchmark NewID, uma algoritmo TDITT clássico, primordialmente desenvolvido para classificação categórica, mas também capaz de utilizar atributos reais. Nas tabelas 2 e 3 mostramos o valor dos parâmetros que otimizam a média da função de mérito, e os correspondentes valores médios de rf e ry.

Algoritmo	P^*	$merit$	rf	ry
REAL	$r = 3.5, ct = 0.4$	3.8	0.22	0.44
Cal5	$S = 0.3, \alpha = 0.1$	3.3	0.35	0.63
NewID	$\phi = 6\%$	2.9	0.25	0.45

Tabela 7.2: Parâmetros ótimos para TEL4

Algoritmo	P^*	$merit$	rf	ry
REAL	$r = 2, ct = 0.30$	4.3	0.24	0.39
Cal5	$S = 0.4, \alpha = 0.2$	3.8	0.23	0.35
NewID	$\phi = 8\%$	2.9	0.42	0.28

Tabela 7.3: Parâmetros ótimos para PET4

Para uma análise de sensibilidade mostramos resultados similares em uma vizinhança dos parâmetros ótimos nas tabelas 4 e 5.

r \ ct	0.35	$ct^* = 0.40$	0.45
3.0	$merit = 3.5$ $rf = 0.18$ $ry = 0.38$	$merit = 3.5$ $rf = 0.26$ $ry = 0.47$	$merit = 3.7$ $rf = 0.30$ $ry = 0.51$
$r^* = 3.5$	$merit = 3.5$ $rf = 0.14$ $ry = 0.34$	$merit = 3.8$ $rf = 0.22$ $ry = 0.44$	$merit = 3.5$ $rf = 0.31$ $ry = 0.51$
4.0	$merit = 3.2$ $rf = 0.27$ $ry = 0.38$	$merit = 3.5$ $rf = 0.24$ $ry = 0.41$	$merit = 3.2$ $rf = 0.27$ $ry = 0.45$

Tabela 7.4: REAL - Análise de sensibilidade para TEL4

r \ ct	0.25	$ct^* = 0.30$	0.35
1.5	$merit = 2.7$ $rf = 0.30$ $ry = 0.24$	$merit = 2.2$ $rf = 0.36$ $ry = 0.35$	$merit = 1.8$ $rf = 0.40$ $ry = 0.43$
$r^* = 2.0$	$merit = 3.0$ $rf = 0.35$ $ry = 0.23$	$merit = 4.3$ $rf = 0.24$ $ry = 0.39$	$merit = 3.1$ $rf = 0.36$ $ry = 0.45$
2.5	$merit = 0.1$ $rf = 0.94$ $ry = 0.02$	$merit = 3.2$ $rf = 0.36$ $ry = 0.26$	$merit = 3.4$ $rf = 0.26$ $ry = 0.32$

Tabela 7.5: REAL - Análise de sensibilidade para PET4

Mais Testes Numéricos

Também testamos os algoritmos REAL e CAL5 com o conjunto de exemplos de van Cutsem para detecção de emergências em redes de potência; para o qual CAL5 teve a melhor performance já publicada [4]. Otimizamos os parâmetros sobre o mesmo grid da seção anterior.

As árvores geradas foram testadas de duas maneiras:

1. A contagem padrão *hits* e *misses*, i.e. classificações corretas e incorretas.

2. Eliminando os exemplos que caem em folhas que não atingem a meta mínima de convicção (para o REAL), ou probabilidade-confiança (para o CAL5), e também eliminando os exemplos de teste que exibem um valor de atributos fora do intervalo do nó em que é utilizado na árvore de classificação, e só então contando os erros e acertos remanecentes, *Hits* e *Misses*.

Algoritmo	P^*	hit	$miss$	Hit	$Miss$
REAL	$(r, ct) = (1.5, 0.20)$	241.7	8.3	230.6	4.1
Cal5	$(S, \alpha) = (0.65, 0.15)$	240.6	9.4	236.2	6.9

Tabela 7.6: Parâmetros ótimos para van Cutsem

Para uma análise de sensibilidade mostramos resultados similares numa visinhança dos parâmetros ótimos, nas tabelas 7 e 8.

r \ ct	0.15	$ct^* = 0.20$	0.25
1.0	$hits = 240.5$	$hits = 239.0$	$hits = 236.4$
$r^* = 1.5$	$hits = 240.7$	$hits = 241.7$	$hits = 240.3$
2.0	$hits = 240.1$	$hits = 239.7$	$hits = 239.8$

Tabela 7.7: REAL - Análise de sensibilidade para van Cutsem

S \ α	0.10	$\alpha^* = 0.15$	0.20
0.6	$hits = 237.8$	$hits = 238.3$	$hits = 239.2$
$S^* = 0.65$	$hits = 238.5$	$hits = 240.6$	$hits = 239.8$
0.7	$hits = 239.9$	$hits = 238.8$	$hits = 239.6$

Tabela 7.8: Cal5 - Análise de sensibilidade para van Cutsem

7.9 Conclusões e Futuras Pesquisas

Os usuários queriam uma ferramenta de classificação que fornecesse regras de classificação "compreensiveis" e baseadas em atributos já familiares. REAL alcançou estes objetivos. A medida simplificada de convicção para cada classificação foi muito apreciada pelos usuários quando tomando decisões de mercado em tempo real. REAL mostrou-se mais eficiente que todos os outros algoritmos TDITD a que tivemos acesso, e mostrou-se uma ferramenta util para suporte à tomada de decisões quanto a aplicação de estratégias de operação no mercado de capitais.

Resultados preliminares nos levam a crer que as vantagems do REAL sobre o CAL5 se acentuam para dados mais ruidosos, mas esta afirmação requer mais testes numéricos. No momento estamos estudando o comportamento do REAL com variantes do procedimento de discretização baseadas em funções de custo alternativas, e também estamos interessados em compara-lo com abordagens diferentes para o problema [2].

Agradecimentos

Agradecemos o suporte recebido do DCC-IME-USP - Departamento de Ciência da Computação da Universidade of São Paulo, do CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico, da FAPESP - Fundação de Amparo à Pesquisa do Estado de São Paulo, e da *BM&F* Bolsa de Mercadorias e de Futuros do Estado de São Paulo. O software usado para computar os indicadores técnicos foi desenvolvido pela Profa. Celma O. Ribeiro, do Departamento de Engenharia Industrial da Escola Politécnica da USP.

Somos gratos a Junior Barrera, Alan M. Durham, Flávio S. C. da Silva, Jacob Zimbarg

Sobrinho and Carlos A. B. Pereira, do IME-USP, por muitos bons comentários, a Wolfgang Mueller, do Fraunhofer Institut, por um executável para SUN-Sparc do CAL5, e a Gerd Kock, do GMD-FIRST Berlin, por toda a ajuda na Alemanha.

Capítulo 8

Fundos de Pensão

A primeira parte deste capítulo descreve a utilização de uma ferramenta para a análise de fluxos de caixa em fundos de pensão no Brasil. Muitos dos fundos de pensão existentes são do tipo Benefício Definido (BD), onde o participante aposentado ou seus dependentes remanescentes recebem uma renda mensal vitalícia. O processo estocástico subjacente é modelado como um processo de ramificação orientado pelas diversas taxas de falha dependentes do tempo. Os fluxos de caixa esperados são computados através de funções recursivas que descrevem o processo de ramificação, evitando assim diversas aproximações que são utilizadas pelos métodos atuariais tradicionais. Essas funções recursivas também fornecem um cálculo direto da variância do fluxo de caixa e outras estatísticas.

As análises e simulações do passivo atuarial são usadas como entradas para a gestão de ativos do plano. Diversos modelos de otimização, geralmente empregando programação dinâmica e estocástica, são utilizados com este objetivo. Estas técnicas são discutidas na segunda parte do capítulo.

8.1 Passivo Atuarial

O principal benefício para o participante de um plano BD (benefício definido) é uma renda vitalícia durante a aposentadoria. Antes da aposentadoria, um membro é chamado ativo. A renda na aposentadoria é uma função dos salários ou contribuições passadas do participante enquanto ativo (p.ex. média dos últimos períodos). O participante ativo efetua contribuições ao plano de pensão, e essas contribuições podem ser complementadas via contribuições de uma patrocinadora (p.ex. o empregador ou o governo). Um participante ativo se tornará inativo quando se aposentar; a aposentadoria poderá ser ordinária (quando o participante cumpre as carências de idade e tempo de contribuição) ou por invalidez (p.ex. acidente ou doença). Um membro ativo também pode desligar-se do plano.

O participante também pode possuir dependentes (usualmente sua família) com direito a uma pensão mensal após a morte do participante. Dependentes podem ser vitalícios, que receberão uma pensão vitalícia (p.ex. esposa/viúva, filhos portadores de deficiência), ou temporários, que receberão a pensão por um tempo limitado (p.ex. filhos normais até a maioridade aos 21 anos). A pensão de cada dependente é uma fração da renda do participante. Um benefício adicional pago à família no momento do falecimento do participante - denominado pecúlio - pode também estar disponível.

Diversas restrições e correções [Alb 93], [Bor 92], [Bow 97], [Day 94], [DeF 91], [Pan 92] aumentam a complexidade deste modelo básico, como por exemplo:

- A aposentadoria e os demais benefícios por ela definidos podem ser corrigidos por um índice de inflação de longo prazo, ou ainda ser reajustados pelo salário de um participante ativo de mesmo nível funcional do aposentado.

- As carências para aposentadoria ordinária podem ser baseadas na idade e tempo de serviço do participante, e também baseadas nas condições determinadas pelo estatuto do plano ou pela legislação governamental vigente, ambas mutáveis ao longo do tempo.

- Os participantes podem receber uma aposentadoria básica do Estado (p.ex. INSS), sendo obrigação do plano complementar esta aposentadoria, até atingir os Benefício Definido (BD) pelo estatuto.

- Mudanças de hábitos sociais e de legislação podem alterar o status dos dependentes legais (p.ex. concubinas e filhos gerados fora do matrimônio).

- Participantes desligados podem reivindicar o resgate de suas contribuições (ou também as da patrocinadora) corrigidas pelos índices de inflação ou de investimentos financeiros.

8.2 Grafos e Formulação Recursiva

Um processo de ramificação é descrito por um grafo, onde cada vértice (ou nó) corresponde a um estado, e cada arco (ou aresta) conectando dois vértices corresponde a uma possível transição de estados. Nos processos atuariais que estamos estudando, um estado é caracterizado pela idade do participante, tempo de serviço, salário, família, etc. Uma transição é caracterizada por sua probabilidade, bem como pelos benefícios e contribuições que a transição implica. Em geral, é conveniente tratar os valores dos benefícios e contribuições como frações do benefício principal (aposentadoria), ou alguma outra unidade adimensional.

O valor esperado de uma variável aleatória (p.ex. benefícios ou contribuições) para um certo participante, num dado período, é a soma ponderada dos valores das variáveis

aleatórias em todas as transições possíveis naquele período:

$$E(X(t)) = \sum_{j \in W(t)} Pr(j) * x(j),$$

onde W é o conjunto de todas as transições possíveis, $x(j)$ o valor da variável aleatória na transição j, e $Pr(j)$ a probabilidade da transição j. O fluxo esperado daquela variável aleatória é a série dos seus valores esperados no futuro (períodos subsequentes, usualmente anos). A descrição dos processos de ramificação na forma de grafos fornece uma formulação algorítmica recursiva para o cálculo de todos esses fluxos de caixa.

Grafo do Participante Aposentado

O estado de um participante aposentado possui sua idade, benefícios e a lista de dependentes. Suponhamos que um participante aposentado possui no máximo um dependente vitalício (esposa). Se o participante e sua esposa estão ambos vivos no instante t, o participante estará, no instante $t+1$, em um dos quatro estados possíveis, dependendo da sobrevivência ou não do mesmo e de sua esposa: sejam (x, y) as idades do participante e de sua esposa no instante t. No instante $t+1$, eles podem alcançar os estados $(x+1, y+1)$, $(x+1, \sim)$, $(\sim, y+1)$ ou $(\sim, \sim)$, onde o til $(\sim)$ representa óbito. A probabilidade de cada uma das quatro transições é dada pela taxa de mortalidade, $h(a)$, nas respectivas idades:

$$Pr(t, (x, y), (x+1, y+1)) = (1 - h(x)) * (1 - h(y)); \tag{8.1}$$

$$Pr(t, (x, y), (x+1, \ )) = (1 - h(x)) * h(y); \tag{8.2}$$

$$Pr(t, (x, y), (\ , y+1)) = h(x) * (1 - h(y)); \tag{8.3}$$

$$Pr(t, (x, y), (\ , \ )) = h(x) * h(y). \tag{8.4}$$

Um participante aposentado deixa o sistema (plano previdenciário) quando cessam todos os fluxos de caixa por ele gerados, possivelmente muito tempo depois de seu próprio falecimento. As folhas da árvore ramificada do participante correspondem ao estado terminal $(\sim, \sim)$. Assume-se que os dependentes temporários (filhos) sempre sobrevivem até sua maioridade.

Como mencionamos na seção 2, podem ocorrer múltiplos dependentes vitalícios. Uma possibilidade seria incorporar os dependentes vitalícios múltiplos diretamente no processo de ramificação, a um alto custo computacional. Ocorre que os estatutos dos planos BD levam em consideração somente o número total de dependentes remanescentes após o falecimento do participante. Isto permite uma simplificação significativa. Modelamos os dependentes vitalícios no processo de ramificação na aposentadoria como um dependente vitalício virtual correspondente ao último dependente vitalício real remanescente. No apêndice E apresentamos um pequeno programa Matlab (denominado *depvital.m*) para

calcular a distribuição conjunta de probabilidades de sobrevivncia de tal dependente virtual. é fácil generalizar o procedimento para três ou mais dependentes vitalícios. Os fluxos de caixa dos dependentes vitalícios que falecem antes do último sobrevivente podem então ser modelados como fluxos de caixa independentes.

A modelagem precisa dos dependentes vitalícios múltiplos possui um impacto significativo sobre os fluxos de caixa de benefícios daqueles dependentes (tipicamente, 30%). Uma vez que essa situação está se tornando cada vez mais frequente, tal análise cuidadosa é recomendável. A figura 8.1 apresenta distribuições comparativas de sobrevivência calculadas pelo programa do apêndice E (as linhas pontilhadas representam as taxas de mortalidade do dependente mais velho e do mais jovem, respectivamente; as linhas sólidas representam as taxas de mortalidade do primeiro e do último dependente a falecer, respectivamente). Em alguns casos a estatística de ordem é aproximada pelas taxas de sobrevivência do dependente vitalício mais jovem. A partir dos gráficos 8.2(a) a **??**(d), pode-se perceber que esta aproximação pode ser bastante enganosa.

Grafo do Participante Ativo

O estado de um participante ativo possui sua idade, tempo de plano, tempo de serviço, escolaridade, salário, etc. Enquanto ativo, é difícil obter uma relação confiável de dependentes; assim, supõe-se que os participantes ativos possuem uma família-padrão, baseada em dados estatísticos e no perfil geral dos participantes. Se um participante está ativo no instante t, com idade a e tempo de serviço e, ele alcançará no instante $t+1$ um dos quatro possíveis estados, dependendo do mesmo ainda estar no plano, ativo, vivo, e válido. Falecimento, invalidez e desligamento são riscos competitivos com funções de probabilidade (condicionais na não ocorrência dos demais riscos precedentes) $hd(a)$, $hb(a)$ e $hw(e)$. Assim, as probabilidades de transição (exceto para a aposentadoria previsível, obtida na maturidade) para falecimento, invalidez, desligamento e permanência em atividade são, respectivamente:

$$hd(a), hb(a), hw(e), and \tag{8.5}$$

$$(1 - hd(a)) * (1 - hb(a)) * (1 - hw(e)). \tag{8.6}$$

Se o participante se desliga do plano, recebe um montante baseado em suas contribuições passadas. Se este morre ou se torna inválido, ele entra em regime de aposentadoria/pensão prematuramente. O processo de ramificação do membro ativo é portanto limitado ao ramo principal correspondente à sobrevivência sobre todos os riscos, uma estrutura mais parecida com um "bambu" do que com uma "árvore". As folhas do bambu correspondem ao estado de desligamento, ou à raiz de um processo de ramificação para aposentadoria.

8.3 Tábuas Biométricas e Outros Ajustes

Tábuas biométricas: Tábuas de taxas de mortalidade são disponíveis em diversos países. Uma das tábuas biométricas mais utilizadas no Brasil é a EB-7. Todavia, uma população específica, como a massa de participantes de uma certa companhia ou plano previdenciário, pode divergir significativamente das médias nacionais. Para planos específicos, alguns com até duzentos mil participantes, foi identificada a necessidade de ajustar essas tabelas. A figura 8.2 apresenta algumas comparações dessas distribuições de sobrevivência (as linhas pontilhadas representam as frequências observadas na população; as linhas sólidas representam as probabilidades ajustadas pelo modelo; as linhas tracejadas representam as taxas fornecidas pela tábua biométrica EB7-1975). Como é usual em atuária, estabelecemos um corte, limitando a idade individual a um máximo (p.ex. 100 anos). O impacto desses ajustes sobre o passivo do plano é considerável, da ordem de até 20%.

Utilizamos um modelo GMDH (Group Method Data Handling) polinomial, usando as tábuas disponíveis (informação a priori) e o histórico populacional (falecimentos observados e censurados) [Far 84]. Os modelos GMDH polinomiais possuem complexidade variável e diversos parâmetros. O melhor modelo foi selecionado automaticamente por uma heurística de busca controlada pelo critério PSE (Predicted Squared Error) [Bar 84]. O objetivo do critério PSE é minimizar erros sobre dados ainda não observados, buscando o equilíbrio entre erros sobre dados de treinamento e uma penalidade de overfit. O modelo final foi validado usando métodos computacionalmente intensivos de reamostragem estatística [Goo 99] [Urb 94].

Correção de unidade: Durante a modelagem de uma transição entre períodos consecutivos, de t a $t+1$, (dependendo de como o modelo é implementado), premissas não-realistas podem ser introduzidas, por exemplo: uma transição de falecimento pode implicar que o participante falece no primeiro (ou último) mês do ano. Para corrigir tal dicotomia booleana (0-1), podemos supor que o falecimento ocorre no mês central, e utilizar um fator de correção $6/12 = 1/2$, ou que o falecimento ocorre no dia central do mês central, e utilizar um fator de correção $(6 + 1/2)/12 = 13/24$, e assim por diante. Esses fatores de correção são denominados correções fracionais (ou correções de discretização) [Bow 97]. Seu impacto sobre os cálculos finais é normalmente pequeno, mas eles são importantes para preservar a consistência do modelo.

Crescimento Salarial: A renda (ou salário) de um participante ativo, que é a base para seus benefícos, é supostamente crescente ao longo de sua vida profissional. A renda usualmente aumenta com o tempo, mas tal crescimento possui um efeito de satuarção. Diversos modelos ajustam-se bem a esta situação [Pas 90], como os modelos Exponencial

Modificado, de Gompertz e Logístico (Pearl):

$$M(t) = a + b * exp(-c * t); \tag{8.7}$$
$$G(t) = exp(a + b * exp(-c * t)); \tag{8.8}$$
$$L(t) = a/(1 + b * exp(-c * t)). \tag{8.9}$$

8.4 Programação Estocástica

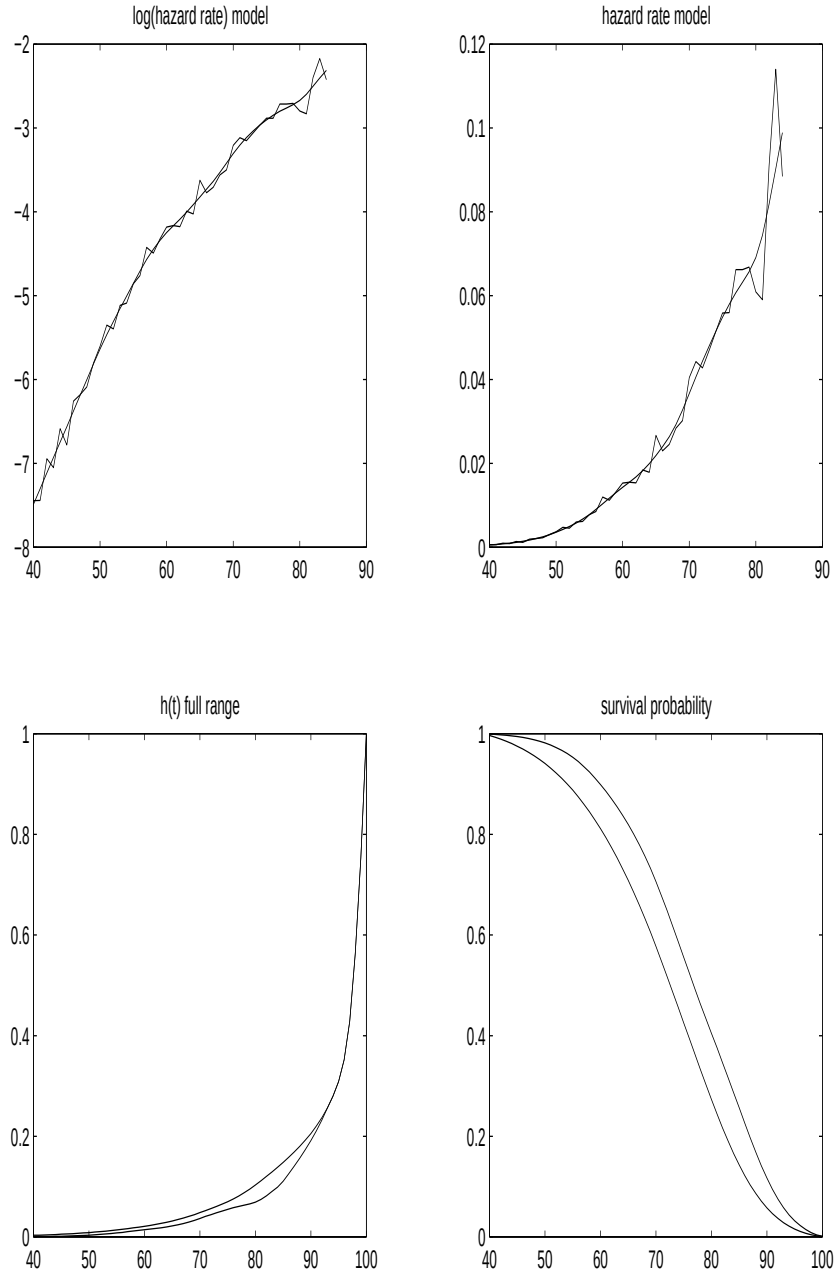


Figura 8.1: Estatística de ordem para modelagem de dependentes vitalícios múltiplos.

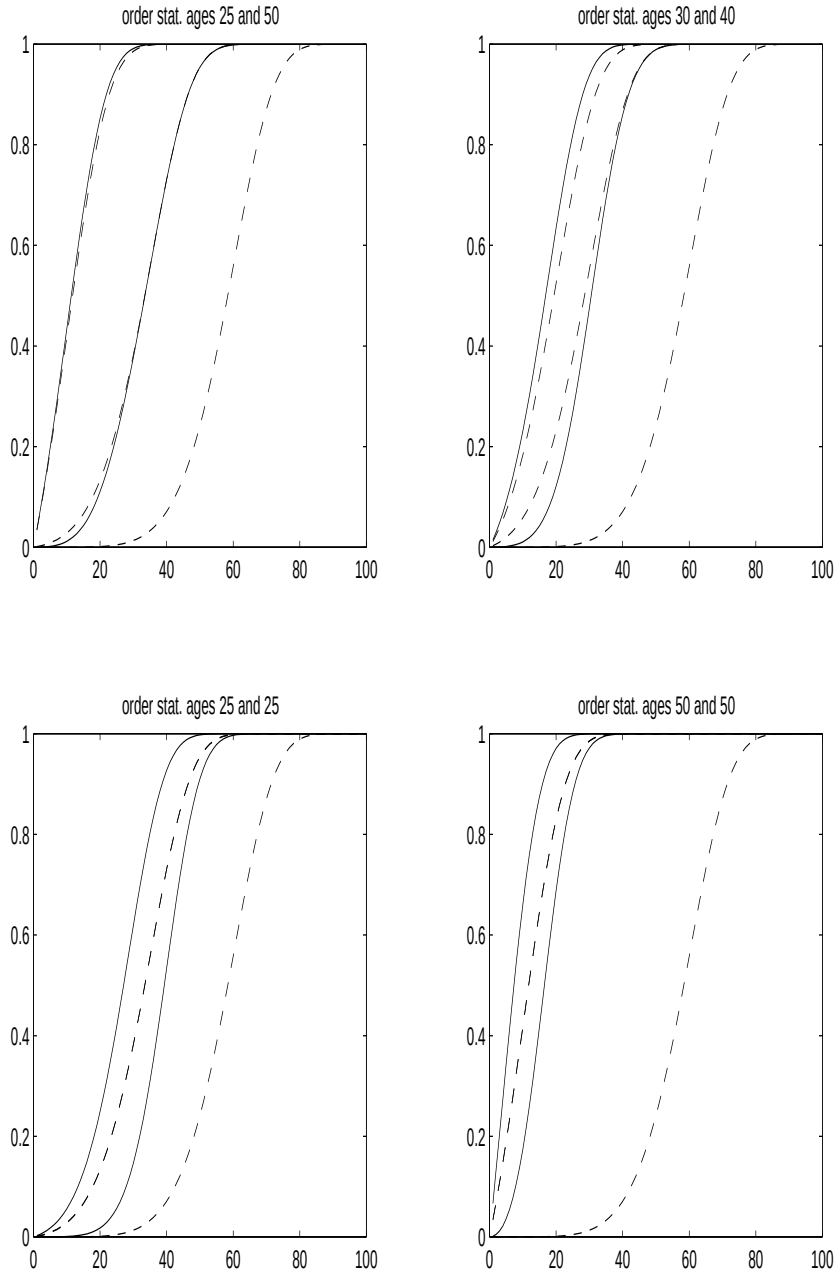


Figura 8.2: Comparações entre tábuas oficiais e tábuas obtidas a partir de uma população.

Capítulo 9

Portfólios Mistos Contendo Opções

Vamos construir neste capítulo portfólios formados por ativos fundamentais e opções européias sobre estes ativos. Para selecionar portfólios eficientes analisando-se média e variância de retornos, usaremos o modelo de Markowitz do capitulo anterior. Deduziremos expressões analiticas para a esperança e covariância entre taxas de retornos de ativos e opções, dados necessários para a construção do modelo baseado nos dois primeiros momentos centrais da carteira.

9.1 O mercado de opções

Opções de compra e de venda são contratos referentes a direitos e deveres de compra e venda adquiridos sobre um determinado ativo, o objeto do contrato.

Nas opções de compra (ou "call") o titular adquire o direito de comprar um determinado ativo numa data futura, a um determinado preço estipulado no contrato. E para adquirir tal direito paga-se um prêmio ao vendedor da opção. Nas opções de venda (ou " put") o titular tem o direito de vender o ativo-objeto a um preço pré-determinado em uma data futura.

Assim as partes integrantes de um contrato de opção englobam:

- Titular: aquele que detém o direito de comprar ou vender o ativo do contrato;
- Emissor (ou lançador): aquele que oferece o direito de compra ou de venda estipulado na opção;
- ativo-objeto: ativo que poderá ser negociado no contrato;
- preço de exercício: preço pelo qual o ativo-objeto poderá ser comprado ou vendido pelo titular;

- prêmio: valor pago ao lançador pelo titular para adquirir o direito de compra ou venda do ativo-objeto.

- data de exercício: data na qual o titular poderá negociar o ativo-objeto do contrato.

A data de exercício em que se pode negociar o ativo dá margem a dois tipos de contratos de opções: a opção americana e a européia. Na opção européia o titular pode exercer o seu direito somente na data estipulada no contrato. Já na opção americana o titular tem a liberdade de exercer seu direito a qualquer tempo até a data de exercício do contrato.

O prêmio pago ao emissor do contrato, que é o preço da opção, leva em consideração a oferta e a damanda do ativo, o preço de exercício, a cotação de mercado à vista do ativo, a volatilidade de preços, entre outros. Os ativos passíveis de negociação em opções são definidos pela instituição que as negocia, assim como o mês de expiração do contrato e os preços de exercício.

Em particular, para as opções sobre ações, a Bolsa de Valores de São Paulo, por exemplo, fixa a data de vencimento do contrato na segunda-feira mais próxima do dia 15 dos meses pares. E os preços de exercício são fixados próximos ao preço de mercado da ação objeto, acima e abaixo, e deixando o prêmio ser um parâmetro livre para negociação em pregão.

As opções não são negociadas diretamente pelos seus compradores e vendedores, e sim por corretoras que atuam em nome de seus clientes dentro de normas estabelecidas pela Bolsa.

Vamos fixar a notação para os elementos formadores de um contrato de opção européia:

- T: data de exercício do contrato;

- $S(t)$: preço do ativo-objeto na data t;

- $K_c(t), K_p(t)$: preço de exercício de uma opção de compra e venda, respectivamente, na data t;

- $C(t), P(t)$: preêmio de uma opção de compra e venda, respectivamente, na data t.

Vamos supor que as taxas de corretagens que fazem parte da negociação de opções já estão contabilizados no preço de exercício do contrato. Desta forma, uma opção de compra européia pode ser vista como um derivativo que paga ao seu titular, na data de exercício (T), a quantia

$$X(T) = \max(S(T) - K_c(t_0), 0) \quad (\geq 0)$$

onde t_0 a data de assinatura do contrato.

Se o preço de mercado do ativo-objeto for maior que o preço de exercício do contrato, ou seja, $S(T) > K_c(t_0)$, então o titular pode comprar o ativo por $K_c(t_0)$, em exercício do seu direito, e vendê-lo no mercado por $S(T)$, ganhando a diferença $S(T) - K_c(t_0)$. Esta diferença pode não cobrir o dinheiro gasto no prêmio pago ao emissor.

Se, por outro lado, $S(T) \leq K_c(t_0)$ perder-se-á $S(T) - K_c(t_0) \quad (\leq 0)$ se o direito de compra for exercido. Neste caso, deixa-se o contrato expirar sem exercício do direito de compra.

Analogamente, uma opção de venda européia pode ser vista como um derivativo cujo valor na data de exercício é dado por

$$X_T = \max(K_p(t_0) - S(T), 0) \quad (\geq 0)$$

Vamos especificar as taxas de retorno para o caso de ações e opções européias, com que vamos trabalhar para definir os momentos de primeira e segunda ordem de carteiras mistas.

Daqui por diante omitires a palavra européia quando falarmos de opções, ficando implícito que é sobre este tipo de contrato que estamos falando.

Para um ativo cujo preço de mercado na data t é dado por $S(t)$, a taxa de retorno r_s no perodo $\tau = T - t$ $(T > t)$ ser dada por

$$r_s = \frac{S(T) - S(t)}{S(t)} \tag{9.1}$$

Para as opes de compra e venda, as taxas de retorno sero dadas por

$$r_c = \frac{\max(S(T) - K_c(t_0), 0) - C(t_0)}{C(t_0)} \quad (\text{“call”}) \tag{9.2}$$

$$r_p = \frac{\max(K_p(t_0) - S(T), 0) - P(t_0)}{P(t_0)} \quad (\text{“put”}) \tag{9.3}$$

onde t_0 é a data de assinatura dos contratos.

9.2 Um Exemplo de Portflio

Vamos considerar um portfólio formado por dois ativos, chamados de ativo A e ativo B, e por opções de compra e venda que tém como ativos-objetos os próprios ativos A e B.

Vamos supor também que as datas de assinatura (t_0) e as datas de exercício (T) são as mesmas para todas as opções.

Denote por S_a e S_b os preços dos ativos A e B, respectivamente. Os retornos logarítmicos de A e B, extrapolados para a data T, são dados por

$$x = ln\left(\frac{S_a(T)}{S_a(t_0)}\right) \cong N(\mu_a, \sigma_a^2) \tag{9.4}$$

e

$$y = ln\left(\frac{S_b(T)}{S_b(t_0)}\right) \cong N(\mu_b, \sigma_b^2) \tag{9.5}$$

e as funções densidade de probabilidade dadas por

$$f(x) = \frac{1}{\sigma_a\sqrt{2\pi}} \exp\left\{-\frac{(x-\mu_a)^2}{2\sigma_a^2}\right\} \tag{9.6}$$

$$f(y) = \frac{1}{\sigma_b\sqrt{2\pi}} \exp\left\{-\frac{(y-\mu_b)^2}{2\sigma_b^2}\right\} \tag{9.7}$$

Sendo ρ o coeficiente de correlação entre os ativos A e B, a função distribuição conjunta é dada por

$$f(x,y) \quad = \frac{1}{\sigma_a\sigma_b\sqrt{1-\rho^2}} \exp\left\{-\frac{1}{2(1-\rho^2)}\left[\left(\frac{x-\mu_a}{\sigma_a}\right)^2 - 2\rho\frac{x-\mu_a}{\sigma_a}\frac{y-\mu_b}{\sigma_b} + \left(\frac{y-\mu_b}{\sigma_b}\right)^2\right]\right\}$$

$$(-\infty < x, y < +\infty)$$

Precisamos colocar as taxas de retorno dadas em 9.1, 9.2 e 9.3 em função das variveis x e y, pois é destas variáveis que temos informações sobre a densidade de probabilidade.

De 9.4 e 9.5 temos

$$S_a(T) = S_a(t_0)exp(x) \quad \text{e} \quad S_b(T) = S_b(t_0)exp(y) \tag{9.8}$$

Substituindo 9.8 em 9.1 teremos

$$r_{s_a} = exp(x) - 1 \quad \text{e} \quad r_{s_b} = exp(y) - 1 \tag{9.9}$$

onde r_{s_a} e r_{s_b} são as taxas de retorno dos ativos A e B, respectivamente.

Para as opções fixaremos a letra K para denotar preço de exercício do contrato, colocando um índice superior à letra para diferenciar calls de puts, e um índice inferior para indicar o ativo-objeto a que se refere. Por exemplo, $K_a^c(t)$ denota o preço de exercício de uma call sobre o ativo A, na data t, e $K_b^p(t)$ indica o preço de exercício de uma put sobre o ativo B, na data t.

Os prêmios das opções de compra serão denotados pela letra C, com um ndice inferior indicando o ativo a que se refere. O mesmo vale para as opçoes de venda, trocando a letra C pela letra P. Por exemplo, $C_a(t)$ indica o prmio de uma call sobre o ativo A na data t, e $P_b(t)$ indica o prmio de uma put sobre o ativo B, na data t.

Portanto, as taxas de retorno das opçoes de compra (assinadas na data t_0) que têm A e B como ativos-objeto serão dadas por

$$\begin{aligned} r_{c_a} &= (\max(S_a(t_0)e^x - K_a^c(t_0), 0) - C_a(t_0))/C_a(t_0) \\ r_{c_b} &= (\max(S_b(t_0)e^y - K_b^c(t_0), 0) - C_b(t_0))/C_b(t_0) \end{aligned} \tag{9.10}$$

Do mesmo modo para as puts,

$$\begin{aligned} r_{p_a} &= (\max(K_a^p(t_0) - S_a(t_0)e^x, 0) - P_a(t_0))/P_a(t_0) \\ r_{p_b} &= (\max(K_b^p(t_0) - S_b(t_0)e^y, 0) - P_b(t_0))/P_b(t_0) \end{aligned} \tag{9.11}$$

Quando não houver dúvidas quanto as datas que estamos usando, omitiremos a letra t dos prêmios, dos preços de exercício e das cotações dos ativos para não sobrecarregar a notação.

Uma outra forma de escrever as expressões 9.10 e 9.11 definir os domínios das v.a. x e y nos quais o termo $\max(S_0 exp(x) - K, 0)$ positivo

$$\begin{aligned} \max(S\exp(x) - K, 0) > 0 &\iff x > \ln\left(\tfrac{K}{S}\right) \\ \max(K - S\exp(x), 0) > 0 &\iff x < \ln\left(\tfrac{K}{S}\right). \end{aligned} \tag{9.12}$$

Considerando 9.12 nas expressões 9.10 e 9.11, redefinimos as taxas de retorno das opções por

$$r_{c_a} = \begin{cases} \frac{1}{C_a(t_0)}\left(S_a(t_0)e^x - K_a^c(t_0) - C_a(t_0)\right) & \text{se} \quad \ln\left(\frac{K_a^c(t_0)}{S_a(t_0)}\right) < x < +\infty \\ 0 & \text{c.c.} \end{cases}$$

$$r_{c_b} = \begin{cases} \frac{1}{C_b(t_0)}\left(S_b(t_0)e^y - K_b^c(t_0) - C_b(t_0)\right) & \text{se} \quad \ln\left(\frac{K_b^c(t_0)}{S_b(t_0)}\right) < y < +\infty \\ 0 & \text{c.c.} \end{cases}$$

$$r_{p_a} = \begin{cases} \frac{1}{P_a(t_0)}\left(K_a^p(t_0) - S_a(t_0)e^x - P_a(t_0)\right) & \text{se} \quad -\infty < x < \ln\left(\frac{K_a^p(t_0)}{S_a(t_0)}\right) \\ 0 & \text{c.c.} \end{cases}$$

$$r_{p_b} = \begin{cases} \frac{1}{P_b(t_0)}\left(K_b^p(t_0) - S_b(t_0)e^y - P_b(t_0)\right) & \text{se} \quad -\infty < y < \ln\left(\frac{K_b^p(t_0)}{S_b(t_0)}\right) \\ 0 & \text{c.c.} \end{cases} \tag{9.13}$$

O conhecimento de uma estimativa do retorno logarítmico para os ativos, no período $[t, T]$ considerado, é necessário para podermos definir as taxas de retorno dadas em 9.13. Estes dados podem ser obtidos através de análise de séries históricas de preços.

Observe que, para o processo de preços do ativo-objeto, que estamos supondo não ser um derivativo, a taxa de retorno $r = \frac{S_T}{S_0}$, $S_T > 0$ e $S_0 > 0$, não assume valor zero e nem valor negativo, estando, assim, o retorno logarítmico $x = lnr$ bem definido, independente da oscilação de preços dos ativos. Por outro lado, os retornos de opções, definidos pelas fórmulas 9.10 e 9.11 podem eventualmente assumir valor zero ou negativo, dependendo do comportamento dos preços do ativo, do preço de exercício e do prêmio. Por isso os retornos logarítmicos $\ln r_c$ e $\ln r_p$ para opções não estão bem definidos, podendo haver indeterminações quando $r_c \to 0$ e $r_p \to 0$. Para evitar inconsistências e padronizar o tipo de retorno utilizado para opções e ativo-objeto adotamos os retornos dados pelas expressões 9.1, 9.2 e 9.3.

A seguir derivaremos as expressões analíticas dos *valores esperados* das taxas dadas em 9.9 e 9.13 e das covariâncias entre elas, para o caso deste portfólio exemplo. Elas serão usadas para calcular o retorno esperado $E(r)$ e a matriz de covariância $Cov(r)$ de portfólios.

9.3 Esperança e Covariância do Retorno de Opções

Vamos voltar ao portfólio formado pelos ativos A e B e por opções sobre estes ativos. Manteremos a notação introduzida na seção anterior, com a exceção de que omitiremos o termo t_0 (data de assinatura das calls e das puts). Portanto os preços das ações serão dados por S_a e S_b, os prêmios das opções por C_a, C_b, P_a e P_b e os preços de exercício por K_a^c, K_a^p, K_b^c e K_b^p.

Vale lembrar ainda que dada uma v.a contínua x em $\mathcal{R}$ e sua respectiva função densidade de probabilidade $p(x)$, a esperança de uma função $g(x)$ é dada por

$$E[g(x)] = \int_{-\infty}^{+\infty} g(x)p(x)\,dx$$

Em virtude das repetições dos cálculos algébricos que envolvem a dedução das expressões das covariâncias, detalhamos esses cálculos em algumas delas. As demais usam raciocínio análogo.

Esperança das taxas de retorno

Notação: $\text{erf(a)} = \int_a^{+\infty} \exp(-\psi^2)\,d\psi$

$$\begin{aligned}
\mu_{sa} &= E[r_{s_a}] = \int_{-\infty}^{+\infty} \exp(x) f(x)\,dx - 1 \\
&= \frac{1}{\sigma_a\sqrt{2\pi}} \int_{-\infty}^{+\infty} \exp\left\{x - \frac{(x-\mu_a)^2}{2\sigma_a^2}\right\} dx - 1 \\
&\overset{1''}{=} \frac{\exp\left(\frac{1}{2}\sigma_a^2 + \mu_a\right)}{\sqrt{\pi}} \int_{-\infty}^{+\infty} \exp\left(-t^2\right)\,dt - 1 \\
&= \exp\left(\mu_a + \frac{1}{2}\sigma_a^2\right) - 1
\end{aligned}$$

$$\begin{aligned}
\mu_{ca} &= E[r_{c_a}] = \frac{1}{C_a}\int_{\ln\left(\frac{K_a^c}{S_a}\right)}^{+\infty}(S_a\exp(x)-K_a^c)f(x)\,dx - 1 \\
&= \frac{S_a}{C_a\sigma_a\sqrt{2\pi}}\int_{\ln\left(\frac{K_a^c}{S_a}\right)}^{+\infty}\exp\left\{x-\frac{(x-\mu_a)^2}{2\sigma_a^2}\right\}dx \\
&\quad -\frac{K_a^c}{C_a\sigma_a\sqrt{2\pi}}\int_{\ln\left(\frac{K_a^c}{S_a}\right)}^{+\infty}\exp\left\{\frac{(x-\mu_a)^2}{2\sigma_a^2}\right\}dx - 1 \\
&\overset{1''}{=} \frac{S_a}{2C_a}\exp\left(\mu_a+\frac{1}{2}\sigma_a^2\right)\left\{1-erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^c}{S_a}\right)-\mu_a-\sigma_a^2\right)\right]\right\} \\
&\quad -\frac{1}{2C_a}(K_a^c)\left\{1-erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^c}{S_a}\right)-\sigma_a^2\right)\right]\right\} - 1
\end{aligned}$$

Mudança de variável usada:

$$1'': \begin{cases} t &= \frac{1}{\sigma_a\sqrt{2}}(x-\mu_a-\sigma_a^2) \\ \\ w &= \frac{1}{\sigma_a\sqrt{2}}(x-\sigma_a) \end{cases}$$

$$\begin{aligned}
\mu_{pa} &= E[r_{p_a}] = \frac{1}{P_a}\int_{-\infty}^{\ln\left(\frac{K_a^p}{S_a}\right)}(K_a^p - S_a\exp(x))f(x)\,dx - 1 \\
&\overset{1''}{=} \frac{1}{2P_a}(K_a^p)\exp\left(\mu_a+\frac{1}{2}\sigma_a^2\right)\left\{1+erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^p}{S_a}\right)-\mu_a-\sigma_a^2\right)\right]\right\} \\
&\quad -\frac{S_a}{2P_a}\left\{1+erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^p}{S_a}\right)-\sigma_a^2\right)\right]\right\} - 1
\end{aligned}$$

Covariâncias

No lugar de calcularmos as expressões de covariâncias para as taxas de retorno simples dadas em 9.9, 9.10 e 9.11 deduziremos expressões para o retorno simples ($= \frac{\text{preço final}}{\text{preço inicial}}$) de ações e opções, sendo a diferença entre as duas grandezas apenas uma constante, isto é,

Taxa de retorno simples = Retorno simples - 1.

O resultado numérico de covariâncias não muda, visto que para duas v.a. X e Y e constantes k_1 e k_2 vale a igualdade

$$cov(X + k_1, Y + k_2) = cov(X, Y).$$

- $Cov(r_{s_a}, r_{s_b})$

$$\begin{aligned} cov(r_{s_a}, r_{s_a}) &= var(r_{s_a}) \\ &= E_{2x} - E_x^2 \end{aligned}$$

$$\begin{aligned} E_{2x} &= \int_{-\infty}^{+\infty} exp(2x) f(x)\, dx \\ &= \frac{1}{\sigma_a\sqrt{2\pi}} \int_{-\infty}^{+\infty} \exp\left\{2x - \frac{(x-\mu_a)^2}{2\sigma_a^2}\right\} dx \\ &\overset{2''}{=} \frac{1}{\sqrt{\pi}} \exp\left(2\mu_a + 2\sigma_a^2\right) \int_{-\infty}^{+\infty} \exp(-\psi^2)\, d\psi \\ &= \exp\left(2\mu_a + 2\sigma_a^2\right) \end{aligned}$$

$$E_x = \exp\left(\mu_a + 0.5\sigma_a^2\right)$$

onde 2" indica a mudança de variável

$$\psi = \frac{x - \mu_a - 2\sigma_a^2}{\sigma_a\sqrt{2}}$$

Logo,

$$var(r_{s_a}) = \exp\left(2\mu_a + \sigma_a^2\right)\left(\exp(\sigma_a^2) - 1\right)$$

- $Cov(r_{s_a}, r_{s_b})$

$$\begin{aligned} cov(r_{s_a}, r_{s_b}) &= E[r_{s_a} r_{s_b}] - E[r_{s_a}]E[r_{s_b}] \\ &= E_{xy} - E_x E_y \end{aligned}$$

sendo

$$\begin{aligned}
E_{xy} &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)\exp(x+y)\,dy\,dx \\
&= \frac{1}{2\pi\sigma_a\sigma_b\sqrt{1-\rho^2}}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\exp(x+y) \\
&\quad \cdot\exp\left\{-\frac{1}{2(1-\rho^2)}\left[(\frac{x-\mu_a}{\sigma_a})^2 - 2\rho\frac{x-\mu_a}{\sigma_a}\frac{y-\mu_b}{\sigma_b} + (\frac{y-\mu_b}{\sigma_b})^2\right]\right\}\,dy\,dx \\
&\overset{3''}{=} \frac{\exp(\mu_a+\mu_b)}{2\pi\sqrt{1-\rho^2}}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\exp\left\{\sigma_a u + \sigma_b v - \frac{1}{2(1-\rho^2)}(u^2 - 2\rho uv + v^2)\right\}\,dv\,du \\
&\overset{4''}{=} \frac{1}{\pi\sqrt{2}}\exp\left(\mu_a+\mu_b+\frac{1}{2}\sigma_b^2(1-\rho^2)\right) \\
&\quad \cdot\int_{-\infty}^{+\infty}\left(\int_{-\infty}^{+\infty}\exp(-\psi^2)\,d\psi\right)\exp(-\frac{1}{2}(u^2 - 2u(\sigma_a+\sigma_b\rho)))\,du \\
&\overset{5''}{=} \frac{1}{\sqrt{\pi}}\exp\left(\mu_a+\mu_b+\frac{1}{2}\sigma_b^2(1-\rho^2)+\frac{1}{2}(\sigma_a+\sigma_b\rho)^2\right)\int_{-\infty}^{+\infty}\exp(-\psi^2)\,d\psi \\
&= \exp\left(\mu_a+\mu_b+\frac{1}{2}(\sigma_b^2+\sigma_a^2)+\sigma_a\sigma_b\rho\right)
\end{aligned}$$

$$
\begin{aligned}
E_x &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)\exp(x)\,dy\,dx \\
&\overset{3''}{=} \frac{\exp(\mu_a)}{2\pi\sqrt{1-\rho^2}}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\exp\left\{\sigma_a u - \frac{1}{2(1-\rho^2)}(u^2-\rho uv+v^2)\right\}dv\,du \\
&\overset{6''}{=} \frac{1}{\pi\sqrt{2}}exp(\mu_a)\int_{-\infty}^{+\infty}\left(\int_{-\infty}^{+\infty}\exp(-\psi^2)\,d\psi\right)\exp(-\frac{1}{2}(u^2-2u\sigma_a))\,du \\
&\overset{7''}{=} \frac{1}{\sqrt{\pi}}\exp\left(\mu_a+\frac{1}{2}\sigma_a^2\right)\int_{-\infty}^{+\infty}\exp(-\psi^2)\,d\psi \\
&= \exp\left(\mu_a+\frac{1}{2}(\sigma_a^2)\right)
\end{aligned}
$$

$$
E_y = \exp(\mu_b+\frac{1}{2}\sigma_b^2)
$$

Mudanças de variável:

$$
3'': \begin{cases} u &= (x-\mu_a)/\sigma_a \\ \\ v &= (y-\mu_b)/\sigma_b \end{cases}
$$

$$
4'': \quad \psi = \frac{1}{\sqrt{2(1-\rho^2)}}(v-u\rho-\sigma_b(1-\rho^2))
$$

$$
5'': \quad w = \frac{1}{\sqrt{2}}(u-\sigma_a-\sigma_b\rho)
$$

$$
6'': \quad \psi = \frac{1}{\sqrt{2(1-\rho^2)}}(v-u\rho)
$$

$$
7'': \quad w = \frac{1}{\sqrt{2}}(u-\sigma_a)
$$

Portanto,

$$
cov(r_{s_a}, r_{s_b}) = \exp(\mu_a+\mu_b+\frac{1}{2}(\sigma_a^2+\sigma_v^2))\,(\exp(\sigma_a\sigma_b\rho)-1))
$$

- $Cov(r_{s_a}, r_{c_a})$

$$cov(r_{s_a}, r_{c_a}) = \frac{1}{C_a}\{S_a E_{2x} - K_a^c E_x\} - \mu_{sa}\mu_{ca}$$

onde

$$E_{2x} = \frac{1}{2}\exp(2\mu_a + 2\sigma_a^2)\left\{1 - erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^c}{S_a}\right) - \mu_a - 2\sigma_a^2\right)\right]\right\}$$

$$E_x = \frac{1}{2}\exp(\mu_a + \frac{1}{2}\sigma_a^2)\left\{1 - erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^c}{S_a}\right) - \mu_a - \sigma_a^2\right)\right]\right\}$$

- $Cov(r_{s_a}, r_{p_a})$

$$cov(r_{s_a}, r_{p_a}) = \frac{1}{P_a}\{-S_a E_{2x} + K_a^p E_x\} - \mu_{sa}\mu_{pa}$$

onde

$$E_{2x} = \frac{1}{2}\exp(2\mu_a + 2\sigma_a^2)\left\{1 + erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^p}{S_a}\right) - \mu_a - 2\sigma_a^2\right)\right]\right\}$$

$$E_x = \frac{1}{2}\exp(\mu_a + \frac{1}{2}\sigma_a^2)\left\{1 + erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^p}{S_a}\right) - \mu_a - \sigma_a^2\right)\right]\right\}$$

- $Cov(r_{s_a}, r_{c_b})$

$$cov(r_{s_a}, r_{c_b}) = \frac{1}{C_b}\{S_b E_{xy} - E_x K_b^c\} - \mu_{cb}\bar{E}_x - \frac{\mu_{sa}}{C_b}\{S_b E_y - K_b^c E_1\} + \mu_{sa}\mu_{cb}$$

onde

$$R_b^c = \frac{1}{\sigma_b}(\ln\left(\frac{K_b^c}{S_b}\right) - \mu_b)$$

$$E_{xy} = \frac{1}{2}\exp(\mu_a+\mu_b+\frac{1}{2}(\sigma_a^2+\sigma_b^2)+\rho\sigma_a\sigma_b)\left\{1-erf\left[\frac{1}{\sqrt{2}}(R_b^c-\sigma_b-\sigma_a\rho)\right]\right\}$$

$$E_x = \frac{1}{2}\exp(\mu_a+\frac{1}{2}\sigma_a^2)\left\{1-erf\left[\frac{1}{\sqrt{2}}\left(R_b^c-\rho\sigma_a^2\right)\right]\right\}$$

$$E_y = \frac{1}{2}\exp(\mu_b+\frac{1}{2}\sigma_b^2)\left\{1-erf\left[\frac{1}{\sqrt{2}}\left(R_b^c-\sigma_b\right)\right]\right\}$$

$$\bar{E}_x = \frac{1}{2}\exp(\mu_a+\frac{1}{2}\sigma_a^2)$$

$$E_1 = \frac{1}{2}\left\{1-erf\left[\frac{1}{\sqrt{2}}\left(R_b^c\right)\right]\right\}$$

- $Cov(r_{s_a}, r_{p_b})$

$$cov(r_{s_a}, r_{p_b}) = \frac{1}{P_b}\left\{-S_bE_{xy}+E_xK_b^p\right\}-\mu_{pb}\bar{E}_x-\frac{\mu_{sa}}{P_b}\left\{-S_bE_y+K_b^cE_1\right\}+\mu_{sa}\mu_{pb}$$

onde

$$R_b^p = \frac{1}{\sigma_b}(\ln\left(\frac{K_b^p}{S_b}\right)-\mu_b)$$

$$E_{xy} = \frac{1}{2}\exp(\mu_a+\mu_b+\frac{1}{2}(\sigma_a^2+\sigma_b^2)+\rho\sigma_a\sigma_b)\left\{1+erf\left[\frac{1}{\sqrt{2}}(R_b^c-\sigma_b-\sigma_a\rho)\right]\right\}$$

$$E_x = \frac{1}{2}\exp(\mu_a+\frac{1}{2}\sigma_a^2)\left\{1+erf\left[\frac{1}{\sqrt{2}}\left(R_b^p-\rho\sigma_a\right)\right]\right\}$$

$$E_y = \frac{1}{2}\exp(\mu_b+\frac{1}{2}\sigma_b^2)\left\{1+erf\left[\frac{1}{\sqrt{2}}\left(R_b^p-\sigma_b\right)\right]\right\}$$

$$\bar{E}_x = \frac{1}{2}\exp(\mu_a+\frac{1}{2}\sigma_a^2)$$

$$E_1 = \frac{1}{2}\left\{1+erf\left[\frac{1}{\sqrt{2}}\left(R_b^c\right)\right]\right\}$$

- $Cov(r_{c_a}, r_{p_a})$

$$cov(r_{c_a}, r_{p_a}) = \begin{cases} \frac{1}{C_a P_a}\left\{-S_a^2 E_{2x} + S_a E_x (K_a^c + K_a^p) - K_a^c K_a^p E_1\right\} - \mu_{ca}\mu_{pa}, & \text{se} \quad K_a^c > K_a^p \\ -\mu_{ca}\mu_{pa}, & \text{se} \quad K_a^c \leq K_a^p \end{cases}$$

onde

$$\begin{aligned} E_{2x} &= \frac{1}{2}\exp(2\mu_a + 2\sigma_a^2) \\ &\quad \cdot \left\{ erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^p}{S_a}\right) - \mu_a - 2\sigma_a^2\right)\right] - erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^c}{S_a}\right) - \mu_a - 2\sigma_a^2\right)\right]\right\} \end{aligned}$$

$$\begin{aligned} E_x &= \frac{1}{2}\exp(\mu_a + \frac{1}{2}\sigma_a^2) \\ &\quad \cdot \left\{ erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^p}{S_a}\right) - \mu_a - \sigma_a^2\right)\right] - erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^c}{S_a}\right) - \mu_a - \sigma_a^2\right)\right]\right\} \end{aligned}$$

$$E_1 = \frac{1}{2}\left\{ erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^p}{S_a}\right) - \mu_a\right)\right] - erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K_a^c}{S_a}\right) - \mu_a\right)\right]\right\}$$

- $Cov(r_{c_a}, r_{c_b})$

$$\begin{aligned} cov(r_{c_a}, r_{c_b}) &= \frac{1}{C_a C_b}\left\{S_a S_b E_{xy} - S_a E_x K_b^c - S_b E_y K_a^c\right\} \\ &\quad -\frac{\mu_{cb}}{C_a}(S_a \bar{E}_x - K_a^c \bar{E}_1) - \frac{\mu_{ca}}{C_b}(S_b \bar{\bar{E}}_y - K_b^c \bar{\bar{E}}_1) + \mu_{ca}\mu_{cb} \end{aligned}$$

onde

$$R_a^c = \frac{1}{\sigma_b}(\ln\left(\frac{K_a^c}{S_b}\right) - \mu_a)$$

$$R_b^c = \frac{1}{\sigma_b}(\ln\left(\frac{K_b^c}{S_b}\right) - \mu_b)$$

$$\begin{aligned} E_{xy} &= \frac{1}{2\sqrt{2\pi}} \exp(\mu_a + \mu_b + \frac{1}{2}(\sigma_a^2 + \sigma_b^2) + \rho\sigma_a\sigma_b) \\ &\cdot \int_{R_a^c}^{+\infty} \left\{ 1 - erf\left[\frac{1}{\sqrt{2(1-\rho^2)}} (R_b^c - \sigma_b(1-\rho^2) - \rho u) \right] \right\} \\ &\cdot exp(-\frac{1}{2}(u - \sigma_a - \rho\sigma_b)^2)\, du \end{aligned}$$

$$\begin{aligned} E_x &= \frac{1}{2\sqrt{2\pi}} \exp(\mu_a + \frac{1}{2}\sigma_a^2) \\ &\cdot \int_{R_a^c}^{+\infty} \left\{ 1 - erf\left[\frac{1}{\sqrt{2(1-\rho^2)}} (R_b^c - \rho u) \right] \right\} \\ &\cdot \exp(-\frac{1}{2}(u - \sigma_a)^2)\, du \end{aligned}$$

$$\begin{aligned} E_y &= \frac{1}{2\sqrt{2\pi}} \exp(\mu_b + \frac{1}{2}\sigma_b^2) \\ &\cdot \int_{R_a^c}^{+\infty} \left\{ 1 - erf\left[\frac{1}{\sqrt{2(1-\rho^2)}} \left(R_b^c - \rho u - \sigma_b(1-\rho^2)\right) \right] \right\} \\ &\cdot \exp(-\frac{1}{2}(u - \sigma_b\rho)^2)\, du \end{aligned}$$

$$E_1 = \frac{1}{2\sqrt{2\pi}} \int_{R_a^c}^{+\infty} \left\{ 1 - erf\left[\frac{1}{\sqrt{2(1-\rho^2)}} (R_b^c - \rho u) \right] \right\} \exp(-\frac{1}{2}u^2)\, du$$

$$\bar{E}_x = \frac{1}{2} \exp(\mu_a + \frac{1}{2}\sigma_a^2) \left\{ 1 - erf\left[\frac{1}{\sqrt{2}} (R_a^c - \sigma_a) \right] \right\}$$

$$\bar{E}_1 = \frac{1}{2} \left\{ 1 - erf\left[\frac{R_a^c}{\sqrt{2}} \right] \right\}$$

$$\bar{\bar{E}}_y = \frac{1}{2} \exp(\mu_b + \frac{1}{2}\sigma_b^2) \left\{ 1 - erf\left[\frac{1}{\sqrt{2}} (R_b^c - \sigma_b) \right] \right\}$$

$$\bar{\bar{E}}_1 = \frac{1}{2} \left\{ 1 - erf\left[\frac{R_b^c}{\sqrt{2}} \right] \right\}$$

- $Cov(r_{c_a}, r_{p_b})$

$$\begin{aligned} cov(r_{c_a}, r_{p_b}) &= \frac{1}{C_a P_b}\{-S_a S_b E_{xy} + S_a E_x K_b^p + S_b E_y K_a^c - K_a^c K_b^p E_1\} \\ &\quad -\frac{\mu_{pb}}{C_a}(S_a \bar{E}_x - K_a^c \bar{E}_1) - \frac{\mu_{ca}}{P_b}(-S_b \bar{\bar{E}}_y + K_b^p \bar{\bar{E}}_1) + \mu_{ca}\mu_{pb} \end{aligned}$$

onde

$$R_a^c = \frac{1}{\sigma_b}(\ln\left(\frac{K_a^c}{S_b}\right) - \mu_a)$$

$$R_b^p = \frac{1}{\sigma_b}(\ln\left(\frac{K_b^p}{S_b}\right) - \mu_b)$$

Os valores de E_{xy}, E_x, E_y , E_1, $\bar{E}_x$,$\bar{E}_1$,$\bar{\bar{E}}_x$ e $\bar{\bar{E}}_1$ são iguais às expressões de $cov(r_{c_a}, r_{c_b})$, trocando o sinal da função erf e trocando também R_b^c por R_b^p.

- $Cov(r_{p_a}, r_{p_b})$

$$\begin{aligned} cov(r_{p_a}, r_{p_b}) &= \frac{1}{P_a P_b}\{S_a S_b E_{xy} - S_a E_x K_b^p - S_b E_y K_a^p + K_a^p K_b^p E_1\} \\ &\quad +\frac{\mu_{pb}}{P_a}(S_a \bar{E}_x - K_a^p \bar{E}_1) + \frac{\mu_{pa}}{P_b}(S_b \bar{\bar{E}}_y - K_b^p \bar{\bar{E}}_1) + \mu_{pa}\mu_{pb} \end{aligned}$$

onde

$$R_a^p = \frac{1}{\sigma_b}(\ln\left(\frac{K_a^p}{S_a}\right) - \mu_a)$$

$$R_b^p = \frac{1}{\sigma_b}(\ln\left(\frac{K_b^p}{S_b}\right) - \mu_b)$$

$$\begin{aligned} E_{xy} &= \frac{1}{2\sqrt{2\pi}}\exp(\mu_a + \mu_b + \frac{1}{2}(\sigma_a^2 + \sigma_b^2) + \rho\sigma_a\sigma_b) \\ &\quad \cdot \int_{-\infty}^{R_a^p}\left\{1 + erf\left[\frac{1}{\sqrt{2(1-\rho^2)}}(R_b^p - \sigma_b(1-\rho^2) - \rho u)\right]\right\} \\ &\quad \cdot exp(-\frac{1}{2}(u - \sigma_a - \rho\sigma_b)^2)\, du \end{aligned}$$

$$\begin{aligned} E_x &= \frac{1}{2\sqrt{2\pi}} \exp(\mu_a + \frac{1}{2}\sigma_a^2) \\ &\cdot \int_{-\infty}^{R_a^p} \left\{ 1 + erf \left[\frac{1}{\sqrt{2(1-\rho^2)}} (R_b^p - \rho u) \right] \right\} \\ &\cdot \exp(-\frac{1}{2}(u - \sigma_a)^2)\, du \end{aligned}$$

$$\begin{aligned} E_y &= \frac{1}{2\sqrt{2\pi}} \exp(\mu_b + \frac{1}{2}\sigma_b^2) \\ &\cdot \int_{-\infty}^{R_a^p} \left\{ 1 + erf \left[\frac{1}{\sqrt{2(1-\rho^2)}} \left(R_b^p - \rho u - \sigma_b(1-\rho^2)\right) \right] \right\} \\ &\cdot \exp(-\frac{1}{2}(u - \sigma_b\rho)^2)\, du \end{aligned}$$

$$E_1 = \frac{1}{2\sqrt{2\pi}} \int_{-\infty}^{R_a^p} \left\{ 1 + erf \left[\frac{1}{\sqrt{2(1-\rho^2)}} (R_b^p - \rho u) \exp(-\frac{1}{2}u^2) \right] \right\} du$$

$$\bar{E}_x = \frac{1}{2} \exp(\mu_a + \frac{1}{2}\sigma_a^2) \left\{ 1 + erf \left[\frac{1}{\sqrt{2}} (R_a^p - \sigma_a) \right] \right\}$$

$$\begin{aligned} \bar{E}_1 &= \frac{1}{2} \left\{ 1 + erf \left[\frac{R_a^p}{\sqrt{2}} \right] \right\} \\ \bar{\bar{E}}_y &= \frac{1}{2} \exp(\mu_b + \frac{1}{2}\sigma_b^2) \left\{ 1 + erf \left[\frac{1}{\sqrt{2}} (R_b^p - \sigma_b) \right] \right\} \end{aligned}$$

$$\bar{\bar{E}}_1 = \frac{1}{2} \left\{ 1 + erf \left[\frac{R_b^p}{\sqrt{2}} \right] \right\}$$

- $Cov(r_{c_{a1}}, r_{c_{a2}})$

  Neste caso, vamos calcular a covariância entre as taxas de retorno de duas opções de compra sobre o mesmo ativo A, mesma data de expiração, mas com preços de exercício diferentes e, portanto, prêmios diferentes.

  Sejam K_1^c e K_2^c os preços de exercício e C_1 e C_2 os prêmios das calls.

$$cov(r_{c_{a1}}, r_{c_{a2}}) = \frac{1}{C_1 C_2}\left\{S_a^2 E_{2x} - S_a E_x(K_1^c + K_2^c) + K_1^c K_2^c E_1\right\} - \mu_{c1}\mu_{c2}$$

$$K = \max\{K_1^c, K_2^c\}$$

onde

$$E_{2x} = \frac{1}{2}\exp(2\mu_a + 2\sigma_a^2)\left\{1 - erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K}{S_a}\right) - \mu_a - 2\sigma_a^2\right)\right]\right\}$$

$$E_x = \frac{1}{2}\exp(\mu_a + \frac{1}{2}\sigma_a^2)\left\{1 - erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K}{S_a}\right) - \mu_a - \sigma_a^2\right)\right]\right\}$$

$$E_1 = \frac{1}{2}\left\{1 - erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K}{S_a}\right) - \mu_a\right)\right]\right\}$$

- $Cov(r_{p_{a1}}, r_{p_{a2}})$

  Aqui calculamos a covariância entre as taxas de retorno de duas opções de venda sobre o mesmo ativo A, mesma data de expiração, mas com preços de exercício e prêmios diferentes.

  Sejam K_1^p e K_2^p os preços de exercício e P_1 e P_2 os prêmios das calls.

$$cov(r_{p_{a1}}, r_{p_{a2}}) = \frac{1}{P_1 P_2}\left\{S_a^2 E_{2x} - S_a E_x(K_1^p + K_2^p) + K_1^p K_2^p E_1\right\} - \mu_{p1}\mu_{p2}$$

$$K = \min\{K_1^p, K_2^p\}$$

onde

$$E_{2x} = \frac{1}{2}\exp(2\mu_a + 2\sigma_a^2)\left\{1 + erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K}{S_a}\right) - \mu_a - 2\sigma_a^2\right)\right]\right\}$$

$$E_x = \frac{1}{2}\exp(\mu_a + \frac{1}{2}\sigma_a^2)\left\{1 + erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K}{S_a}\right) - \mu_a - \sigma_a^2\right)\right]\right\}$$

$$E_1 = \frac{1}{2}\left\{1 + erf\left[\frac{1}{\sigma_a\sqrt{2}}\left(\ln\left(\frac{K}{S_a}\right) - \mu_a\right)\right]\right\}$$

- $Cov(r_{c_{aT}}, r_{c_{a\tau}})$

  Neste caso calculamos a covariância entre duas opções de compra sobre o mesmo ativo, mas com datas de expiração diferentes (dadas por T e τ, $\tau < T$).

  Suponha que temos a informação

$$x_T = \ln\frac{S_a(T)}{S_a(t_0)} \cong N(\mu, \sigma^2)$$

  Denote por x_τ o retorno logarítimico do ativo no período t_0 a τ e $x_{T-\tau}$ o retorno logarítimico do ativo no período τ a T. Então

$$x_\tau = \ln\frac{S_a(\tau)}{S_a(t_0)} \cong N(\mu_\tau, \sigma_\tau^2)$$

$$x_{T-\tau} = \ln\frac{S_a(T-\tau)}{S_a(\tau)} \cong N(\mu_{T-\tau}, \sigma_{T-\tau}^2)$$

  onde

$$\begin{aligned}
\mu_\tau &= \mu_T\frac{\tau - t_0}{T - t_0} \\
\sigma_\tau^2 &= \sigma_T^2\frac{\tau - t_0}{T - t_0} \\
\mu_{T-\tau} &= \mu_T\frac{T - \tau}{T - t_0} \\
\sigma_{T-\tau}^2 &= \sigma_T^2\frac{T - \tau}{T - t_0}
\end{aligned}$$

  E a correlação entre x_τ e x_T é dada por

$$corr(x_T, x_\tau) = \frac{\sigma_T^2 + \sigma_\tau^2 - \sigma_{T-\tau}^2}{2\sigma_T\sigma_\tau}$$

Desta forma, podemos calcular a covariância entre as duas opções usando a mesma expressão da $Cov(r_{c_a}, r_{c_b})$, considerando $x = x_T$, $y = x_\tau$ e $\rho = corr(x_T, x_\tau)$.

- $Cov(r_{p_{aT}}, r_{p_{a\tau}})$

  Idem ao caso anterior, onde a expressão usada para o cálculo da covariância é a dada por $Cov(r_{p_a}, r_{p_b})$.

- $Cov(r_{c_{aT}}, r_{p_{a\tau}})$

  Idem aos casos anteriores, usando a expressão dada em $Cov(r_{c_a}, r_{p_b})$ para o cálculo da covariância.

Appendix A

Matlab

A.1 Histórico

Matlab, de Matrix Laboratory, é um ambiente interativo para computação envolvendo matrizes. Matlab foi desenvolvido no início da década de 80 por Cleve Moler, no Departamento de Ciência da Computação da Universidade do Novo México, EUA.

Matlab coloca à disposição do usuário, num ambiente interativo, as bibliotecas desenvolvidas nos projetos LINPACK e EISPACK. Estes projetos elaboraram bibliotecas de domínio público para Álgebra Linear. LINPACK tem rotinas para solução de sistemas de equações lineares, e EISPACK tem rotinas para cálculo de autovalores. Os manuais destes projetos são portanto documentação complementar à documentação do Matlab.

Versões posteriores de Matlab foram desenvolvidas na firma comercial MathWorks Inc., atualmente na versão 5.1, que detêm os direitos autorais destas implementações. As versões recentes do produto MATLAB melhoram significativamente o ambiente interativo, incluindo facilidades gráficas de visualização e impressão; todavia a "Linguagem Matlab" manteve-se quase inalterada. Existem vários interpretadores da linguagem Matlab em domínio publico, como Matlab 1.0, Octave, rlab e Scilab. Existem também outros interpretadores comerciais de Matlab, como CLAM. Existem ainda várias Tool Boxes, bibliotecas vendidas pela MathWorks e por terceiros, com rotinas em Matlab para áreas específicas.

Usaremos a grafia de nome próprio, **Matlab**, como referência a linguagem, o nome em maiúsculas, **MATLAB**, como referência ao produto comercial da MathWorks, e a grafia em minúsculas, **matlab**, como referência a um interpretador genérico da linguagem Matlab.

A.2 O Ambiente

Para entrar no ambiente matlab, simplesmente digite "matlab". O prompt do matlab é `>>` , que espera por comandos. Para sair use o comando `quit`. Dentro do ambiente matlab, um comando precedido do bang, `!`, é executado pelo sistema operacional, assim: usando `!dir` ou `!ls` ficaremos sabendo os arquivos no diretório corrente, e usando `!edit` , `!vi` ou `!emacs` , seremos capazes de editar um arquivo. Normalmente Matlab distingue maiúsculas de minúsculas.

O comando `help` exibe todos os comandos e símbolos sintáticos disponíveis. O comando `help nomecom` fornece informações sobre o comando de nome *nomecom*. O comando `diary nomearq` abre um arquivo de nome *nomearq*, e ecoa tudo que aparece na tela para este arquivo. Repetindo o comando `diary` fechamos este arquivo. O formato dos números na tela pode ser alterado com o comando `format`.

Os comandos `who` e `whos` listam as variáveis em existência no espaço de trabalho. O comando `clear` limpa o espaço de trabalho, extinguindo todas as variáveis. O comando `save nomearq` guarda o espaço de trabalho no arquivo *nomearq*, e o comando `load nomearq` carrega um espaço de trabalho previamente guardado com o comando `save`.

Em Matlab há dois terminadores de comando: a vírgula, `,` , e o ponto-e-vírgula, `;` . O resultado de um comando terminado por vírgula é ecoado para a tela. O terminador ponto-e-vírgula não causa eco. Resultados ecoados na tela são atribuídos a variável do sistema `ans` (de answer, ou resposta). O terminador vírgula pode ser suprimido no último comando da linha. Para continuar um comando na linha seguinte devemos terminar a linha corrente com três pontos, `...` , o símbolo de continuação. O sinal de porcento, `%` , indica que o resto da linha é comentário.

A.3 Matrizes

O tipo básico do Matlab é uma matriz de números complexos. Uma matriz real é uma matriz que tem a parte imaginária de todos os seus elementos nula. Um vetor linha é uma matriz $1 \times n$, um vetor coluna uma matriz $n \times 1$, e um escalar uma matriz 1×1.

As atribuições
`A = [1, 2, 3; 4, 5, 6; 7, 8, 9];` ou
`A = [1 2 3; 4 5 6; 7 8 9];` ,
são equivalentes, e atribuem à variável A o valor

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Matrizes são delimitadas por colchetes, elementos de uma mesma linha são separados por vírgulas (ou apenas por espaços em branco), e linhas são separadas por ponto-e-vírgula (ou pelo caracter nova-linha). O apóstrofe, `'` , transpõem uma matriz. É fácil em Matlab compor matrizes blocadas, desde que os blocos tenham dimensões compatíveis! Por exemplo:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \ , \ B = \begin{bmatrix} 5 \\ 6 \end{bmatrix} \ , \ C = \begin{bmatrix} 7 \\ 8 \\ 9 \end{bmatrix} \ , \ \texttt{D=[A,B;C']}; \ D = \begin{bmatrix} 1 & 2 & 5 \\ 3 & 4 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Cuidado ao concatenar matrizes com os espacos em branco, pois estes são equivalentes a vírgulas, separando elementos. Assim: `[1,2+3]==[1 5]` mas `[1,2 +3]==[1,2,3]`.

Há várias funções para gerar matrizes e vetores especiais: `zeros(m,n)`, `ones(m,n)` e `rand(m,n)` são matrizes de dimensão $m \times n$ com, respectivamente, zeros, uns, e números aleatórios em [0,1]. O vetor `i:k:j` é o vetor linha $[i, i+k, i+2k, \ldots i+nk]$, onde $n = \max m \mid i + mk \leq j$. Podemos suprimir o "passo" $k = 1$, escrevendo o vetor `i:1:j` simplesmente como `i:j`. Se v é um vetor, `diag(v)` é a matriz diagonal com diagonal v, se A é uma matriz quadrada, `diag(A)` é o vetor com os elementos da diagonal principal de A. A matriz identidade de ordem n é `eye(n)`, o mesmo que `diag(ones(1,n))`.

`A(i,j)` é o elemento na i-ésima linha, j-ésima coluna de A, $m \times n$. Se vi e vj são vetores de índices em A, i.e. vetores linha com elementos inteiros positivos, em vi não excedendo m, e em vj não excedendo n, `A(vi,vj)` é a sub-matriz do elementos de A com índice de linha em vi e índice de coluna em vj. Em particular `A(1:m,j)`, ou `A(:,j)` é a j-ésima coluna de de A, e `A(i,1:j)`, ou `A(i,:)`, é a i-ésima linha de A. Exemplo:

$$A = \begin{bmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \\ 31 & 32 & 33 \end{bmatrix} \quad vi = \begin{bmatrix} 3 \\ 1 \end{bmatrix} \quad vj = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \quad A(vi, vj) = \begin{bmatrix} 32 & 33 \\ 12 & 13 \end{bmatrix}$$

As operações de adição, subtração, produto, divisão e potência, `+ - * / ^`, são as usuais da álgebra linear. O operador de divisão à esquerda, `\`, fornece em `x = A\b;` uma solução $x \mid A * x = b$. O operador de divisão a direita, `/` , fornece em `a = b/A;` uma solução $x \mid x * A = b$. Quando o sistema é bem determinado isto é o mesmo que, respectivamente, `inv(A)*b` ou `b*inv(A)`. (Quando o sistema é super-determinado x é uma solução no sentido dos mínimos quadrados.) Os operadores aritméticos de produto, potência e divisão tem a versão pontual, `.* .^ ./ .\` , que são executados elemento a elemento. Exemplo:

$$x = \begin{bmatrix} 5 & 5 & 5 \end{bmatrix} \ , \ y = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \ , \ x .* y = \begin{bmatrix} -5 & 0 & 5 \end{bmatrix}$$

Matlab é uma linguagem declarativa (em oposição a imperativa) onde as variáveis são declaradas, dimensionadas e redimensionadas dinamicamente pelo interpretador. Assim,

se A presentemente não existe, `A=11;` declara e inicializa uma matriz real 1×1. Em seguida, o comando `A(2,3)=23;` redimensionaria A como a matriz 2×3 `[11, 0, 0; 0, 0, 23]`. A nome da matriz nula é `[ ]`. A atribuição `A(:,2)=[];` anularia a segunda coluna de A, tornado-a a matriz 2×2 `[11, 0; 0, 23]`.

A.4 Controle de Fluxo

Os operadores relacionais da linguagem são `< <= > >= == ~=`, que retornam valor 0 ou 1 conforme a condição seja verdadeira ou falsa. Os operadores lógicos, *não e ou*, são, respectivamente, `~ & |` .

Matlab possui os comandos de fluxo *for – end*, *while – end* e *if – elseif – else – end*, que tem a mesma sintaxe do Fortran, exemplificada a seguir. Lembre-se de **não** escrever a palavra `elseif` como duas palavras separadas.

```
for i=1:n
  for j=1:m
    H(i,j)=1/(i+j-1);
  end
end
```

```
if(x<0)
  signx=-1;
elseif(x>0)
  signx=1;
else
  signx=0;
end
```

```
fatn=1;
while(n>1)
  fatn=fatn*n;
  n=n-1;
end
```

Uma consideração sobre eficiência: Matlab é uma linguagem interpretada, e tempo de interpretação de um comando simples pode ser bem maior que seu tempo de execução. Para tornar o programa rápido, tente operar sobre matrizes ou blocos, evitando loops explícitos que operem numa matriz elemento por elemento. Em outras palavras, tente evitar loops que repetem um comando que atribui valores a elementos, por atribuições a vetores ou matrizes. As facilidades no uso de vetores de índices, os operadores aritméticos pontuais, e funções como `max` , `min` , `sort` , etc., tornam esta tarefa fácil, e os programas bem mais curtos e legíveis.

A.5 Scripts e Funções

O fluxo do programa também é desviado pela invocação de subrotinas. A subrotina de nome *nomsubr* deve estar guardada no arquivo `nomsubr.m` ; por isto subrotinas são também denominadas M-arquivos (M-files). Há dois tipos de subrotinas: Scripts e Funções.

Um script é simplesmente uma seqüência de comandos, que serão executados como se fossem digitados ao prompt do matlab. Subrotinas podem invocar outras subrotinas, inclusive recursivamente.

Um M-arquivo de função em Matlab começa com a declaração da forma
`[ps1, ps2,... psm] = nomefunc( pe1, pe2,... pen )`
A lista entre parênteses é a lista de parâmetros de entrada da função, e a lista entre colchetes são os parâmetros de saída. Parâmetros são variáveis locais, assim como são variáveis locais todas as variáveis no corpo da função.

Ao invocarmos a função *nomefunc* com o comando
`[as1,... asm] = nomefunc(ae1,... aen);`
Os argumentos de entrada, $ae1, \ldots aen$, são passados por valor aos (i.e. copiados nos) parâmetros de entrada, e ao fim do M-arquivo, ou ao encontrar o comando `return` , os parâmetros de saída são passados aos argumentos de saída. Exemplos:

```
function [mabel, mabin] = vmax(v)
% procura o maior elemento, em valor absoluto,
% dentro de um vetor, linha ou coluna.
%Obs: Esta funcao NAO segue os conselhos
%para operar sobre blocos, e nao elementos!

[m,n]=size(v);
if( m ~= 1  &  n ~= 1 )
  erro;
else
  K=max([m,n]);
  mabel= abs(v(1)); mabin= 1;
  for k=2:K
    if( abs(v(k)) > mabel )
      mabel= abs(v(k)); mabin= k;
    end%if
  end%for
end%else
```

Para nos referir, dentro de uma função, a uma variável externa, esta deve ter sido declarada uma variável global com o comando `global`. A forma de especificar uma variável como global muda um pouco de interpretador para interpretador, e mesmo de versão para versão: Diga `help global` para saber os detalhes de como funciona a sua implementação.

Appendix B

Manual do Critical-Point for Windows

V. 3.00 01-12-1998

Critical-Point for Windows é um programa integrado para resolver e analisar modelos de otimização de portfólios de tipo média-variância (modelo de Markowitz). O programa é distribuido nas versões estudantil e profissional. As versões são praticamente idênticas, tanto na forma como na apresentação da análise. A versão profissional pode resolver modelos de grande porte, sendo orientada para uso comercial. Já a versão estudantil permite a análise de modelos com até 12 ativos, 4 opções e 20 restrições, mais ativos sem risco. A versão estudantil deve ser utilizada exclusivamente para pesquisa acadêmica e atividades sem fim lucrativo, estando disponível para estes propósitos. Graças ao apoio da BM&F, Bolsa de Mercadorias e de Futuros de São Paulo, e ao NOPEF-USP - Núcleo de Otimização e Processos Estocásticos Aplicados a Economia e Finanças da Universidade de São Paulo. Os direitos autorais das duas versões pertencem ao autor do programa, Julio M. Stern, que pode ser contactado em *jstern@ime.usp.br* ou *www.ime.usp.br/ jstern*

B.1 Instalação

A configuração mínima exigida pela versão estudantil é um micro IBM-PC Pentium com 16MB de RAM, 3MB livres no disco rígido, e MS-Windows 95.

Para iniciar a instalação siga os seguintes passos:

1. Coloque o disco de instalação no drive A ;
2. Execute (do Windows) A:\SETUP .

O Critical-Point versão estudantil será instalado no sub-diretório C:\SCRIPO, e a versão profissional no diretório C:\CRIPO. Não altere o nome e a localização destes diretórios no processo de instalação.

Observação: Se o programa de instalação abortar com uma mensagem de erro a respeito de um arquivo *.DLL ou *.OCX, saia do Windows, apague este arquivo do diretório C:\WINDOWS\SYSTEM e recomece a instalação.

Abertura

Para abrir o aplicativo Critical-Point:

1. Abra o Windows e duplo-clique o ícone do Critical-Point;
2. Digite o Password: "student". Use apenas letras minúsculas e sem as aspas;
3. A barra de menu do Critical-Point aparecerá no canto superior esquerdo da tela.

B.2 Geração da Fronteira Eficiente

Para traçar a Fronteira Eficiente de Markowitz, é necessário clicar, em sequência, os botões dos programas:

1. Filtro (F),
2. incluir Derivativos (D),
3. Compilador MDL (M),
4. Programação Quadrática Paramétrica (Q)

na barra de botões. Alternativamente, pode-se executar estes programas através do menu Run. Esta sequência resolve o modelo especificado nos arquivos MODEL.CP e MODPS.CP como será apresentado adiante.

Visualização da Fronteira Eficiente

Clique o botão Graph (G). O gráfico contendo a fronteira eficiente aparecerá. Os pontos críticos, utilizados para o cálculo da fronteira são identificados pelo símbolo "+". Cada Ponto Crítico é caracterizado por alguma mudança qualitativa no portfólio ótimo como um ativo que entrou ou saiu do portfólio, ou uma restrição que passou de ativa (justa) a inativa(folgaada), ou vice-versa.

Os títulos, cores, formato numérico, visualização da reta tangentes e muitas outras opções poderão ser modificadas através do menu Options Graph. As novas opções do usuário serão salvas automaticamente. Utilize o comando Options Reset Graph para retornar aos valores default.

Numeric Template

Esta é a janela que fornece as informações numéricas sobre a Fronteira Eficiente. Para chamá-la, acione o botão Numeric Template (N). A janela do Numeric Template é composta de seis linhas e oito colunas mais os botões Recalculate, Plot, Report e Save. As linhas são denominadas Seleção A até Seleção F e podem ser utilizadas para analisar até seis portfólios eficientes simultaneamente.

A primeira coluna apresenta valor de η, o parâmetro de ponderação entre o retorno esperado e o risco do portfólio. Trata-se de um parâmetro de propensão ao risco (em oposição à aversão ao risco). Quanto maior η, maior a importância do retorno esperado do portfólio em relação ao risco do portfólio.

A segunda coluna apresenta o retorno esperado do portfólio (e).

A terceira exibe a variância do portfólio (v).

A quarta coluna mostra o desvio-padrão (s) do portfólio.

A quinta coluna mostra o valor da taxa de retorno sem risco, r. Esta é a taxa de juros para o qual um determinado portfólio é tangente. Alternativamente, dada uma taxa de juros, o Critical-Point calcula qual o portfólio tangente correspondente.

A sexta e sétimas colunas mostram os parâmetros k e l. O parâmetro k indica que o portfólio situa-se entre os pontos críticos k e $k+1$. O parâmetro l fornece a posição relativa entre estes dois extremos, de tal forma que $l = 0$ significa que o portfólio é o ponto crítico k, $l = 0.5$ significa que o o portfólio está a meio caminho entre k e $k+1$, e $l = 1$ significa que o portfólio é o ponto crítico $k+1$.

A oitava coluna indica o status da seleção. O símbolo:

- $\emptyset$ mostra que a seleção ainda não foi recalculada;
- $+$ significa que está em cima de um ponto crítico;
- $\surd$ significa que a seleção não é um ponto crítico;
- $\uparrow\downarrow$ indicam que o parâmetro de entrada está muito alto ou muito baixo.

Podemos selecionar um portfólio digitando o valor de qualquer um destes parâmetros: acione o mouse sobre o campo de entrada e digite o valor do parâmetro (este campo

aparecerá em azul e sublinhado). Em seguida acione o botão Recalculate e todos os demais parâmetros (que aparecerão em preto) serão recalculados consistentemente com o parâmetro digitado. Sempre que uma seleção é recalculada, uma janela auxiliar apresentará a composição do portfólio selecionado.

Os parâmetro e e s podem ser selecionados diretamente na janela Graph: use o cursor do mouse para apontar para uma posição selecionada na fronteira eficiente; em seguida clique o botão da esquerda para copiar o parâmetro do retorno esperado no campo apropriado da linha selecionada (utilize o botão da direita para copiar a coordenada s).

O botão Plot traça a Fronteira Eficiente além de exibir os pontos selecionados (indicados por pequenos círculos sobre a curva), a reta tangente à fronteira eficiente, etc. Quando um portfólio é selecionado por sua taxa de juros (r), também a linha tangente é desenhada.

O botão Report gera uma descrição de todos os portfólios selecionados num arquivo ASCII chamado REPORT.CP. Este arquivo pode ser lido e impresso através do MS-DOS Editor que é acionado automaticamente através do menu Open File na barra de botões. O editor default pode ser modificado nas opções do menu. O botão Save grava os resultados antes da saída do programa.

O formato numérico do Numeric Template pode ser alterado através do menu Options Numeric. Para voltar ao formato original, use o comando Options Reset Numeric.

Impressão

Acione File Print Graph no menu ou clique o último botão da barra de botões para a impressão da Fronteira Eficiente. A impressora e os defaults de impressão usados serão aqueles definidos para o ambiente Windows. O comando File Print Setup do menu permite a reconfiguração da impressora. Através do comando Options To Clipboard, pode-se enviar o gráfico para o Clipboard do Windows no formato Bitmap ou Metafile.

B.3 Descrição do modelo e linguagem MDL

Os arquivos DERIV.CP e MODEL.CP contém as informações que o usuário deve fornecer ao Critical-Point. Estes arquivos ASCII podem ser editado pelo editor MS-DOS Editor ou qualquer outro editor sem formatação. O arquivo MODEL.CP contém os nomes e informações a respeito de cada ativo que podem compor o portfólio, bem como restrições que o portfólio deve obedecer. Um exemplo é o seguinte:

```
all  = {TEL4,ELE6,PET4,BB4,BBD4,SCO4,CEV4,BRH4};
```

```
#sectors
energy  = {ELE6,PET4};
bank = {BB4,BBD4};
food = {SCO4,CEV4,BRH4};
state = {TEL4,ELE6,PET4,BB4};
private = ~state;

# constraints
normal: sum[all] $ ==1;
statelim: sum[state] $ <= 0.5;
statebanks= state & banks;
statebklim:  sum[statebanks] $ <= 0.1;
foodsbound:  sum[food] $ >= 0.2;

liquindex[all] = {1.0@TEL4,0.6@ELE6,0.5@PET4,0.4@BB4,
                  0.4@BBD4,0.2@SCO4,0.2@CEV4,0.3@BRH4};
liqconstr: for[all] $ <= 0.5*liquindex;
```

A primeira linha define o conjunto universo, isto é, quais são os ativos que podem compor o portfólio.

A quarta linha define o subconjunto ou setor de energia e as linhas seguintes definem outros setores (bancário,alimentação, estatais, privadas).

A primeira restrição (normal) estabelece que a soma das participações de todos os ativos no portfólio é igual a 1. Esta é a restrição normal que tem de estar sempre presente, indicando o total a ser investido no portfólio. Este total pode ser um número de unidades monetárias, mas geralmente é tomado igual a 1 ou 100 de modo que, as quantidades a serem investidas em cada ativo representem frações da unidade ou percentuais do investimento total. O símbolo "==" é o símbolo de igualdade, não devendo ser confundido com "=", o sinal de atribuição.

A segunda restrição determina que a soma dos investimentos nos ativos estatais (definido como `{TEL4,ELE6,PET4,BB4}`) deve ser menor ou igual a 0.5

A linha que segue a segunda restrição define o conjunto de bancos estatais, `statebanks`, contendo os ativos que pertencem aos conjuntos de bancos e de estatais. A terceira restrição estabelece que a soma dos investimentos nos bancos estatais deve ser menor ou igual a 0.1.

A restrição em alimentação, `foodsbound`, determina que a soma dos investimentos neste setor deve exceder 0.2.

A última restrição estabelece que o investimento em cada ativo do universo nã deve exceder ao índice de liquides definido no vetor `liquindex`.

Retornos Esperados e Risco

O arquivo MODPS.CP contém informações sobre o retorno esperado de cada ativo, ao fim do período em consideração no modelo, bem como o desvio padrão (ou a volaatilidade, uma medida de risco) associado a este prognóstico.

```
all={
TEL4,ELE6,PET4,BB4,BBD4,SCO4,CEV4,BRH4};

er[all]={
4.521883e-003@TEL4, 9.349340e-002@ELE6, 1.414101e-001@PET4,
4.441184e-002@BB4, 4.125617e-002@BBD4, 2.153917e-002@SCO4,
-1.467467e-001@CEV4, 7.254108e-002@BRH4 };

std[all]={
2.105123e-001@TEL4, 3.214724e-001@ELE6, 2.988641e-001@PET4,
2.952717e-001@BB4, 2.019181e-001@BBD4, 1.709987e-001@SCO4,
2.471665e-001@CEV4, 1.866201e-001@BRH4 };

Hurst=0.50;
extrap=30;
sample=90;
```

O vetor er contém os retornos esperados, e o vetor std o desvio padrão destes retornos. Dizemos que esperamos do ativo asseta um retorno de er[asseta], "mais ou menos" std[asseta].

Ao escrever um número decimal, não ESQUEÇA O ZERO ANTES DO PONTO DECIMAL.

A Linguagem de Descrição do Modelo

Vejamos em maior detalhe a linguagem MDL ou Model Description Language. Esta linguagem tem por objetivo descrever, fácil e intuitivamente, as restrições que o portfólio deve obedecer.

Nomes

Um nome válido é uma sequência de letras, dígitos, sublinhas, e pontos, começando com uma letra, e tendo no máximo 15 caracteres. Não é permitido utilizar o mesmo nome

para dois ou mais objetos em um programa na linguagem MDL. Também não é permitido utilizar como nomes palavras reservadas da linguagem. As palavras reservadas são: abs, exp, for, ln, max, min, maxe, mine, print, sign, sum.

Índices

Os modelos tipo Markowitz constroem portfólios ótimos a partir de ativos dentro de um dado universo. Este conjunto de todos os ativos em consideração, também denominado conjunto universo, é o conjunto *all*, ou o conjunto de índices do modelo. A definição do conjunto universo deve ser a primeira linha em um programa MDL.

Conjuntos

Um conjunto é definido por uma atribuição constituida por:

1. O nome do conjunto
2. O símbolo de atribuição "=".
3. Uma lista de nomes de ativos
4. O terminador de sentença ";"

Uma lista de nomes é constituida de:

1. O símbolo de começo de lista, "{",
2. Uma sequência de nomes de ativos, separados por vírgulas.
3. O símbolo de fim de lista, "}".

Além de definir conjuntos pela lista de seus elementos, podemos ter conjuntos definidos por operações entre outros conjuntos. Estas operações são:

~ O complemento de um dado conjunto, é o conjunto de todos os elementos pertencentes a all que não estão presentes no conjunto.

| A união de dois conjuntos é o conjunto dos elementos no primeiro, no segundo, ou em ambos os conjuntos.

& A intersecção de dois conjuntos é o conjunto dos elementos que pertencem tanto ao primeiro quanto ao segundo conjunto.

\ A diferença entre dois conjuntos é constituída pelos elementos que pertencem ao primeiro mas não ao segundo conjunto.

Nos exemplos seguintes definimos alguns conjuntos novos através de conjuntos pré-existentes, por exemplo:

```
private = ~state;
statebanks = state & banks;
foodandenergy = food | energy;
```

A precedência destes operadores é a mesma da aritmética comum: em uma dada expressão i.e. primeiro fazemos todas as operações de complemento, depois as operações de intersecção, e finalmente as uniões e diferenças. A precedência de cálculo pode ser modificada utilizando parênteses como sinal de pontuação.

Escalares

Um escalar é simplesmente um número inteiro ou real. Números inteiros são formados por uma sequência de um ou mais dígitos decimais: 1,2,...,9,0. Um número inteiro pode ainda ser precedido de um sinal, + ou -, sem espaço entre o sinal e a sequência de dígitos. Assim são exemplos de números inteiros: 0; -998; 754.

Um número real pode ser representado em dois formatos:

Notação de ponto flutuante

Notação de ponto fixo

1. Notação de ponto fixo:

   Um número escrito nesta forma é uma sequência de: um inteiro, o ponto decimal e um inteiro sem sinal.

   Note que, em ponto fixo, é necessário escrever ao menos um dígito, possivelmente o zero, tanto antes como depois do ponto. Note também que o separador decimal é o ponto, e não a vírgula ou qualquer outro símbolo. Assim, são exemplos corretamente grafados 0.0, 37.65; -8.83, mas são **incorretos** os exemplos 10. ; .15 e -.88.

2. Notação de ponto flutuante (científica):

   (a) Um número inteiro ou um real em ponto fixo. Esta é a mantissa do número em notação científica.

   (b) A letra E, maiúscula ou minúscula.

(c) Um número inteiro. Este é o expoente do número em notação científica.

O expoente do número em notação científica indica de quantos dígitos, para a direita ou para a esquerda, devemos deslocar o ponto na mantissa. Esta notação é útil para escrever números muito grandes ou muito pequenos, pois evita longas seqüências de zeros antes ou depois do ponto. Assim, podemos representar 1000000 por 1E6, -1000000 por -1E6, 0.000001 por -1E-6 e -0.000001 por -1E-6.

Vetores Associativos

Vetores associam um valor numérico a cada ativo. Atribuições a vetores são sempre restritas ao conjunto de domínio da definição que é indicado no lado esquerdo da atribuição. Por exemplo: usando o vetor já definido anteriormente, temos:

```
std[all]={
2.105123e-001@TEL4, 3.214724e-001@ELE6, 2.988641e-001@PET4,
2.952717e-001@BB4, 2.019181e-001@BBD4, 1.709987e-001@SCO4,
2.471665e-001@CEV4, 1.866201e-001@BRH4 };
mincapt[private] = { 0.05@CEV4, 0.03@SCO4, 0.02@BBD4 };
maxmin[private] = max( minpart , mincapt );
```

A expressão <valor>@<ativo> associa o valor do vetor ao ativo.

As seguintes regras se aplicam a atribuições:

1- Uma atribuição não muda os valores de um vetor fora dos limites do domínio da atribuição.

2- Quando o domínio é omitido, é tomado como o universo, all.

3- O valor inicial de um vetor em todos os ativos é zero.

Podemos também atribuir valores a vetores através operações aritméticas (adição, subtração, produto, divisão e exponenciação entre dois vetores. As operações são calculadas para cada posição, isto é, elemento a elemento. Por exemplo, após a atribuição

```
example[private] = std*er;
```

os valores dos elementos do vetor example nas posições correspondentes às empresas privadas é `std*er;` e os valores em outros elementos permanecem inalterados ou nulos (caso seja a primeira atribuição ao vetor).

Podemos também realizar operações aritméticas entre escalares e vetores. Neste caso o escalar é interpretado como um vetor constante.

Exemplo:

```
er3[a]=er*3    # cada elemento de p que pertence ao subconjunto a
    # eh multiplicado por tres e armazenado em p3
```

Finalmente, podemos acessar e modificar o valor de um vetor em um particular ativo. Por exemplo: `std[CEV4]=0.55;`

Funções Escalares

As funções logarítmo, ln(), exponencial, exp(), valor absoluto, abs(), sinal, sign(), maior elemento maxe(), menor elemento mine() e soma interna sum() podem ser aplicadas a escalares e vetores. Assim como expressões aritméticas, estas funções são aplicadas a cada elemento de um vetor. Por exemplo:

```
lner[a] = ln(er);
```

As funções máximo e mínimo podem ser calculadas em pares de escalares ou vetores. Estas funções também são calculadas pontualmente quando assumem argumentos que são vetores. Por exemplo:

```
a1={1@AA, 2@BB, 3@DD};    # zero@CC is implicit
b1={5@BB, 1@DD};
c1=max(a1, b1);

# c1  is  {1@AA, 5@BB, 0@CC, 3@DD};
```

Restrições

As restrições do problema definem as condições a serem satisfeitas pelo portfólio. A primeira restrição, imprescindível, é a condição de normalização:

```
normal: sum[all] $ == 1;
```

Ela indica a quantidade total a ser investida no portfólio, que em geral é tomada como 1 ou 100, de forma que a composição de portfólio selecionada aparece como frações da unidade ou por fração percentual. São as variáveis de decisão do modelo, representando

o valor investido no i-ésimo ativo do portfólio. Equações de igualdade são indicadas pelo sinal "==", e inequações de desigualdade são indicadas pelos sinais "<=" e ">=".

Observação importante:

1. Não confunda a atribuição "=" com o sinal de igualdade "==".
2. As restrições de igualdade devem ser definidas antes das desigualdades.

Podemos definir restrições em termos de somas ponderadas dos investimentos feitos em um conjunto de ativos, como em `foodsbound : sum[food] $ <= 0.2;` estabelecendo que a soma dos investimentos no setor de alimentação não deve ultrapassar 20 por cento.

Podemos, da mesma forma, estabelecer conjuntos de restrições como:

```
ineqmin: $[TEL4]>=0.3;
upperbound: for[all] $ <=0.4;
ineqmax: for[private] $ <= 0.1*x;
```

estabelecendo que o investimento em `TEL4` deve ser de pelo menos 30%, que cada ativo individualmente não deve exceder 40% do portfólio e que o investimento em cada ativo do setor privado não deve ultrapassar 10% do valor que o vetor "x" associa àquele banco.

B.4 Modelo de Markowitz

As variáveis de decisão em um modelo do tipo Markowitz são interpretadas como as participações que os diversos ativos têm no portfólio. O objetivo é otimizar a função utilidade sobre o retorno esperado do portfólio, comparado à variância deste retorno. As principais entradas deste modelo são os retornos esperados, em uma matriz p, e a correspondente matriz de covariância S. O total investido em cada ativo, digamos, x, deve ser sempre positivo. Estas são as restrições de sinal. Outras restrições de igualdade e de desigualdade podem ser adicionadas na forma

$$Te * x == te$$

e

$$Tl * x <= tl.$$

A forma geral do problema de otimização é a seguinte:

$$max \;\; U(x) = \eta x' * p - x' * S * x$$

onde

x é o vetor de participações dos ativos no portfólio (a ser encontrado),

er=E(retorno) é o vetor de taxas de retorno esperadas dos ativos

S=Cov(retorno) é a matriz de covariância dos retornos dos ativos

Te é a matriz de coeficientes das restrições de igualdade

te é o vetor de termos independentes das restrições de igualdade

Tl é a matriz de coeficientes das restrições de desigualdade

tl é o vetor de termos independentes das restrições de desigualdade

η (lê-se eta) é o parâmetro de ponderação entre a utilidade da esperança do retorno do portfólio versus a aversão ao risco no investimento: quanto maior η, maior a propensão do investidor ao risco para valores de retorno esperado elevados. Valores reduzidos de η quantificam uma alta aversão ao risco por parte do investidor.

Trata-se portanto de um problema de Otimização Quadrática Paramétrica. A solução, para um dado valor de η é o vetor viável x (i.e. um vetor x satisfazendo todas as restrições do modelo) que otimiza a função utilidade. Logo, x@asseta é o total investido em asseta no portfólio x. Uma vez que para cada valor de eta temos uma solução ótima diferente, devemos escrevê-la como x(η).

Cada uma destas soluções ótimas, ou eficientes, representa um portfólio cuja esperança de retorno é e(η)=x(η)' * p, com variância v(η)=x(η)' * S * x(η), e desvio padrão s(η)= $\sqrt{v(\eta)}$. A curva e(η) versus s(η) é a Fronteira Eficiente do modelo.

A Fronteira Eficiente é formada por segmentos de parábolas que se unem formando uma curva côncava. Ela representa a fronteira superior do conjunto de portfólios viáveis. Para calculá-la é necessário primeiro determinar os pontos críticos. Cada ponto crítico é um ponto na curva correspondente a um η crítico.

Os etas críticos são aqueles em que ocorre alguma mudança qualitativa no portfólio x(η), i.e.,

1. um ativo entra ou sai do portfólio, ou
2. uma restrição de desigualdade passa de uma desigualdade estrita (folgada ou inativa) a uma igualdade (desigualdade justa ou ativa), ou vice-versa.

Para referir-se à posição de um dado portfólio, é usual a frase "o portfólio de retorno *e* mais ou menos *s*" para indicar o portfólio de retorno esperado *e* e desvio padrão *s*.

Exemplo de Aplicação

Tomemos um exemplo do modelo da seção que explica o arquivo MODEL.CP, com retorno esperado e desvio padrão obtidos através dos parâmetros padrão do filtro.

Utilizando o Numerical Template para selecionar alguns pontos da fronteira eficiente podemos obter:

```
Selected Portfolios: Parameters, Assets and Composition

Parameters of portfolio A
eta        esp      var         std         rate      k     l
4,22E-01  9,39E-02  4,28E-02  2,07E-01  -7,51E-03    1     0,00E+00

Assets and Composition of portfolio A
5,00E-01@PET4  2,00E-02@BBD4  3,00E-02@SCO4   5,00E-02@CEV4
4,00E-01@BRH4

Parameters of portfolio B
eta      esp       var       std      rate      k        l
3,21E-01   8,72E-02    3,78E-02   1,95E-01   -3,05E-02   2    0,00E+00

Assets and Composition of portfolio B
4,03E-01@PET4  2,00E-02@BBD4  3,00E-02@SCO4  5,00E-02@CEV4  4,97E-01@BRH4
```

e assim por diante.

B.5 Filtro

O modelo exige, como insumo fornecido pelo usuário, a matriz de correlação dos retornos esperados dos ativos, o vetor de retornos esperados e o vetor de desvios padrão. As estimativas destes parâmetros podem ser feitas com base nos dados históricos de cada ativo, ou por qualquer outro meio que o usuário julgar conveniente. Uma ferramenta simples e rubusta para este propósito, i.e. um filtro, é distribuido junto com o programa.

O filtro opera sobre longas séries históricas de preços dos ativos sendo considerados. O disquete de instalação contém alguns exemplos de séries históricas de alguns dos ativos do mercado brasileiro. Históricos atualizados diariamente podem ser comprados de firmas divulgadoras de dados financeiros.

Os parâmetros do filtros são ajustados no menu Options Filter. São eles:

Final date	data da última cotação a ser considerada
Interval	intervalo entre cotações a serem consideradas
Samples	Número de cotações a serem consideradas
Extrap	Número de intervalos, adiante da data final, para que se projetam retorno esperado, desvio padrão e covariâncias
Deflator	Nome do arquivo contendo as datas de cotação
Extension	Extensão dos arquivos de cotações
Hurst	Parâmetro característico do movimento Browniano (entre 0.5 e 1)

O filtro (F) calcula o retorno médio e o desvio padrão para cada ativo na série histórica. As amostras dos preços são tomadas no domínio do tempo T, especificado pelos parâmetros do filtro:

t in T = [data-final, datafinal-intervalo, datafinal -2intervalos, ..., datafinal-nro-de-amostras*intervalo]

Os filtro estima retornos logarítmicos (ou contínuos). Denotando por $price(i,t)$ e $r(i,t)$ o preço e o retorno do ativo i no período t, o filtro calcula:

$$r(i,t) = ln\left(\frac{price(i,t+1)}{price(i,t)}\right).$$

A seguir é calculado o retorno médio de cada ativo meanl(i) $= \frac{1}{samples}\sum_t r(i,t)$ e a matriz de covariância, desvios padrão e matriz de correlação das amostras:

$$Sl(i,j) = \frac{1}{samples}\sum_t (rl(i,t) - meanl(i)) * (rl(j,t) - meanl(j))$$

$$sl(i) = \sqrt{Sl(i,i)}$$

$$corrl(i,j) = \frac{Sl(i,j)}{sl(i)*sl(j)}$$

Em seguida o filtro extrapola o retorno esperado e o desvio padrão por extrap intervalos adiante:

$$erl(i) = extrap * meanl \ ; \quad std(i) = extrap^{Hurst} * sl(i);$$

Finalmente o filtro convertetodas as medidas logarítmicas de retorno e risco para medidas simples (de retorno e risco). Para preços deterministicos em $t = 0$ e $t = T$, o retorno logarítmico é $rl = ln(S(T)/S(0))$, e o retorno simples é $r = (S(T) - S(0))/S(0)$, portanto $r = exp(rl) - 1$. O filtro assume que os preços tem uma distribuição log-normal, de modo que as regras de transformação fical algo mais complicadas:

$$r = exp(erl + 0.5stdl^2) - 1 \ ; \quad s^2 = exp(2r + sl^2)(exp(sl^2) - 1).$$

Finalmente o filtro salva o retorno médio e o desvio padrão extrapolados como vetores associativos no arquivo FILTER.CP a matriz de correlação no arquivo matricial CORRELF.M .

O conjunto universo, i.e. o conjunto all, utilizado pelo filtro é o conjunto original que também aparece no arquivo FILTER.CP. No arquivo MODEL.CP podemos restringir o universo original eliminando alguns de seus elementos. Quando restringimos o universo original, o compilador MDL extrai automaticamente de er, std e correl, apenas as submatrizes e subvetores de que necessita. Os usuários da versão profissional do Critical Point podem receber através de modem atualizações destas matrizes e vetores, que são geradas para diversos ativos brasileiros. Os filtros utilizados para calculá-los são mais sofisticados que os filtros utilizados pelo Critical Point. Estes filtros incorporam tecnologias avançadas como análise fractal, filtros adaptativos, tratamentos especiais para valores faltantes etc.

B.6 Derivativos

A ferramenta para derivativos expande os vetores de retorno esperado e desvio padrão, bem como a matriz de correlação, para incluir os ativos derivativos sobre ativos do conjunto fundamental.

```
put={OTP19@TEL4, OTP24@TEL4 };
call={OTC16@TEL4, OTC17@TEL4 };

exdays={40@OTP19, 30@OTP24, 40@OTC16, 40@OTC17};

O={6.7@OTP19, 10.9@OTP24,10.9@OTC16, 6.7@OTC17};
S={63.2@OTP19, 63.2@OTP24, 63.2@OTC16, 63.2@OTC17};
K={64@OTP19, 72@OTP24, 56@OTC16, 68@OTC17};
```

No arquivo exemplo DERIV.CP acima, temos o conjunto put declarando as opções de venda (europeias) OTP19 e OTP24 sobre o ativo fundamental TEL4; da mesma forma, declaramos opções de compra OTC16 e OTC17. Para definir opções temos quew especificar o preço de contrato de opção (O), o preço do ativo fundamental (S), o preço de exercício (K), e o número de dias até o exercício (exdays).

Covariância de Derivativos

A ferramenta para derivativos escreve o conjunto universo extendido (all), com o nome de todos os ativos (fundamentais e derivativos), bem como seus retornos esperados e desvios padrão, no arquivo ERSX.CP, e a matriz de correlação extendida em CORRELX.M.

Como exemplo simples, a expressão para retorno de em call é:

$$rc = (max(S(t1) - K(t0), 0) - O(t0))/O(t0)$$

Os retornos experados, desvios padrão e correlações para derivativos são calculados asssumindo distribuição log-normal, usando as expressões desenvolvidas no capítulo 4 e técnicas avançadas de integração numérica.

B.7 Arquivos de Dados e Matriciais

O compilador MDL compila, i.e. traduz, o modelo descrito na linguagem MDL para a forma matricial. O problema descrito desta forma, no arquivo MARKIN.M é entrada para o solver do Programa de Programação Paramétrica (Q). Para construir modelos mais sofisticados devemos entender a estrutura dos arquivos matriciais.

A ordem na qual os ativos aparecem no vetor ou nas matrizes é aquela na qual eles aparecem no universo original, segundo o arquivo MODPS.CP. O filtro tem um procedimento para computar a matriz de correlação. Cada elemento da matriz de correlação $Cor(i, j)$ pode ser interpretado como uma medida da interferência mútua entre o retorno do i-ésimo e do j-ésimo ativos. Por exemplo, se os dois ativos têm retornos proporcionais $p_i = a*p_j$, $a > 0$ então $Cor(i, j) = 1$. Se, por outro lado, a constante de proporcionalidade for negativa, $a < 0$ (isto é o preço do ativo i crescer implica que o preço j decresce) então $Cor(i, j) = -1$. Finalmente se ambos os retornos são mutuamente independentes, $Cor(i, j) = 0$.

Um usuário mais sofisticado pode construir sua própria matriz de correlação. O Critical Point permite manipulação de arquivos matriciais, entretanto, a utilização de uma matriz que não satisfaça todas as propriedades matemáticas da matriz de correlação pode causar problemas numéricos no procedimento de programação quadrática paramétrica (Q). Dentre as propriedades da matriz de correlação temos:

1- Dominância diagonal	cor(i,i) ==1; abs(cor(i,j)) $<$ 1, i $\neq$ j;
2- Simetria:	cor(i,j) == cor(j,i);
3- Positividade	min(eig(cor)) $\geq$ 0

Retornos Históricos versus Análise de Mercado

Após a leitura dos arquivos MODPS.CP e MODEL.CP, o compilador MDL lê o arquivo CORREL.M a fim de reconstruir a matriz de covariância.

$$S(i, j) = s(i)' * Corr(i, j) * s(j)$$

;

Finalmente o compilador MDL salva as restrições estabelecidas no MODEL.CP, bem como o vetor de retornos esperados e a matriz de covariância, no arquivo MARKIN.M, que será entrada do procedimento de programação quadrática paramétrica (Q).

O compilador MDL trata os vetores de retornos esperados e desvios padrão como qualquer vetor associativo de forma que ele também possa ser modificado utilizando os recursos da linguagem MDL. Para evitar inconsistências no cálculo dos retornos esperados e desvios padrão, estas modificações devem ser feitas no arquivo DERIV.CP. Cabe mencionar que analistas de mercado profissionais raramente utilizam médias extrapoladas como previsão de retorno. Os retornos médios e também os desvios padrão extrapolados pelo filtro, são utilizados somente para "calibrar" o modelo mas estes valores podem ser ajustados pelos analistas. Entre as mais populares técnicas de ajuste estão:

1. Análise fundamentalista
2. Predição através de análise gráfica
3. Previsão através de análise técnica
4. Modelos matemáticos de redes neurais, filtros adaptativos e árvores de classificação.

Estas técnicas fogem do escopo deste manual e são objeto de estudo de disciplinas específicas dentro do NOPEF-USP.

Arquivos de Dados

O filtro lê séries históricas de preços do diretório DATA. O arquivo dolof.ofc contém a taxa de conversão de Reais para Dólares americanos. Este arquivo também define o calendário oficial , isto é, os dias de negociação. Outros arquivos *.ofc têm os arquivos com séries de preços deflacionados por dólar para alguns ativos brasileiros. As três primeiras linhas de cada arquivo definem:

1. O nome do ativo
2. O deflator de preços
3. O número de ações

O terminador de série é o símbolo asterisco, *. O próximo exemplo mostra o arquivo TEL3.ofc, com os preços, em dólares, por 10000 ações da TELebrás, na BOVESPA, nos últimos 4 dias de 1994. As datas devem estar no formato dd/mm/yy e os preços em notação de ponto flutuante.

```
Asset: TEL3
Deflator: DOLOF.OFC
Shares: 1E+4
Date        Price
30/12/94    4.314E+02
29/12/94    4.304E+02
28/12/94    4.030E+02
27/12/94    4.025E+02
*        (T.S. terminator)

(End of File)
```

B.8 Restrições lógicas sobre conjuntos

A linguagem MDL permite restringir conjuntos através de expressões lógicas. Exemplo: `exampleset[energy|bank]=(arrbank>0.5)|(arrenergy<0.2);` após este comando, somente os ativos do setor bancário que também satisfazem a condição `arrbank>0.5` ou os ativos do setor de energia que satisfazem `arrenergy` < 0.2 permanecem no conjunto `exampleset`.

Condições lógicas podem ser construídas utilizando os operadores de comparação:

$>$	maior que
$>=$	maior ou igual
$<$	menor que
$<=$	menor ou igual
$==$	igual
$\tilde{}=$	diferente

Expressões mais complexas podem ser realizadas entre conjuntos e vetores. A representação de um conjunto nada mais é que um vetor associativo booleano (0 ou 1). Logo em atribuições a conjuntos devemos interpretar o nome de um ativo, asseta, como sendo equivalente à expressão 1@asseta. Também interpretamos operações lógicas entre conjuntos utilizando os operadores lógicos "não" ($\tilde{}$), "e", (&) e "ou" (|), como assumindo o valor 0 quando falso e qualquer valor não nulo quando verdadeiro.

Quando restringimos o conjunto universo, all, proibimos qualquer ativo eliminado de entrar no portfólio, e somente os subvetores e submatrizes necessários de p, std, e cov

serão escritos no arquivo MARKIN.M pelo compilador MDL. Para manter este arquivo consistente, devemos primeiramente restringir o universo em MODEL.CP, e só então estabelecer as restrições do modelo.

B.9 Depuração

Mesmo os mais experientes programadores comentem erros e, apesar de que muito poucos modelos requeiram todos os recursos da linguagem MDL, alguns deles tornam-se muito complexos. O compilador MDL possui algumas ferramentas simples para auxiliar a detecção de erros. Quando o compilador não consegue analisar uma frase, exibe uma mensagem de erro, contendo:

1. o nome do arquivo onde ocorreu o erro.
2. o número da linha onde este erro aconteceu.
3. a última palavra válida, a última lida e o código ASCII.

O comando print x; força o compilador MDL a escrever o objeto x (escalar, conjunto ou vetor) no arquivo PRINT.LOG, exatamente como este objeto se encontra no momento que o comando é acionado pelo programa MDL.

A calculadora de média variância

A calculadora de média variância (botão C) fornece uma maneira simples de calcular o desvio padrão e e retorno esperado de portfólios pré-definidos. Os portfólios são definidos no arquivo FOLIOS.CP, por exemplo:

```
papas={5.0E-1@PET4,2.0E-2@BBD4,3.0EE-2@SCO4,5.0E-2@CEV4,4.0E-0@BRH4}
```

A calculadora imprime o valor calculado no arquivo RFOLIOS.CP, por exemplo:

```
papas:          e = 9.39E-02        s = 2.07E-01
```

B.10 Short Selling e Tracking

Na descrição do modelo, antes da declaração de restrições, o subconjunto short pode ser declarado. O efeito de ter um ativo em short é trocar o sinal de seu retorno esperado, bem como o de sua correlação com outros ativos.

A interpretação financeira de ter um ativo em short é a de vender este ativo a descoberto no sentido de Linter: Uma venda a descoberto implica em um depósito colateral,

igual ao valor vendido. Assim a venda a descoberto não pode ser usada como uma fonte de recursos, mas é sim mais um uso de recurso, a ser incluido no portfólio.

Caso o investidor receba um juro pelo colateral, o vetor de retornos esperados deve ser ajustado (uma simples atribuição usando o conjunto short como domínio). No próximo exemplo uma taxa fixa de 1% é paga sobre o colateral.

```
all={ TEL4, ELE6, PET4, BB4, BBD4, SCO4, CEV4, BRH4, IBOVESPA };
short={TEL4,ELE6};
er[short]= er -0.01;           #adding 1% to  -er
normal: sum[all] $==1;
```

Em muitas aplicações queremos formar um portfólio para perseguir (track) um investimento de referência. A diferença de retorno entre o portfólio e sua referência é denominad erro de perseguição (tracking error). No próximo exemplo procuramos um portfólio para perseguir o índice IBOVESPA. O gráfico de média-variância obtido para este modelo não mais se refere a retornos absolutos, mas a retornos relativos ao investimento de referência (IBOVESPA).

```
all={ TEL4, ELE6, PET4, BB4, BBD4, SCO4, CEV4, BRH4, IBOVESPA };
short={IBOVESPA};
normal: sum[all] $==2;
track: for[short] $==1;
```

Autores do programa

Julio Michael Stern escreveu a versão original do solver (em Matlab) e especificou a linguagem MDL e a interface com o usuário. Jacob Zimbarg Sobrinho traduziu o solver para a linguagem C (Watcom 10.0) e escreveu o filtro. Fabio Nakano escreveu o compilador MDL (utilizando GNU lex e yacc) e a versão atual da interface com o usuário (em Visual Basic). Cibele Dunder escreveu a ferramenta para ativos derivativos. Por favor envie sugestões e comunique eventuais erros a jstern@ime.usp.br (internet).

Appendix C

Álgebra Linear Computacional

C.1 Notação e Operações Básicas

Este parágrafo define algumas notações matriciais. Indicamos por $(1:n)$ a lista $[1,\dots n]$, e $j \in (1:n)$ indica que o índice j está neste domínio. Uma lista de matrizes tem um (ou mais) índices superscritos, $S^1 \dots S^m$. Assim $S^k_{h,i}$ é o elemento na linha h e coluna i da matriz S^k. A matriz

$$A = \begin{bmatrix} A^{1,1} & \cdots & A_{1,s} \\ \vdots & \ddots & \vdots \\ A^{r,1} & \cdots & A_{r,s} \end{bmatrix}$$

é uma matriz blocada, onde $A^{p,q}$ é o $p-q$-ésimo bloco, ou sub-matriz.

Quando estamos falando de apenas uma matriz, X, costumamos esceve-la com o índice de linha subscrito, e o índice de coluna superscrito Assim x_i, x^j, e x^j_i são a linha i, a coluna j, e o elemento (i,j) da matriz X. ESta notação é mais compacta, e resalta o fato de vermos a matriz X como uma matriz blocada por vetores coluna. A matriz $X^{j:k}_{h:i}$ é um bloco extraido da matriz X, fazendo os índices de linha e coluna percorrer os domínios indicados. $\mathbf{0}$ e $\mathbf{1}$ são matrizes de zeros e uns, geralmente vetores coluna , $n \times 1$. Quando a dimensão não está indicada, ela pode ser deduzida do contexto. $V > 0$ é uma matriz positiva definida. Definimos a p-norma de um vetor x por $||x||_p = (\sum |x_i|^p)^{-p}$. Assim, se x para um vetor não negativo, podemos escrever sua 1-norma como $||x||_1 = \mathbf{1}'x$.

O produto de Kroneker de duas matrizes é uma matriz blocada onde o bloco (i,j) é a segunda matriz multiblicada pelo elemento $(1,j)$ da primeira matriz:

$$A \otimes B \ = \ \begin{bmatrix} A^1_1 B & A^2_1 B & \cdots \\ A^1_2 B & A^2_2 B & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

As seguintes propriedades podem ser facilmente verificadas:

- $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$
- $(A \otimes B)' = A' \otimes B'$
- $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$

O operador Vec "empilha" as colunas de uma matriz em um único vetor coluna, i.e., se A é $m \times n$,

$$\mathrm{Vec}(A) = \begin{bmatrix} A^1 \\ \vdots \\ A^n \end{bmatrix}$$

As seguintes propriedades podem ser facilmente verificadas:

- $\mathrm{Vec}(A + B) = \mathrm{Vec}(A) + \mathrm{Vec}(B)$
- $\mathrm{Vec}(AB) = \begin{bmatrix} AB^1 \\ \vdots \\ AB^n \end{bmatrix} = (I \otimes A)\,\mathrm{Vec}(B)$

C.2 Espaços Vetoriais com Produto Interno

Dados dois vetores $x, y \in \mathcal{R}^n$, o seu **produto escalar** é definido como

$$< x \mid y > \equiv x'y = \sum_{i=1}^{n} x_i y^i \ .$$

Com esta definição vê-se que o produto escalar é um operador que satisfaz as propriedades fundamentais de **produto interno**, a saber:

1. $< x \mid y > = < y \mid x >$, simetria.
2. $< \alpha x + \beta y \mid z > = \alpha < x \mid z > + \beta < y \mid z >$, linearidade.
3. $< x \mid x > \geq 0$, semi-positividade.
4. $< x \mid x > = 0 \Leftrightarrow x = 0$, positividade.

Através do produto interno, definimos a norma:

$$\|x\| \equiv < x \mid x >^{1/2} \ ;$$

e definimos também o ângulo entre dois vetores:

$$\Theta(x, y) \equiv \arccos(< x \mid y > / \|x\|\|y\|) \ .$$

C.3 Projetores

Consideremos o subespaço linear gerado pelas colunas de uma matriz A, m por n, $m \geq n$:

$$C(A) = \{y = Ax, x \in \mathcal{R}^n\} \ .$$

Denominamos $C(A)$ de imagem de A, e o complemento de $C(A)$, $N(A)$, de espaço nulo de A,

$$N(A) = \{y \mid A'y = 0\} \ .$$

Definimos a projeção de de um vetor $b \in \mathcal{R}^m$ no espaço das colunas de A, pelas relações:

$$y = P_{C(A)}b \leftrightarrow y \in C(A) \wedge (b - y) \perp C(A)$$

ou, equivalentemente,

$$y = P_{C(A)}b \leftrightarrow y = Ax \wedge A'(b - y) = 0 \ .$$

No que se segue suporemos que A tem posto pleno, i.e. que suas colunas são linearmente independentes. Provemos que o projetor de b em $C(A)$ é dado pela aplicação linear

$$P_A = A(A'A)^{-1}A' \ .$$

Se $y = A((A'A)^{-1}A'b)$, então obviamente $y \in C(A)$. Por outro lado,

$$A'(b - y) = A'(I - A(A'A)^{-1}A')b = (A' - IA')b = 0 \ .$$

C.4 Matrizes Ortogonais

Dizemos que uma matriz quadrada e real é **ortogonal** sse sua inversa é igual a sua transposta.Dada Q uma matriz ortogonal, suas colunas formam uma base ortonormal de $\mathcal{R}^n$, como pode ser visto da identidade $Q'Q = I$. A norma quadrática de um vetor v, ou seu quadrado

$$\|v\|^2 \equiv \sum (v_i)^2 = v'v$$

permanece inalterada por uma transformação ortogonal, pois

$$(Qv)'(Qv) = v'Q'Qv = v'Iv = v'v \ .$$

Dado um vetor em $\mathcal{R}^2$, $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, a rotação deste vetor por um ângulo θ é dada pela transformação linear

$$G(\theta)x = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} .$$

Notemos que uma rotação é uma transformação ortogonal, pois

$$G(\theta)'G(\theta) = \begin{bmatrix} \cos(\theta)^2 + \sin(\theta)^2 & 0 \\ 0 & \cos(\theta)^2 + \sin(\theta)^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} .$$

A rotação de Givens é um operador linear cuja matriz coincide com a identidade, exceto num par de linhas onde imergimos uma matriz de rotação bidimensional:

$$G(i,j,\theta) = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & \cos(\theta) & & \sin(\theta) & & \\ & & & \ddots & & & \\ & & -\sin(\theta) & & \cos(\theta) & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} .$$

Dizemos que a aplicação deste operador numa matriz A, $G'A$, roda as linhas i e j de A de um ângulo θ (no sentido ant-horário).

Como o produto de transformações ortogonais continua ortogonal (prove), podemos usar uma seqüência de rotações para construir transformações ortogonais.

Listamos agora algumas rotações em $\mathcal{R}^2$, construidas a partir de simples relações trigonométricas, que serão utilizadas como blocos de construção para diversos agoritmos.

Consideremos, em $\mathcal{R}^2$, v um vetor, S uma matriz simétrica, e A uma matriz assimétrica,

$$v = \begin{bmatrix} x \\ y \end{bmatrix} , \quad S = \begin{bmatrix} p & q \\ q & r \end{bmatrix} , \quad A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

Para anular a segunda componente do vetor v por uma rotação á esquerda, $G(\theta_v)'\, v$, basta tomar um ângulo

$$\theta_v = \arctan\left(\frac{y}{x}\right) .$$

Para diagonalizar a matriz simétrica por uma rotação simétrica, $G(\theta_{diag})'\, S\, G(\theta_{diag})$, basta tomar um ângulo

$$\theta_{diag} = \frac{1}{2}\arctan\left(\frac{2q}{r-p}\right) .$$

Para simetrizar a matriz assimétrica por uma por uma rotação à esquerda, $G(\theta_{sym})'\, A$, basta tomar um ângulo

$$\theta_{sym} = \arctan\left(\frac{b-c}{a+d}\right) .$$

Assim, para diagonalizar a matriz assimétrica basta simetriza-la, e em seguida diagonaliza-la. Alternativamente, podemos diagonalizar a matriz assimétrica por um par de "rotações de Jacobi", à esquerda e à direita, $J(\theta_r)'\, A\, J(\theta_l)$; bastando tomar os ângulos

$$\theta_{sum} = \theta_r + \theta_l = \arctan\left(\frac{c+b}{d-a}\right) \ , \quad \theta_{dif} = \theta_r - \theta_l = \arctan\left(\frac{c-b}{d+a}\right) \quad \text{ou}$$

$$J(\theta_r)' = G(\theta_{sum}/2)'\, G(-\theta_{dif}/2)' \ , \quad J(\theta_l) = G(\theta_{dif}/2)\, G(\theta_{dif}/2) \ .$$

No cálculo das rotações, as funções trigonométricas, Seno, Coseno e Arco-Tangente não são realmente utilizadas, já que nunca utilizamos os ângulos propriamente ditos, mas apenas $c = \sin(\theta)$ e $s = \sin\theta$, que podemos computar diretamente como

$$c = \frac{x}{\sqrt{x^2+y^2}} \ , \quad s = \frac{-y}{\sqrt{x^2+y^2}} \ .$$

Para prevenir overflow podemos utilizar o cálculo:

- Se $y == 0$, então $c = 1$, $s = 0$.
- Se $y \geq x$, então $t = -x/y$, $s = 1/\sqrt{1+t^2}$, $c = st$.
- Se $y < x$, então $t = -y/x$, $c = 1/\sqrt{1+t^2}$, $s = ct$.

C.5 Fatoração QR

Dada A uma matriz real de posto pleno $m \times n$, $m \geq n$, podemos sempre encontrar uma matriz ortogonal Q tal que $A = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$, onde R é uma matriz quadrada e triangular superior. Esta decomposição é dita uma fatoração "QR", ou fatoração ortogonal, da matriz A. O fator ortogonal $Q = [C \mid N]$ nos da uma base ortonormal de $\mathcal{R}^m$ onde as n primeiras colunas são uma base ortonormal de $C(A)$, e as $m-n$ últimas colunas são uma base de $N(A)$, como pode ser visto diretamente da identidade $Q'A = \begin{bmatrix} R \\ 0 \end{bmatrix}$. Construiremos a seguir um método para fatoração ortogonal.

Abaixo ilustramos uma seqüência de rotações de linhas necessárias que leva uma matriz 5×3 à forma triangular superior. Cada par de índices, (i, j), indica que rodamos estas linhas do ângulo apropriado para zerar a posição na linha i, coluna j. Supomos que inicialmente a matriz é densa, i.e. todos os seus elementos são diferentes de zero, e ilustramos o padrão de esparsidade da matriz nos estágios assinalados com um asterisco na seqüência de rotações.

$$(1,5) * (1,4)(1,3)(1,2) * (2,5)(2,4)(2,3) * (3,5)(3,4) *$$

$$\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \\ 0 & x & x \end{bmatrix} \quad \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix} \quad \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & x \\ 0 & 0 & x \end{bmatrix} \quad \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

C.5.1 Mínimos Quadrados

Dado um sistema superdeterminado, $Ax = b$ onde a matriz A $m \times n$ tem $m > n$, dizemos que x^* "resolve" o sistema no sentido dos mínimos quadrados, ou que x^* é a "solução" de mínimos quadrados, sse x^* minimiza a norma quadrática do resíduo,

$$x^* = Arg \min_{x \in \mathcal{R}^n} \|Ax - b\| \ ,$$

Dizemos também que $y = Ax^*$ é a melhor aproximação, no sentido dos mínimos quadrados de b em $C(A)$.

Como a multiplicação por uma matriz ortogonal deixa inalterada a norma quadrática de um vetor, podemos procurar a solução deste sistema (no sentido dos mínimos quadrados) minimizando a transformação ortogonal do resíduo usada na fatoração QR de A,

$$\|Q'(Ax - b)\|^2 = \| \begin{bmatrix} R \\ 0 \end{bmatrix} x - \begin{bmatrix} c \\ d \end{bmatrix} \|^2 = \|Rx - c\|^2 + \|0x - d\|^2.$$

Da última expressão vê-se que a solução, a aproximação e o resíduo do problema original são dados por, respectivamente

$$x^* = R^{-1}c \ , \quad y = Ax^* \ \text{ e } \ z = Q \begin{bmatrix} 0 \\ d \end{bmatrix}.$$

Como já havíamos observado, as $m - n$ últimas colunas de Q formam uma base ortonormal de $N(A)$, logo $z \perp C(A)$, de modo que concluímos que $y = P_A b$!

C.6 Fatorações LU e Cholesky

Dada uma Matriz A, a **operação elementar** determinada pelo **multiplicador** m_j^i, é subtrair da linha j a linha i multiplicada por m_j^i. A operação elementar aplicada a matriz

identidade gera a correspondente **matriz elementar**,

$$M(i,j) = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & & & i \\ & & \vdots & \ddots & & & \\ & & -m_j^i & & 1 & & j \\ & & \vdots & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \quad .$$

Aplicar uma operação elementar a matriz A equivale a multiplicá-la a esquerda pela correspondente matriz elementar.

Na **Fatoração de Gauss**, ou fatoração LU, usamos uma seqüência de operações elementares para levar A a forma triangular superior,

$$MA = M(n-1,n)\cdots M(2,3)\cdots M(2,n)M(1,1)\cdots M(1,n-1)M(1,n)A = U \ .$$

A inversa da produtória desta seqüência de matrizes elementares tem a forma triangular inferior (verifique)

$$L = M^{-1} = \begin{bmatrix} 1 & & & & \\ m_2^1 & 1 & & & \\ \vdots & \vdots & \ddots & & \\ m_{n-1}^1 & m_{n-1}^2 & & 1 & \\ m_n^1 & m_n^2 & \cdots & m_n^{n-1} & 1 \end{bmatrix} \quad .$$

No caso de fatorarmos uma matriz simétrica, $S = LU$, podemos por em evidencia os elementos diagonais de U obtendo $S = LDL'$. Se S for positiva definida assim será D, de modo que podemos escrever $D = D^{1/2}D^{1/2}$, $D^{1/2}$ a matriz diagonal contendo a raíz dos elementos em D. Definindo $C = LD^{1/2}$, temos $S = CC'$, a **fatoração de Choleski** de S.

C.6.1 Programação Quadrática

O problema de **programação quadrática** consiste em minimizar a função

$$f(y) \equiv (1/2)y'Wy + c'y \ , \quad W = W'$$

sujeitos às **restrições**

$$g_i(y) \equiv N_i'y = d_i.$$

Os gradientes de f e g_i são dados, respectivamente, por

$$\nabla_y f = y'W + c' \ , \text{ e } \nabla_y g_i = N_i' \ .$$

As **condições de otimalidade** de primeira ordem (condições de Lagrange) estabelecem que as restrições sejam obedecidas, e que o gradiente da função sendo minimizada seja uma combinação linear dos gradientes das restrições. Assim a solução pode ser obtida em função do **multiplicador de Lagrange**, i.e. do vetor l de coeficientes desta combinação linear, como

$$N'y = d \ \wedge \ y'W + c' = l'N' \ ,$$

ou em forma matricial,

$$\begin{bmatrix} N' & 0 \\ W & N \end{bmatrix} \begin{bmatrix} y \\ l \end{bmatrix} = \begin{bmatrix} d \\ c \end{bmatrix} \ .$$

Este sistema de equações é conhecido como o **sistema normal**. O sistema normal tem por matriz de coeficientes uma matriz simétrica. Se a forma quadrática W for **positiva definida**, i.e.se $\forall x \ x'Wx \geq 0 \ \wedge \ x'Wx = 0 \Leftrightarrow x = 0$, e as restrições N forem lineramente independentes, a matriz de coeficientes do sistema normal será também positiva definida.

C.7 Fatoração SVD

A fatoração SVD decompõem uma matriz real A, $m \times n$, $m \geq n$, em um produto $D = U'AV$, onde D é diagonal, e U, V são matrizes ortogonais. Consideremos primeiramente o caso $m = n$, i.e. uma matriz quadrada.

O algoritmo de Jacobi é um algoritmo iterativo que, a cada iteração, "concentra a matriz na diagonal", através de rotações de Jacobi.

$$J(i,j,\theta_r)' \, A^k \, J(i,j,\theta_l) = A^{k+1} = \begin{bmatrix} A^{k+1}_{1,1} & \cdots & A^{k+1}_{1,i} & \cdots & A^{k+1}_{1,j} & \cdots & A^{k+1}_{1,n} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ A^{k+1}_{i,1} & \cdots & A^{k+1}_{i,i} & \cdots & 0 & \cdots & A^{k+1}_{i,n} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ A^{k+1}_{j,1} & \cdots & 0 & \cdots & A^{k+1}_{j,j} & \cdots & A^{k+1}_{j,n} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ A^{k+1}_{n,1} & \cdots & A^{k+1}_{n,i} & \cdots & A^{k+1}_{n,j} & \cdots & A^{k+1}_{n,n} \end{bmatrix}$$

Consideremos a soma dos quadrados dos elementos fora da diagonal na matriz A $\mathrm{Off}_2(A)$. Vemos que

$$\mathrm{Off}_2(A^{k+1}) = \mathrm{Off}_2(A^k) - (A^k_{i,j})^2 - (A^k_{j,i})^2$$

Assim, escolhendo a cada iteração o par de índices que maximiza a soma dos quadrados do par fora da diagonal a ser anulado, temos um algoritmo que converge linearmente para uma matriz diagonal.

O algoritmo de Jacobi nos dá uma prova construtiva da existência da fatoração SVD, e é a báse para vários algoritmos mais eficientes de fatoração SVD.

Se A é uma matriz retangular, basta inicialmente fatorar $A = QR$, e aplicar o algoritmo de Jacobi ao bloco quadrado superior de R. Se A é quadrada e simétrica, a fatoração obtida é denominada decomposição de autovalores de A.

As matrizes U e V podem ser interpretadas como bases ortogonais dos respectivos espaços de dimensão m e n. Os valores na diagonal de S são denominados valores singulares da matriz A, e podem ser interpretados geométricamente como fatores multiplicadores do mapa $A = UDV'$, que leva cada versor da base V para um múltiplo de um versor da báse U.

C.8 Matrizes Complexas

A maioria das técnicas aqui desenvolvidas para matrizes reais, podem ser extendidas para matrizes complexas. Uma forma prática e elegante para tal são as transformções Q (TQ) de Hemkumar, aplicadas a uma dada matriz M, complexa 2×2, na forma de um par "interno" de transformações de fase unitárias, e um par "externo" de rotações,

$$\begin{bmatrix} c(\phi) & -s(\phi) \\ s(\phi) & c(\phi) \end{bmatrix} \begin{bmatrix} e(i\alpha) & 0 \\ 0 & e(i\beta) \end{bmatrix} \begin{bmatrix} Ae(ia) & Be(ib) \\ Ce(ic) & De(id) \end{bmatrix} \begin{bmatrix} e(i\gamma) & 0 \\ 0 & e(i\delta) \end{bmatrix} \begin{bmatrix} c(\psi) & -s(\psi) \\ s(\psi) & c(\psi) \end{bmatrix}$$

A elegância da TQ vem da seguinte observação: Enquanto a transformação interna afeta apenas os expoentes imaginários dos elementos da matriz, a transformação externa pode ser aplicada independentemente à parte real e à parte imaginária da matriz, i.e.

$$\begin{bmatrix} e(i\alpha) & 0 \\ 0 & e(i\beta) \end{bmatrix} \begin{bmatrix} Ae(ia) & Be(ib) \\ Ce(ic) & De(id) \end{bmatrix} \begin{bmatrix} e(i\gamma) & 0 \\ 0 & e(i\delta) \end{bmatrix} =$$

$$\begin{bmatrix} Ae(ia') & Be(ib') \\ Ce(ic') & De(id') \end{bmatrix} = \begin{bmatrix} Ae(i(a+\alpha+\gamma)) & Be(i(b+\alpha+\delta)) \\ Ce(i(c+\beta+\gamma)) & De(i(d+\beta+\gamma)) \end{bmatrix}$$

$$\begin{bmatrix} c(\phi) & -s(\phi) \\ s(\phi) & c(\phi) \end{bmatrix} \begin{bmatrix} A'_r + iA'_i & B'_r + iB'_i \\ C'_r + iC'_i & D'_r + iD'_i \end{bmatrix} \begin{bmatrix} c(\psi) & -s(\psi) \\ s(\psi) & c(\psi) \end{bmatrix} =$$

$$\begin{bmatrix} c(\phi) & -s(\phi) \\ s(\phi) & c(\phi) \end{bmatrix} \begin{bmatrix} A'_r & B'_r \\ C'_r & D'_r \end{bmatrix} \begin{bmatrix} c(\psi) & -s(\psi) \\ s(\psi) & c(\psi) \end{bmatrix}$$

$$+i \left(\begin{bmatrix} c(\phi) & -s(\phi) \\ s(\phi) & c(\phi) \end{bmatrix} \begin{bmatrix} A'_i & B'_i \\ C'_i & D'_i \end{bmatrix} \begin{bmatrix} c(\psi) & -s(\psi) \\ s(\psi) & c(\psi) \end{bmatrix} \right)$$

A tabela seguinte define algumas transformações internas úteis. Transformações do tipo I trocam os expoentes imaginários dos elementos da matriz em uma das diagonais. Transformações do tipo R, C e D tornam reais os elementos em uma linha, coluna ou diagonal.

Tipo	Valores
I_{main}	$\alpha = -\beta = \gamma = -\delta = (d-a)/2$
I_{off}	$\alpha = -\beta = -\gamma = \delta = (c-b)/2$
R_{up}	$\alpha = \beta = -(b+a)/2\ ;\ \gamma = -\delta = (b-a)/2$
R_{low}	$\alpha = \beta = -(d+c)/2\ ;\ \gamma = -\delta = (d-c)/2$
C_{left}	$\alpha = -\beta = (c-a)/2\ ;\ \gamma = \delta = -(c+a)/2$
C_{right}	$\alpha = -\beta = (d-b)/2\ ;\ \gamma = \delta = -(d+b)/2$
D_{main}	$\alpha = \beta = -(d+a)/2\ ;\ \gamma = -\delta = (d-a)/2$
D_{off}	$\alpha = \beta = -(b+c)/2\ ;\ \gamma = -\delta = (b-c)/2$

É fácil ver que aplicar uma seqüência de transformações internas corresponde à uma única transformação interna, cujos parámetros são a soma dos respectivos parámetros das transformações na seqüência.

Combinando transformações internas e externas, podemos criar TQs para vários algoritmos interessantes. Por exemplo, as QTs de tipo I, II e III na tabela seguinte, podem ser usadas para obter a fatoração SVD de matrizes complexas, à semelhança do algoritmo de Jacobi. A TQ-I aplica R_{low} seguida de uma rotação, tendo o efeito de tormar a matriz triangular superior. A TQ-II aplica D_{main}, I_{off} seguida uma diagonalização. Para matrizes Hermitianas (auto-adjuntas), a diagonalização é obtida com apenas uma TQ do tipo III.

Tipo	Interna	Externa
I	$\alpha = \beta = -(d+c)/2\ ;\ \gamma = -\delta = (d-c)/2$	$\phi = 0\ ;\ \psi = \arctan(C/D)$
II	$\alpha = -(a+b)/2\ ;\ \beta = \gamma = -\delta = (b-a)/2$	$\phi \pm \psi = \arctan(B/(D \mp A))$
III	$\alpha = -\beta = -\gamma = \delta = -b/2$	$\phi = \psi = \arctan(2B/(D-A))/2$

C.9 Probabilidades em Sub-Espaços Lineares

C.10 Exercícios

1. Use as propriedades fundamentais do produto interno para provar:

   (a) A desigualdade de Cauchy-Scwartz: $| < x \mid y > | \leq \|x\|\|y\|$. Sugestão: Calcule $\|x - \alpha y\|^2$ para $\alpha =< x \mid y >^2 / \|y\|$.

   (b) A Desigualdade Triagular: $\|x + y\| \leq \|x\| + \|y\|$.

   (c) Em que caso temos igualdade na desigualdade de Cauchy-Schwartz? Relacione sua resposta com a definição de ângulo entre vetores.

2. Use a definição do produto interno em $\mathcal{R}^n$ para provar a Lei do Paralelogramo: $\|x + y\|^2 + \|x - y\|^2 = 2\|x\|^2 + 2\|y\|^2$.

3. Uma matriz é idempotente,ou um projetor não ortogonal sse $P^2 = P$. Prove que:
   (a) $R = (I - P)$ é idempotente.
   (b) $\mathcal{R}^n = C(P) + C(R)$.
   (c) Todos os autovalores de P são 0 ou +1. Sugestão: Mostre que se 0 é uma raiz do polinômio característico de P, $\varphi_P(\lambda) \equiv \det(P - \lambda I)$, então $(1 - \lambda) = 1$ é raiz de $\varphi_R(\lambda)$.

4. Prove que $\forall P$ idempotente e simétrico, $P = P_{C(P)}$. Sugestão: Mostre que $P'(I - P) = 0$.

5. Prove que o operador de projeção num dado sub-espaço vetorial V, P_V, é único e simétrico.

6. Prove o theorema de Pitágoras: $\forall b \in \mathcal{R}^m, u \in V$ temos que $\|b - u\|^2 = \|b - P_V b\|^2 + \|P_V b - u\|^2$.

7. Suponha termos a fatoração QR de uma matriz A. Considere uma nova matriz $\tilde{A}$ obtida de A pela substituição de uma única coluna. Como podemos atualizar nossa fatoração ortogonal usando apenas $3n$ rotações de linha? Sugestão: (a) Remova a coluna alterada de A e atualize a fatoração usando no máximo n rotações. (b) Compute a nova coluna alterada pelo fator ortogonal corrente, $\tilde{a} = Q'a = R^{-t}A'a$. (c) Adicione $\tilde{a}$ como a ultima coluna de $\tilde{A}$, e torne a atualizar a fatoração com $2n$ rotações.

8. Compute as fatorações LDL e Cholesky da matriz

$$\begin{bmatrix} 4 & 12 & 8 & 12 \\ 12 & 37 & 29 & 38 \\ 8 & 29 & 45 & 50 \\ 12 & 38 & 50 & 113 \end{bmatrix}.$$

9. Prove que
   (a) $(AB)' = B'A'$.
   (b) $(AB)^{-1} = B^{-1}A^{-1}$.
   (c) $A^{-t} \equiv (A^{-1})' = (A')^{-1}$.

10. Descreva quatro algoritmos, para computar $L^{-1}x$ e $L^{-t}x$, acessando a matriz L, triangular inferior de diagonal unitária, por linha ou por coluna.

Appendix D

Probabilidade

D.1 Interpretação Freqüentista

Em muitas circunstâncias estamos interessados em estudar situações que produzem resultados imprevisíveis. Chamamos estas situações de *experimentos*. Por exemplo, ao lançarmos um dado, consideramos seis resultados possíveis, cada resultado correspondendo a uma determinada face do dado voltada para cima. Chamamos de *Espaço Amostral*, A, ao conjunto de todos os possíveis resultados. Espaços amostrais *discretos* contêm um número finito (ou enumerável) de resultados. Sempre que possível apresentaremos a teoria em espaços discretos, onde ela é muito mais simples que em espaços contínuos. Chamamos *evento* um subconjunto do espaço amostral, $E \subseteq A$. Assim, no exemplo do dado, uma representação do espaço amostral é o conjunto $\{F1, F2, F3, F4, F5, F6\}$; nesta representação, o evento "obter uma face ímpar" corresponde ao subconjunto $\{F1, F3, F5\}$.

Nos experimentos que estudaremos é útil atribuirmos a cada um dos eventos um valor numérico. Esta atribuição, ou função, chama-se *variável aleatória*. No exemplo do dado é usual atribuir valores, entre 1 e 6, a cada um dos eventos, i.e. atribuir $X(Fk) = k$.

Consideremos outro exemplo: lançamos dois dados, um verde e outro amarelo, cada qual tendo suas faces numeradas de 1 a 6. No exemplo dos dados verde e amarelo, entre muitas outras, poderíamos considerar as seguintes variáveis aleatórias:

1. O número decimal de dois dígitos cujo primeiro dígito corresponde a face de cima do dado verde, e o segundo dígito corresponde a face de cima do dado amarelo.

2. A soma dos números na face de cima dos dois dados.

Note que construímos duas variáveis aleatórias distintas, sobre um mesmo experimento: o lançamento dos dados verde e amarelo. Note também que pelo valor da primeira

variável aleatória podemos saber exatamente o resultado obtido no experimento; o mesmo já não é verdade para a segunda variável aleatória. Em geral, quando lidamos com um experimento, temos uma dada variável aleatória em mente e, não havendo ambigüidade, quando mencionamos o experimento já subentendemos a variável aleatória apropriada.

Definimos a imagem de um evento $E \subseteq A$ por X, $X(E)$, como o conjunto de todos os valores assumidos por X dentro de E, i.e., $X(E) \equiv \{x = X(e), e \in E\}$. Analogamente definimos a pré-imagem de C por X, $X^{-1}(C)$, como o evento formado pelos resultados do experimento onde a variável X assume valores em C, i.e., $X^{-1}(C) = \{a \in A \mid X(a) \in C\}$.

Dado um experimento que possa ser repetido, interpretamos a *probabilidade* de um evento como a freqüência com que o obtemos como resultado do experimento. A probabilidade de que uma variável aleatória X assuma um valor dentro de um conjunto C, é simplesmente a probabilidade da pré-imagem de C por X. A *distribuição de probabilidade* de uma variável aleatória (discreta) é uma tabela especificando a probabilidade de que a variável assuma cada um dos seus possíveis valores.

Exemplos:

- Um dado é dito honesto se a freqüência com que gera cada um dos valores $\{1, \ldots, 6\}$ é a mesma. A probabilidade de obtermos um dado valor com um dado honesto é portanto de $1/6$.
- No experimento descrito no exemplo dos dados verde e amarelo, assumindo que ambos os dados são honestos, a probabilidade de cada variável aleatória assumir cada um dos valores possíveis é dada pelas tabelas seguintes:

  1. $Pr(X = x) = 1/36$, $\forall x \in \{11, 12, \ldots, 16, 21, \ldots, 26, \ldots, 61, \ldots, 66\}$.
  2. $X^{-1}(2) = \{(1,1)\} \Rightarrow Pr(X = 2) = 1/36$, $X^{-1}(3) = \{(1,2),(2,1)\} \Rightarrow Pr(X = 2) = 2/36$, $\ldots$, $X^{-1}(12) = \{(6,6)\} \Rightarrow Pr(X = 12) = 1/36$.

Propriedades da Probabilidade

Para assegurar a possibilidade de interpretar probabilidade de um evento como a de sua freqüência relativa, exigiremos que uma (medida de) probabilidade sobre o espaço amostral de um experimento aleatório, satisfaça as três propriedades (axiomas) seguintes. Notaremos por $\emptyset$ o conjunto vazio, e por $\bar{B} = A - B$ o complementar de um evento B no espaço amostral A do experimento.

1. $\forall B \subseteq A, \quad Pr(B) \geq 0$,
2. $Pr(A) = 1$,
3. Se $B \cap C = \emptyset$, então $Pr(B \cup C) = Pr(B) + Pr(C)$.

Exercício: Prove, a partir dos axiomas, as seguintes propriedades da probabilidade:

4. $Pr(\emptyset) = 0$,
5. $Pr(\bar{B}) = 1 - Pr(B)$,
6. $Pr(B \cup C) = Pr(B) + Pr(C) - Pr(B \cap C)$

D.2 Inferência

Em muitas situações temos que lidar com experimentos cujas especificações não conhecemos, ou conhecemos incompletamente. Por exemplo: Suponha sabermos que o dado verde foi construído tomando todos os cuidados para garantir que, exceto por pequenas marcas que distinguem suas faces, ele é absolutamente regular. Suponha também que o dado amarelo foi trazido por um desconhecido. Sabemos de antemão que o dado verde é honesto, mas não sabemos a distribuição de probabilidade para o segundo dado.

Para obter alguma informação sobre o dado amarelo podemos lançá-lo repetidas vezes, anotando a seqüência de valores. Este histórico de valores é o que se chama em estatística de uma *amostra*.

A partir da amostra poderíamos induzir alguma informação sobre a distribuição de probabilidade do dado amarelo: poderíamos por exemplo calcular a freqüência relativa de cada um dos valores na amostra; e então usar estas freqüências relativas como *estimativa* da distribuição de probabilidade do experimento. Este é um exemplo do que chamamos *inferência estatística*.

Note que a probabilidade da variável aleatória X assumir o valor x, $\Pr(X = x)$, e a estimativa utilizada, freqüência relativa na amostra, são conceitos distintos: A distribuição de probabilidade é um atributo do experimento em si, enquanto a estimativa baseia-se numa particular amostra de resultados do experimento. No uso prático, geralmente, a amostra resume nossas observações passadas, queremos basear alguma decisão futura na distribuição de probabilidade do experimento (que todavia desconhecemos), e usamos a estimativa como elo de ligação para tomar esta decisão.

Máxima Verossimilhança

Como fazer e justificar estimativas e inferências que sejam ótimas, segundo critérios pré-estabelecidos, é o objeto de estudo da estatística matemática. No caso do dado amarelo, o critério usado para fazer a estimativa foi o da *máxima verossimilhança*: A melhor estimativa da distribuição, segundo o critério da máxima verossimilhança, é a que maximiza a probabilidade de obter, num experimento com a distribuição estimada, uma amostra idêntica a amostra original.

Se a amostra original foi uma seqüência de n resultados, com $n_1,\ n_2, \ldots, n_6$ ocorrências dos valores $1,\ 2, \ldots, 6$, a probabilidade de obter uma amostra com as mesmas feqüências, independentemente da ordem, é

$$Pr(X = [n_1,\, n_2, \ldots n_6]) = \frac{n!}{n_1!\, n_2! \ldots n_6!} p_1^{n_1} p_2^{n_2} \ldots p_6^{n_6}$$

É um exercício de cálculo mostrar que a distribuição que maximiza esta probabilidade é a das freqüências relativas, $p_i = f_i$, onde $f_i = n_i/n$. Prove este resultado no caso de uma moeda.

D.3 Esperança

Dado um experimento, o valor esperado de uma variável aleatória X, ou sua *esperança*, é a média aritmética sobre o conjunto dos valores possíveis, $x \in A$, ponderados pela distribuição de probabilidade:

$$E(X) = \sum_{x \in X(A)} x * \Pr(X = x) \quad .$$

Dada uma amostra, i.e. uma seqüência de valores,

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad ,$$

definimos a sua **média** aritmética como sendo

$$\mu_x = \frac{1}{n} \sum_{i=1}^{n} x_i \quad .$$

Dada uma amostra, usaremos a sua média como estimativa da esperança. Esta estimativa é consistente com a estimativa da distribuição de probabilidade idêntica às freqüências relativas na amostra, que por sua vez tinha justificativa no critério da máxima verossimilhança.

Propriedades de Transformação

Estaremos sempre interessados em estudar transformações lineares de variáveis aleatórias, $Z = \alpha X + \beta Y + \gamma$ e como estas transformações se refletem nas quantidades estatísticas a serem definidas, por exemplo:

$$E(\alpha X + \beta Y + \gamma) = \alpha E(X) + \beta E(Y) + \gamma$$

A prova pode ser dividida em duas partes:

$$\begin{aligned} E(\alpha X+\gamma) &= \sum_x (\alpha x+\gamma)\Pr(X=x) \\ &= \gamma + \alpha \sum_x x\Pr(X=x) \\ &= \gamma + \alpha E(X) \end{aligned}$$

$$\begin{aligned} E(X+Y) &= \sum_{x,y} (x+y)\Pr(X=x \wedge Y=y) \\ &= \sum_{x,y} x\Pr(X=x \wedge Y=y) \quad + \sum_{x,y} y\Pr(X=x \wedge Y=y) \\ &= \sum_x x\Pr(X=x \wedge Y \in Y(A)) + \sum_y y\Pr(Y=y \wedge X \in X(A)) \\ &= E(X)+E(Y) \end{aligned}$$

D.4 Variância

A variância de uma variável aleatória é uma medida de erro ou dispersão desta variável em torno da sua esperança:

$$Var(X) = E((X-E(X))^2) \quad .$$

É fácil ver que também podemos calcular a variância como

$$\begin{aligned} Var(X) &= E((X-E(X))^2) \\ &= E(X^2 - 2XE(X) + E(X)^2) \\ &= E(X^2) - E(X)^2 \end{aligned}$$

Desvio Padrão

Desvio padrão, $\sigma_x = \sqrt{var(x)}$ tem a mesma dimensão, ou unidade de medida, que $E(x)$ ou x, i.e., é uma medida de desvio comensurável com a média ou os valores assumidos pela variável aleatória, sendo portanto de interpretação mais natural.

Justificativa da Norma Quadrática

Existem outras alternativas para medir o tamanho do erro, x, além da norma $L_2 = (x'x)^{1/2}$; Por exemplo as normas $L_1 = \mathbf{1}'\text{abs}(x)$, ou $L_\infty = \max(\,\text{abs}(x)\,)$. Tanto L_1 como L_∞ são usadas na estatística em certas situações especiais, todavia, L_2 é usada na maioria das situações. Além de ser computacionalmente mais simples, é a única que tem a propriedade de ser invariante por uma (pelo grupo de) rotação.

Covariância

A covariância entre duas variáveis aleatórias, X e Y, é definida como

$$Cov(X,Y) = E((X - E(X)) * (Y - E(Y))) \quad .$$

A covariância é uma medida da "interdependência" dos desvios de ambas as variáveis em relação a média. Adiaremos uma interpretação intuitiva mais detalhada para o conceito de correlação, discutido adiante. Por hora, podemos verificar as seguintes propriedades:

$$\begin{aligned} Cov(X,Y) &= E(XY - YE(X) - XE(Y) + E(X)E(Y)) \\ &= E(XY) - E(Y)E(X) \end{aligned}$$

donde

$$Cov(X,X) = E(X^2) - E(X)^2 = Var(X)$$

Propriedades de Transformação

Lema :

$$\begin{aligned} &Cov(\alpha X + \beta Y + \gamma, Z) \\ &= E((\alpha X + \beta Y + \gamma)Z) - E(\alpha X + \beta Y + \gamma)E(Z) \\ &= \alpha E(XZ) + \beta E(YZ) + \gamma E(Z) - \alpha E(X)E(Z) - \beta E(Y)E(Z) - \gamma E(Z) \\ &= \alpha Cov(X,Z) + \beta Cov(Y,Z) \end{aligned}$$

Do lema segue que:

$$Var(\alpha X + \beta Y + \gamma) = \alpha^2 Var(X) + \beta^2 Var(y) + 2\alpha\beta Cov(X,Y)$$

Notação Matricial

$$Cov(X)_{i,j} \equiv Cov(X_i, X_j)$$

Dado X um vetor de n variáveis aleatórias, A uma matriz real $m \times n$, e b um vetor real $m \times 1$, os resultados precedentes implicam na forma genérica da esperança e variância de uma transformação linear, que é dada por:

$$E(AX + b) = AE(X) + b$$

$$Cov(AX + b) = A\,Cov(X)\,A'$$

Em estatística é usual a notação $Cov(X)_{i,j} = \sigma_{i,j}$. Nesta notação o desvio padrão é $\sigma_i = \sqrt{\sigma_{i,i}}$.

Consideremos realizar n vezes e independentemente, um experimento cujos resultados são medidos pelo vetor de variáveis aleatórias $X = [X_1, \ldots, X_n]$, a variância da média é dada por

$$E(\bar{X}) = E(\ (1/n)\ \mathbf{1}'X\ ) = (1/n)\ \mathbf{1}'E(X) = E(X_1)$$

na ultima equação denotamos a somatória $\sum X$ por $\mathbf{1}'X$, onde $\mathbf{1}' = [1, 1, \ldots 1]$.

$$\begin{aligned} Var(\bar{X}) &\equiv Var(\ (1/n)\ \mathbf{1}'X\ ) \\ &= (1/n^2)\ \mathbf{1}'\ \mathrm{diag}([\sigma_{1,1}, \ldots \sigma_{n,n}])\ \mathbf{1} \\ &= (1/n^2)(\ \sigma_{1,1} + \ldots + \sigma_{n,n}\ ) \\ &= (1/n)\sigma_{1,1} \end{aligned}$$

Correlação

A correlação entre duas variáveis aleatórias, $Cor(X, Y)$ ou $\rho_{i,j} \equiv\ Cor(X_i, X_j)$, é a covariância "normalizada" pelo desvio padrão:

$$Cor(X, Y) = \frac{Cov(X, Y)}{\sigma_X \sigma_Y} \quad \text{ou} \quad \rho_{i,j} = \frac{\sigma_{i,j}}{\sigma_i \sigma_j}$$

A correlação é uma medida incompleta de independência entre variáveis aleatórias, como mostraremos a seguir:

Dois eventos, B e C, são *independentes* sse $Pr(B \cap C) = Pr(B) * Pr(C)$. Duas variáveis aleatórias, X e Y, são independentes sse, para qualquer conjunto de valores, C e D, $Pr(X \in C \mathrm{e} Y \in D) = Pr(X \in C) * Pr(Y \in D)$. Temos então as seguintes propriedades:

- Se duas variáveis aleatórias (de variância limitada), X e Y, são independentes, então $Cov(X, Y) = 0$, pois $E(XY) = E(X)E(Y)$.
- Todavia, correlação nula não garante independência! Considere duas variáveis aleatórias definidas sobre os resultados de um dado honesto: X assumindo valor -1 em $F1$, 1 em $F6$, e valor 0 em todas as outras faces; Y assumindo valor 1 em $\{F1, F6\}$, e valor 0 em todas as outras faces. As variáveis aleatórias X e Y não são independentes, embora tenham correlação nula (verifique).
- No caso de uma dependência linear, $Y = \alpha X + \gamma$, temos que:

  $$Cor(Y, X) = \frac{Cov(\alpha X + \gamma, X)}{\sigma(\alpha X + \gamma)\sigma(X)} = \frac{\alpha Var(X)}{\sqrt{\alpha^2}\sigma_X \sigma_X} = sign(\alpha)$$

  onde $sign(x) \equiv\ \ 1$ se $x > 0$, $\ -1$ se $x < 0$, e $\ 0$ se $x = 0$.

Provemos finalmente que $-1 \leq \rho_{i,j} \leq 1$:
Tomemos $X = [X_1, X_2]'$, e $Y = [a_1, a_2]X$.

$$\begin{aligned} Var(Y) &= \left[\begin{array}{cc} a_1 & a_2 \end{array}\right] \left[\begin{array}{cc} \sigma_{1,1} & \sigma_{1,2} \\ \sigma_{2,1} & \sigma_{2,2} \end{array}\right] \left[\begin{array}{c} a_1 \\ a_2 \end{array}\right] \\ &= \left[\begin{array}{cc} a_1 & a_2 \end{array}\right] \left[\begin{array}{cc} \sigma_1 & 0 \\ 0 & \sigma_2 \end{array}\right] \left[\begin{array}{cc} 1 & \rho_{1,2} \\ \rho_{2,1} & 1 \end{array}\right] \left[\begin{array}{cc} \sigma_1 & 0 \\ 0 & \sigma_2 \end{array}\right] \left[\begin{array}{c} a_1 \\ a_2 \end{array}\right] \\ &= \left[\begin{array}{cc} b_1 & b_2 \end{array}\right] \left[\begin{array}{cc} 1 & \rho \\ \rho & 1 \end{array}\right] \left[\begin{array}{c} b_1 \\ b_2 \end{array}\right] \\ &= b_1^2 + 2\rho b_1 b_2 + b_2^2 \end{aligned}$$

Observemos agora que, pela definição de variância, $Var(Y) \geq 0$. Observemos ainda que se $-1 \leq \rho \leq 1$, podemos sempre "completar os quadrados" de modo a reescrever $Var(Y) = (b_1 \pm b_2)^2 + \text{abs}((1 \pm \rho)b_1 b_2)$, quantidade obviamente positiva. Todavia se $\text{abs}(\rho) > 1$, podemos sempre escolher valores para b_1 e b_2 em $\{-1, 1\}$ que tornam $b_1^2 + 2\rho b_1 b_2 + b_2^2 < 0$, uma contradição.

D.5 Espaços de probabilidade

A exposição das seções precedentes é válida para espaços amostrais finitos. Para espaços não finitos, especialmente os não enumeráveis, uma estrutura mais mais complexa é necessária (vide [Billingsley]). Um espaço de probabilidade, $(\Omega, \mathcal{A}, P)$, é um espaço amostral, Ω, uma σ-algebra, $\mathcal{A}$ e uma medida de probabilidade, $P : \mathcal{A} \mapsto [0, 1]$. Uma variável aleatória é uma função $x : \Omega \mapsto \mathcal{R}$, tal que $x^{-1}(t) \in \mathcal{A}$, $\forall t \in \mathcal{R}$.

A distribuição (cumulativa) de uma variável aleatória x, $F : \mathcal{R} \mapsto [0, 1]$, é definida por $F(t) \equiv P(\{\omega \mid x(\omega) \leq t\})$.

A esperança de uma variável aleatória, $E(x)$, é definida por

$$E(x) \equiv \int_{-\infty}^{\infty} t dF(t).$$

No caso de uma distribuição diferenciável, ou discreta, temos, respectivamente

$$E(x) = \int t f(t) dt \quad \text{ou} \quad E(x) = \sum t f(t) \ .$$

O k-ésimo **momento** de uma variável aleatória é a esperança de sua k-ésima potência (omitiremos o índice k para $k = 1$), $\mu_k(x) \equiv E(x^k)$. O k-ésimo **momento central** de uma variável aleatória é a esperança da k-ésima potência do desvio em relação a sua esperança, $\mu_k^c(x) \equiv E((x - \mu_1)^k)$. A variância corresponde ao segundo momento central.

Podemos agora considerar um espaço vetorial sobre as variáveis aleatórias (neste Espaço de probabilidade) com segundo momento limitado, $L^2(\Omega, \mathcal{A}, P)$, de elementos

$\{x \mid E(x^2) < \infty\}$. A origem de L^2 é a variável aleatória identicamente nula, $x \equiv 0$, e o oposto de uma variável aleatória x é $-x = (-1)x$ (explique).

As operações usuais de soma e produto por escalar de variáveis aleatórias está bem definida neste espaço, pois

1. $E((\alpha x)^2) = \alpha^2 E(x^2) < \infty$.

2. $E((x+y)^2) \leq E(2x^2 + 2y^2) \leq 2E(x^2) + 2E(y^2) < \infty$.

Em L^2 adotamos a seguinte definição de produto interno:

$$< X \mid Y > \equiv E(XY) \ ,$$

que satisfaz trivialmente as propriedades de simetria, linearidade, e semi-positividade (prove). Algumas tecnicalidades são necessárias para assegurar a positividade deste produto interno.

D.6 Utilidade e Decisões

Consideremos a seguinte aposta: Um banqueiro lança um dado honesto e paga o resultado, X o número na face de cima, em gramas de ouro. Um apostador, para apostar, deve pagar 4 gramas de ouro. Consideraremos nesta seção um modelo matemático, a *teoria estatística da decisão*, que nos dá critérios para tomar a decisão de participar ou não da aposta.

Além da descrição da aposta em si, precisamos de uma descrição das preferências do apostador. Esta descrição é dada por uma *função de ganho*, $G(x)$. $G(x)$ nos dá o ganho atribuído pelo apostador a cada um dos possíveis valores assumidos pela variável aleatória que caracteriza a aposta.

Consideremos um apostador, A, que se importe apenas com a quantidade de ouro ganha ou perdida no final da aposta. As preferências de A são adequadamente descritas pela função de ganho $G_A(x) = x - 4$.

Considere um segundo apostador, B, que possui 4 gramas de ouro, mas que só poderá participar de um negócio que com certeza lhe triplicará o capital se dispuser de 6 gramas. A função de ganho de B é: $G_B(x) = x - 4$ se $x \leq 5$, e $G(6) = 3 * 6 - 4 = 14$.

Um critério para medir a utilidade de um apostador, com função de ganho $G()$, participar de uma aposta caracterizada pela variável aleatória X, é o *ganho esperado*:

$$U(X) = E(G(X)) = \sum_x G(x) * \Pr(X = x)$$

Nos dois exemplos anteriores teríamos

$$\begin{aligned} U_A(X) &= E(G_A(X)) = \sum_{x=1}^{6}(x-4)*(1/6) = -0.5 \\ U_B(X) &= E(G_B(X)) = \sum_{x=1}^{5}(x-4)*(1/6) \ +14*(1/6) = 1.5 \end{aligned}$$

ou seja, como resultado final de seu envolvimento na aposta, A espera perder 0.5 gramas, enquanto B espera ganhar 1.5 gramas de ouro. Usando o critério da utilidade esperada B deveria fazer a aposta, enquanto A todavia não.

Critério de Média-e-Variância

Em finanças e economia é usual considerar investidores (ou agentes econômicos, produtores, consumidores, etc.) que desejam maximizar seus ganhos (ou minimizar seus custos). Quando os ganhos do investidor dependem do valor de variáveis aleatórias, é razoável assumir que este deseje maximizar o valor esperado do ganho. Todavia, apenas a esperança do ganho, pode ser uma caracterização incompleta das preferências do investidor. Muitos investidores são "avessos ao risco", preferindo usar um processo cujo ganho tem um dado valor esperado e variância, a um segundo processo cujo ganho tem valor esperado ligeiramente superior, mas mas variância bem maior. Usaremos a seguir a teoria de utilidade para estudar situações deste tipo.

A função utilidade (ou critério) da média-e-variância é definida como:

$$U(G, X, \alpha, \beta) = \alpha E(G(X)) - \beta Var(G(X))$$

Esta função utilidade (ou critério) é muito empregado em Economia e Finanças, onde o parâmetro β/α é interpretado como uma medida da *aversão ao risco* do investidor. Tivemos oportunidade de desenvolver modelos deste tipo no capítulo 3.

Um critério quadrático pode também ser visto como a aproximação truncada da série de Taylor de uma função genérica. Consideremos a variável aleatória auxiliar $y = G(x)$, e a série de taylor:

$$U(y) = U[E(Y)] + \sum_{k=1}^{\infty} U^{(k)}[E(Y)] * (y - E(Y))^k/k!$$

Truncando a série após o termo quadrático, temos

$$U(y) \approx U[E(Y)] + U'[E(Y)] * (y - E(Y)) + U''[E(Y)] * (y - E(Y))^2/2$$

Do valor esperado desta aproximação obtemos

$$E(U(Y)) \approx U[E(Y)] + (U''[E(Y)]/2)Var(Y)$$

A última aproximação nos dá uma nova interpretação para o parâmetro β no critério da média-e-variância. Em particular, obtemos uma legítima aversão ao risco, i.e. $\alpha > 0$ e $\beta > 0$, na situação convencional de uma utilidade côncava, onde $U()$ é crescente, $U'() > 0$, com incrementos decrescentes, $U''() < 0$. Poderíamos aprofundar o estudo de como obter os parâmetros da função quadrática a partir de pressupostos sobrea a "real" função utilidade do investidor; todavia, para aplicações a finanças, temos disponíveis métodos bem mais diretos, conforme vimos no capítulo 3.

D.7 Modelos Lineares

Um modelo linear tem a forma

$$\tilde{y} = Ap + \tilde{v}$$

onde A, $m \times n$, A é conhecida e de posto pleno, $\rho(A) = n$.

A *observação* y é uma realização do vetor aleatório $\tilde{y}$, p são os *parâmetros* do modelo a serem estimados, e A a *matriz de coeficientes*. O *ruído* $\tilde{v}$ é um vetor de variáveis aleatórias, não observadas, e de primeiro e segundo momentos $E(\tilde{v}) = 0$ e $Cov(\tilde{v}) = \sigma^2 I$. O ruído é interpretado como erros de medida, flutuações no processo etc. Um ruído cuja matriz de covariância é a identidade é dito **branco**. Estudaremos agora modelos lineares simples, cujo ruído é branco.

Como $E(\tilde{y}) = E(Ap + \tilde{v}) = Ap \in C(A)$, é geometricamente intuitivo considerarmos $\hat{y} = P_A y$ ou equivalentemente, $\hat{y} = Arg\min_{z \in C(A)} \|y - z\|$, ou ainda,
$\hat{y} = A\hat{p}$, $\hat{p} = Arg\min_p \|y - Ap\|$, como estimativas, respectivamente, do valor médio de $\tilde{y}$ e dos parâmetros do modelo. Estes são os **estimadores de mínimos quadrados** (LSE).

Um estimador de um escalar $\pi = l'p$ em função de y é **linear** sse tem a forma $\Pi(y) = \alpha + a'y$. Um estimador $\Pi(y)$, do escalar $\pi = l'p$, é **não tendencioso** sse $E(\Pi(y)) = \pi$. $\beta + b'y$ é o melhor estimador linear não tendencioso, **BLUE**, de $\pi = l'p$ sse dentre todos os estimadores lineares e não tendenciosos de π, este for de mínima variância. Mostremos agora o seguinte resultado (Gauss-Markov):

Num dado modelo linear, o BLUE de uma combinação linear dos parâmetros, $\pi = l'p$, é dado por $\hat{\pi}$, definido pela mesma combinação linear do LSE de p, $\hat{\pi} = \Pi(y) = l'\hat{p}$.

Em primeiro lugar notemos que $\hat{y} = P_A y$ é uma transformação linear de y; Ademais, como assumimos que A tem posto pleno, é sempre possível encontrar um vetor $a' \mid l' = a'A$. Assim $\hat{\pi}$, $l'\hat{p} = a'Ap = a'\hat{y} = a'P_A y$ é realmente um estimador linear em y. $\hat{\pi}$ é também não tendencioso, pois

$$E(\hat{\pi}) = E(l'\hat{p}) = E(a'P_A\hat{y}) = a'P_A E(\hat{y}) = a'P_A Ap = a'Ap = l'p \equiv \pi.$$

Finalmente, da formula do projetor, podemos explicitar

$$\hat{\pi} = a'P_A y = a'A(A'A)^{-1}A'y = l'(A'A)^{-1}A'y.$$

Consideremos agora o BLUE de $\pi = l'p$, $\pi^* = \beta + b'y$. Como o BLUE é por definição não tendencioso, devemos ter $l'p = E(\beta + b'\tilde{y}) = \beta + b'Ap$, qualquer que seja o verdadeiro parâmetro p, portanto $b'A = l'$. A variância deste estimador é

$$Var(\pi^*) = b'Ib = b'b.$$

Portanto o BLUE de $\pi = l'p$ será dado por $\pi^* = b'y$, onde b' é especificado pelo problema de programação quadrática

$$b = Arg \min_{c|A'c=l} c'c.$$

é fácil verificar que $b' = l'(A'A)^{-1}A'$ resolve o sistema normal deste problema (verifique), o que demonstra que $\pi^* = \hat{\pi}$, (teorema de Gauss-Markov).

Q.E.D.

Usaremos agora a fatoração de Cholesky para transformar um modelo linear geral, cujo ruído não é branco, num modelo linear simples equivalente. Consideremos

$$y = Ap + v \ , \ E(v) = 0 \ , \ Cov(v) = V = L'L \ ,$$

e o modelo simplificado

$$\bar{y} \equiv L^{-1}y = L^{-1}Ap + L^{-1}v = \bar{A}p + \bar{v} \ .$$

Note que

$$Cov(\bar{v}) = L^{-1}(LL')L^{-t} = (L^{-1}L)(L'L^{-t}) = I \ ;$$

Assim, transformando os dados e a matriz de coeficientes do modelo geral pela inversa do fator de Cholesky da matriz de covariância do ruído, podemos estimar os parâmetros num modelo simples equivalente. No modelo simplificado estamos pois minimizando

$$\|\bar{y} - \bar{A}p\|^2 = (y - Ap)'L^{-t}L^{-1}(y - Ap) = (y - Ap)'W(y - Ap) \ ,$$

onde $W = V^{-1}$ é denominada a **matriz de informação** do ruído original.

D.8 Interpretação Bayesiana

Consideremos uma partição do espaço amostral,

$$A = \bigcup_1^n A_i \ , \quad Pr(A_i) > 0 \ , \ A_i \cap A_j = \emptyset, i \neq j.$$

A probabilidade condicional de A_i dado o evento B é, por definição, $Pr(A_i \mid B) = Pr(A_i \cap B)/Pr(B)$. Usando $Pr(B) = \sum_1^n Pr(A_i)Pr(B \mid A_i)$, temos que

$$Pr(A_i \mid B) = Pr(A_i)Pr(B \mid A_i) / \sum_{j=1}^{n} Pr(A_j)Pr(B \mid A_j)$$

Este é o chamado teorema ou regra de Bayes, que nos permite calcular $Pr(A_i \mid B)$ a partir da informação "inversa", $Pr(B \mid A_j),\ j = 1 \ldots n$.

Consideremos agora uma distribuição de probabilidades que é função de um parâmetro dentro de um domínio bem especificado, o espaço paramétrico,

$$f(x \mid \theta)\ ,\quad \theta \in \Theta.$$

Quando estimamos o parâmetro da distribuição, podemos expressar nossa incerteza assumindo que o próprio paraâmetro θ tem uma distribuição $p(\theta)$ no espaço paramétrico. Assim, a expressão $f(x \mid \theta)$ deve ser interpretada como a probabilidade condicional de x para um dado valor de θ.

Dada uma amostra de n experimentos independentes, $x = [x_1 \ldots x_n]$, sua distribuição conjunta é

$$f(x \mid \theta) = \prod_{i=1}^{n} f(x_i \mid \theta),$$

que podemos integrar sobre θ obtendo

$$g(x) = \int_{\Theta} f(x \mid \theta) p(\theta) d\theta.$$

Usando a regra de Bayes, podemos também obter a probabilidade condicional

$$f(\theta \mid x) = f(x \mid \theta) p(\theta) / g(x).$$

Neste contexto $p(\theta)$ é denominada a distribuição "a priori" (inicial) de θ, $f(\theta \mid x)$ é a probabilidade "a posteriori" (depois de observar x), e o cálculo de $f(\theta \mid x)$ é denominado a "operação Bayesiana".

A operação Bayesiana é "recursiva", i.e. se um novo vetor de observações y estiver disponivel, podemos calcular a nova posteriori $f(\theta \mid x, y)$ usando como priori na operação Bayesiana $f(\theta \mid x)$, que já incorporou toda a informação contida em x. Esta recursividade implica em grande eficiência computacional pois, ao adquirirmos uma nova observação, podemos "atualizar" a posteriori realizando uma operação Bayesiana sobre esta única observação, sem ter que refazer o cálculo usando todas as observações anteriores. Podemos também postergar o cálculo da constante de normalização $g(x)$, que envolve uma integração que pode ser trabalhosa, usando nos passos intermediarios de aquisição de dados a função de verossimilhança,

$$l(\theta \mid x) = f(x \mid \theta) p(\theta) \propto f(\theta \mid x).$$

A noção de priori é algo controversa. Alguns argumentam que nossa incertesa sobre o verdadeiro valor do parâmetro não implica que este possa ser tratado como uma variável aleatória, tendo uma distribuição de probabilidade. Normalmente utiliza-se uma priori

"neutra" como uma distribuição uniforme sobre todo o espaço paramétrico. Outros argumentam que nada há de neutro nesta escolha, sendo a priori uniforme tão subjetiva como qualquer outra. Um contra-argumento é a coveniência de incorporar a subjetividade da priori na modelagem, incorporando na priori "conhecimento subjetivo" i.e. previamente adquirido fora de um contexto estatístico formal, mas que nem por isto deveria ser menosprezado.

A linha subjetivista postula que o próprio conceito de probabilidade é essencialmente subjetivo. Esta linha encara o conceito de probabilidade como retratando nossa ignorancia (ou informação parcial) a respeito do mundo, bem à maneira da física estatística no contexto das leis determinísticas da física clássica. O debate sobre a validade de uma distribuição no espaço paramétrico lembra as críticas históricas ao conceito de uma função de onda não observavel em mecânica quântica. A postura subjetivista novamente lembra a historia da mecânica quântica, os célebres debates Einstein vs Heisemberg sobre o princípio de incerteza, só que agora com posições invertidas, i.e. a posição Bayesiana parece inovadora ao propor uma distribuição no espaço paramétrico, sendo todavia conservadora ao procurar refúgio na interpretação subjetivista para o conceito de probabilidade.

D.9 Exercícios

1. Formule o problema de mínimos quadrados como um problema de programação quadrática.
   (a) Assuma dada uma base N de $N(A)$.
   (b) Calcule diretamente o resíduo $z = b - y$ em função de A.
2. Prove que a distribuição F é
   (a) Não decrescente, com $F(-\infty) = 0$ e $F(\infty) = 1$.
   (b) Sempre contínua à esquerda e continua à direita exceto num numero enumerável de pontos.
3. Se as variáveis aleatórias x e y têm, respectivamente, distribuições F e G, determine a distribuição de
   (a) αx.
   (b) $x + y$.
   (c) xy.
4. Dadas x e y, variáveis aleatórias, mostre que:
   (a) $E(\alpha x + \beta y) = \alpha E(x) + \beta E(y)$.

(b) $Var(\alpha x + \beta y) = \alpha^2 Var(x) + \beta^2 Var(y) + 2\alpha\beta Cov(x, y)$.

5. Dado x, um vetor de variáveis aleatórias, mostre que:

   (a) $E(Ax) = AE(x)$.

   (b) $Cov(Ax) = ACov(x)A'$.

6. O **traço** de uma matriz A é definido por $tr(A) \equiv \sum A_i^i$. Mostre que

   (a) $tr(A + B) = tr(A) + tr(B)$.

   (b) $tr(AB) = tr(BA)$.

   (c) $x'Ay = tr(Ayx') = tr(yx'A)$.

   (d) Se A, $m \times n$, tem posto pleno, $\rho(A) = n$, então $tr(P_A) = n$.

   (e) Nas condições do ítem anterior, definindo $R_A = (I - P_A)$, temos que $tr(R_A) = m - n$.

7. Dado x um vetor de variáveis aleatórias, $E(x) = \mu$, $Cov(x) = V$, e S uma matriz simétrica, temos que
$$E(x'Sx) = tr(SV) + \mu'S\mu.$$
Sugestão: Use que $E(x'Sx) = tr(SE(xx'))$.

8. Num modelo linear de posto pleno, com $\hat{p} = (A'A)^{-1}A'y$ estimando o parâmetro p, mostre que

   (a) $Cov(\hat{p}) = \sigma^2(A'A)^{-1}$.

   (b) O erro quadrático médio, $MSE \equiv \|y - P_A y\|^2/(m - n)$, é um estimador não tendencioso de σ^2. Sugestão: Use $MSE = y'R_A y/(m-n)$, onde $R_A = (I-P_A)$.

Appendix E

Programas

E.1 bigode.m

```
%Mestrado Profissionalizante
%Modelagem Matematica em Financas - Turma III
%Programacao dinamica: Bigode
%Daniel Granja

%Condicoes iniciais
infty = 1E10; eps = 1E-10;
h=2; q = 5; fmax = 3; kmax = 2;
s0 = 10; b0 = 20; sx = 5; c0 = 5;

%constantes independentes dos movimentos trinomiais
m1s = 1; m1b = 1; m2s = 1; m2b = 1; m3s = 1; m3b = 1;
fsu  = 1.3; psu = 0.3; psd = 0.7;
fbu = 0.9; pbu = 0.6; pbd = 0.4;
fbe = 0.0; fse = 0.0;

%simplificacao para binomial
psd=1-psu; pbd = 1 - pbu;

%constantes independentes
fbu=1.1; pbu=0.6;
fsd=1/fsu; fbd=1/fbu;

%h=7;%Horizonte de tempo, isto eh, numero de periodos
fmax=3; %estoque fisico maximo de salsichas pereciveis
s0=10; %preco de partida
r0=20; %o fator de desconto,B eh definido como: 1/(1+r),
%onde a taxa de juros eh ao ano
```

```
%-Opcoes

c0=3; %custo de uma opcao de salsicha para comprar
%q salsichas a exercicio sx/salsicha.
opt=1; %estoque maximo de opces. Dara' a possibilidade
%de formar q*opt salsichas longa-vida.
sx=5; %Preco de exercicio.
%O custo de exercicio de q opcoes eh sx*q
q=3; %numero de salsichas no exercicio de uma opcao;
if (opt==0) q=0; end

%formacao dos estados possiveis da natureza
%passeio aleatorio binomial geometrico estacionario
fsu=1.3; psu=0.3; psd=1-psu;
fbu=1.1; pbu=0.6; pbd=1-pbu;
fsd=1/fsu;
fbd=1/fbu;
%constantes limitadoras para otimizacao:
infty=1E10;
eps=1E-10;

%arvore binomial para salsicha `a vista
S(1,1)=s0; PS(1,1)=1; dimst=1;
for t=2:(h+2)
   for m=1:t
      if (m<t)
         S(t,m)=S(t-1,m)*fsd;
         if (m==1) PS(t,m)=PS(t-1,m)*psd;
         else
            PS(t,m)=PS(t-1,m)*psd+PS(t-1,m-1)*psu;
         end
      else
         S(t,m)=S(t-1,m-1)*fsu;
         PS(t,m)=psu^(t-1);
      end
      dimst=dimst+1;
   end %m
end %t

%arvore binomial para taxa de juros de 6 meses,
%mas expressa ao ano
R(1,1)=r0; PR(1,1)=1; dimRt=1;
for t=2:(h+2)
   for m=1:t
      if (m<t)
         R(t,m)=R(t-1,m)*fbd;
         if (m==1) PR(t,m)=PR(t-1,m)*pbd;
         else
```

```
            PR(t,m)=PR(t-1,m)*pbd+PR(t-1,m-1)*pbu;
         end

      else
         R(t,m)=R(t-1,m-1)*fbu;
         PR(t,m)=pbu^(t-1);
      end
      dimRt=dimRt+1;
   end %m
end %t

%inicializa controle otimo
for t=(h+1):-1:1
   instante(t)=0;
   bestwl(t)=666;
   bestwk(t)=666;
   bestfoti(t)=666;
end%t
instante(h+1)=0;

% prepara para condicao de contorno

%obs: como nao eh possivel no matlab declarar matriz
%com indice zero, adiciono 1 nos limites de estoque
for k=1:(opt+1)
   for l=1:(k*q+fmax+1)
      for Rind=1:dimRt
         for sind=1:dimst

            fot(h+2,sind,Rind,l,k)=0;
            %a condicao de contorno eh para o
            %instante seguinte ao final

         end%sind
      end%Rind
   end%l
end%k

%bestfoti=infty;
estoquetp1(h+1)=0;
estoque_p_tpl(h+1)=0;

v=1;
%%%%%%%%%%%%%%%%%%%%%%
%Resolucao da equacao de Bellman para t>1
%%%%%%%%%%%%%%%%%%%%%%
for k=1:(opt+1)
   %novamente, no matlab, nao posso partir de zero...
```

```
%entao somo 1.Entenda-se que k=1 eh primeiro estado
%das quantidades das opcoes, onde tenho quantidade zero.
%Define-se qk como a quantidade, propriamente dita nesse estado
qk=k-1;

% a equacao de Bellman eh resolvida do final para o inicio
for t=(h+1):-1:2

   tp1=t+1; %instante ‘a frente do instante atual, t
   instante(t)=t
   %a cada loop em t, a funcao custo do instante t+1
   %recebe o resultado da otimizacao do instante anterior:
   dimRtp1=dimRt;
   dimstp1=dimst;
   for j=1:(opt+1)
      for l=1:(j*q+fmax+1)
         for Rind=1:dimRtp1
            for sind=1:dimstp1
               fotp1(t,sind,Rind,l,j)=fot(t+1,sind,Rind,l,j);
            end%sind
         end%Rind
      end%l
   end%k
   dimRt=t;%redimensionamento
   dimst=t;

   %inicia varredura dos estados

   %estados definidos pelas quantidades possiveis de salsichas
   for l=1:(fmax+1)
      %novamente, no matlab, nao posso partir de zero...entao adiciono 1.
      %Entenda-se que l=1 eh primeiro estado das quantidades de salsichas,
      %onde tenho quantidade zero.
      %Define-se ql como a quantidade, propriamente dita nesse estado
      ql=l-1;
      % if (ql==0) qlptp1=0; end
      qtot=ql+qk*q*0;
      %estados definidos pelas possibilidades de taxas de juros
      %no instante presente
      for mr=1:t
         %como a arvore eh binomial, em cada instante de tempo
         %ha t estados para r
         realr=R(t,mr);
         %estados definidos pelas possibilidades de precos
         %das salsichas no mercado ‘a vista
         for ms=1:t
            reals=S(t,ms);
            bestfoti(t)=6666;
```

```
%wk e wl sao as variaveis de controle
%limites para numeros de opcoes executadas
if (qk>0 & qtot<=(fmax+q*qk)) wkmax=1; else wkmax=0; end
%estados de possibilidades de execucao de opcoes
for wk=0:wkmax


   if (ql==0 & wk==0) wlmin=1; else wlmin=0; end
   %o que pode ser comprado, se o estoque de salsichas
   %pereciveis estiver abaixo do maximo, eh o quanto falta
   %para chegar nesse maximo, mais um, pois esta eh a demanda
   %para o periodo corrente:
   if (ql<=fmax & wk==0) wlmax=(fmax-ql)+1; else wlmax=0; end
   %estados definidos pelas posibilidades de quantidade de compra:
   for wl=wlmin:wlmax
      %equacao de evolucao dos estoques:
      qtottp1=qtot+wk*q+wl-1;
      estoqueauxtp1(t)=qtottp1;
      %variavel auxiliar para ser usada na decisao do melhor
      %qlptp1=wl-1;
      ltottp1=qtottp1+1;
      %variavel estado correspondente a quantidade qltp1
      %equacao de evolucao dos estoques de opcoes:
      qktp1=qk-wk;
      ktp1=qktp1+1;
      %variavel estado correspondente a quantidade qktp1
      %custo do controle:
      auxfoti=wk*q*sx+wl*reals;
      %tratamento dos estados possiveis y=x(t+1)
      %wr e ws sao as transicoes possiveis de t para t+1
      for wr=mr:(mr+1)
         % possibilidade seguinte eh para cima ou para baixo
         %pwr=PR(t+1,wr)/PR(t,mr);
         if (wr==mr) pwr=pbd; else pwr=pbu; end
         %probabilidade de transicao para o proximo:
         %para cima e para baixo
         for ws=ms:(ms+1)%idem
            if (ws==ms) pws=psd; else pws=psu; end
            %pws=PS(t+1,ws)/PS(t,ms);
            %adicionamos ao custo do controle o custo futuro
            %multiplicado pela probabilidade (tiramos a media,
            %portanto) de transicao do estado presente para os
            %possiveis estados futuros e descontamos pela taxa
            %de juros
            auxfoti=auxfoti+(pwr*pws*(1/((1+realr/100)^(1/2))))...
               *fotp1(t,ws,wr,ltottp1,ktp1);
            teste(v)=auxfoti;
            v=v+1;
```

```
                    end%ws
                end%wr

                if((t<=(h+1)) & (t>=(h+1-fmax))& estoqueauxtp1(t)...
                      <=(h+1-t)& estoqueauxtp1(t)>=0)
                   if(auxfoti<bestfoti(t))
                      bestfoti(t)=auxfoti;
                      bestfotiescolhido(t)=auxfoti;
                      bestwk(t)=wk;
                      bestwl(t)=wl;

                      estoque(t)=qtot;

                      estoquetp1(t)=qtottp1;
                      best_num_op_estoque=qk;
                   end
                else
                   if(t<(h+1-fmax))
                      if(auxfoti<bestfoti(t))
                         bestfoti(t)=auxfoti;
                         bestfotiescolhido(t)=auxfoti;
                         %variavel auxiliar para controlar a queda de bestfoti
                         bestwl(t)=wl;
                         bestwk(t)=wk;
                         estoque(t)=qtot;

                         estoquetp1(t)=qtottp1;
                         best_num_op_estoque=qk;
                      end
                   end%if
                end%if
             end%wl
          end%wk
          fot(t,ms,mr,l,k)=bestfoti(t);
          wlot(t)=bestwl(t);
          wkot(t)=bestwl(t);
       end%ms
    end%mr
 end%l
end%t

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Equacao de Bellman em t=1 (inicial)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

t=1;
tp1=t+1;
%instante ‘a frente do instante atual, t
```

```
instante(t)=t
%a cada loop em t, a funcao custo do instante t+1
%recebe o resultado da otimizacao do instante anterior:
dimRtp1=dimRt;
dimstp1=dimst;
for j=1:(opt+1)
   for l=1:(j*q+fmax+1)
      for Rind=1:2
         for sind=1:2
            fotp1(t,sind,Rind,l,j)=fot(t+1,sind,Rind,l,j);
         end%sind
      end%Rind
   end%l
end%k
dimRt=t;%redimensionamento
dimst=t;

%inicia varredura dos estados


%estados definidos pelas quantidades possiveis de salsichas:
for l=1:(fmax+1)
   %novamente, no matlab, nao posso partir de zero...entao somo 1.
   %Entenda-se que l=1 eh primeiro estado das quantidades de salsichas,
   %onde tenho quantidade zero.
   %Define-se ql como a quantidade, propriamente dita nesse estado
   ql=l-1;
   %qtot=q*qk+ql;
   %estados definidos pelas possibilidades de taxas de juros
   %no instante presente
   mr=1;
   realr=R(t,mr);
   %estados definidos pelas possibilidades de precos das salsichas
   %no mercado 'a vista
   ms=1;
   reals=S(t,ms);
   bestfoti(t)=infty;
   %limites para o n'umero de opcoes compradas
   if (qk<opt) wkmax=opt-qk; else wkmax=0; end
   for wk=0:wkmax
      %limites para compra no mercado fisico
      if(ql==0) wlmin=1; else wlmin=0; end
      if(ql<=(fmax)) wlmax=fmax-ql+1;else wlmax=0; end
      for wl=wlmin:wlmax
         %eq. evolucao dos estoques
         qltp1=ql+wl-1;
         estoqueauxtp1(t)=qtottp1;
         %variavel auxiliar para ser usada na decisao do melhor
```

```
            ltp1=qltp1+1;
            %variavel estado correspondente a quantidade qltp1
            qktp1=qk+wk;
            ktp1=qktp1+1;
            %variavel estado correspondente a quantidade qktp1
            %custo de controle
            auxfoti=wk*q*c0+wl*reals;
            %tratamento dos estados possiveis y=x(t+1)
            %wr e ws sao as transicoes possiveis de t para t+1
            for wr=mr:(mr+1)
               % possibilidade seguinte eh para cima ou para baixo
               %pwr=PR(t+1,wr)/PR(t,mr);
               if (wr==mr) pwr=pbd; else pwr=pbu; end
               for ws=ms:(ms+1)%idem
                  if (ws==ms) pws=psd; else pws=psu; end
                  %pws=PS(t+1,ws)/PS(t,ms);
                  %adicionamos ao custo do controle o custo futuro multiplicado
                  %pela probabilidade (tiramos a media, portanto)
                  %de transicao do estado presente para os possiveis
                  %estados futuros e descontamos pela taxa de juros
                  auxfoti=auxfoti+(pwr*pws*(1/((1+realr/100)^(1/2))))...
                     *fotp1(t,ws,wr,ltp1,ktp1);
               end%ws
            end%wr
            if(auxfoti<bestfoti(t))
               bestfoti(t)=auxfoti;
               bestfotiescolhido(t)=auxfoti;
               bestwl(t)=ql;
               bestwk(t)=wk;
               estoque(t)=ql;
               estoquetp1(t)=qltp1;
               best_num_op_estoque=qk;
            end
         end%wl
      end%wk
      fot(t,ms,mr,l,k)=bestfoti(t);
      %wlot(t,ms,mr,l,k)=bestwl(t);
      %wkot(t,ms,mr,l,k)=bestwk(t);
      wlot(t)=bestwl(t);
      wkot(t)=bestwk(t);
   end%l
end%k
```

E.2 simplex.m

```
function [Vo,b,xb,r] = driver(A,d,c)
% arquivo driver.m  driver e funcao simplex

  clc;
  disp('Criterio para a escolha do indice residual:');
  disp('1 - menor j tal que z(j) > 0');
  disp('2 - j tal que z(j) eh maximo');
  criterio = input('1/2? ');
  disp('Deseja ver cada iteracao em um arquivo?');
  trace = input('Digite 1 para sim ou 0 para nao: ');
  if (trace)
    disp('Os resultados serao gravados no arquivo simplex.sai')
    disp('Tecle algo para continuar: ');
    pause;
    diary simplex.sai;
  end

  clc;
  disp('PPL: ');
  disp('Matriz A: '); disp (A);
  disp('Vetor  d: '); disp (d');
  disp('Vetor  c: '); disp (c);
  disp('Criterio para a escolha de j: ');
  if (criterio == 1)
    disp('menor j tal que z(j) > 0');
  else
    disp('j tal que z(j) eh maximo');
  end
  disp('');

  [m,n] = size(A);
  aux = diag(sign(d));
  Aaux = [A aux];
  czeros = zeros(1,n);
  uns = ones(1,m);
  caux = [czeros uns];
  r = 1:n;
  b = n+1:n+m;
  [Vo,b,xb,r] = simplex(Aaux,d,caux,b,r,criterio,trace);
  if (Vo == 0)
    raux = find(r <= n);
    r = r(raux);
    [Vo,b,xb,r] = simplex(A, d, c, b, r, criterio, trace);
  else
    disp('Problema sem solucao');
```

```
  end %if
  if (trace)
    diary;
  end
end

function [Vo,b,x,r] = simplex(A,d,c,b,r,criterio,trace)
% arquivo simplex.m **********************************

  [m,n] = size(A);
  acabou = 0;
  while ~acabou
    B = A(:,b);
    R = A(:,r);
    if (det(B) == 0)
      disp('Erro: Det(B) = 0')
      return
    else
      Rtil = B\R;
      dtil = B\d;
      z = c(b)*Rtil - c(r);
      if ( any(z > 0) )
        indices = find(z > 0);
        if criterio == 1
          j = indices(1);
        else
          [aux,j] = max(z);   %maior declive
        end %if criterio
        k = find( Rtil(:,j) > 0 );
        epsilon = dtil(k) ./ Rtil(k,j);
        [aux,i] = min( epsilon );
        i = k(i);
        b = [b r(j)];
        r = [r b(i)];
        b(i) = [];
        r(j) = [];
      else % acabou
        acabou = 1;
      end %if indices
      x = zeros(n,1);
      x(b) = A(:,b)\d;
      Vo = c * x;
      if (trace)
        disp('');
        disp('Vo: '); disp(Vo);
        disp('indices da base: '); disp(b);
        disp('indices residuais: '); disp(r);
        disp('Vertice: '); disp(x'),
```

```
        disp('');
      end
    end % if det(B)
  end %while
end


function C = conta(l,m,n)
% todas as combinacoes de [l...m] n a n

C=[];
if( l>m | m-l+1<n )
  C=[];
elseif( n==1 )
  C=[l:m]';
else
  CA=[];
  CA= conta1(l,m-1,n-1);
  [car,cac]= size(CA);  %rows,columns in CA
  for j=1:car
    CAJ= CA(j,:);
    for k= (CAJ(cac)+1):m
      C=[C;[CAJ,k]];
    end
  end
end
```

E.3 mindist.m

```
% Este eh o arquivo MATLAB "mindist.m".
% Tudo q segue um "%" eh apenas comentario.
% Consulte o capitulo sobre a linguagem Matlab

n = 5;
% numero de vertices
C = [ 0  , 2  , 9E9, 1  , 2;
      2  , 0  , 2  , 9E9, 0;
      9E9, 2  , 0  , 3  , 4;
      1  , 9E9, 3  , 0  , 0;
      2  , 0  , 4  , 0  , 0;   ];
%matriz de custos
fo(1,1:n) = C(1,1:n);
%custo do caminho de tamanho 1
for t = 2:n-1;
  fo(t,1:n) = 9E9*ones(1,n);
  for j = 1:n
    for i = 1:n
      fo(t,j) = min( fo(t,j) , fo(t-1,i)+C(i,j)  );
    end
  end
end
fo,


% Este eh o arquivo MATLAB "dijk.m".
n = 5;
% numero de vertices
C = [ 0  , 2  , 9E9, 1  , 2;
      2  , 0  , 2  , 9E9, 0;
      9E9, 2  , 0  , 3  , 4;
      1  , 9E9, 3  , 0  , 0;
      2  , 0  , 4  , 0  , 0;   ];
%matriz de custos
inder = 2:n;
%vertices cuja minima distancia nao eh a correta
dcor = [1;0];  %distancias corretas
fo = C(1,2:n);
%custo do caminho de tamanho 1
for t = 2:n-1;
  [d,k] = min(fo);  %minimo da lista auxiliar
  i = inder(k);     %vertice correspondente
  dcor = [dcor,[i;d]];
  inder(k) = [];
  fo(k) = [];
  fo = min( fo , C(i,inder)+d*ones(1,n-t)  );
end
dcor = [dcor,[inder;fo]];
dcor,  %lista de vertices e distancias,
       %na ordem em que foram determinados
```

E.4 Fatoração QR

```
function rinv(dummy)

global A U b r
[m,n]=size(A);

%basis reinversion

for k=1:m
  U(:,k)= A(:,b(k));
end

for j=1:m-1
  pivj = U(j,j);
  for i=j+1:m

    %compute Givens rotation Q s.t. Q*[v1,v2]=[norm(2,v),0]
    %Q=[gcos, gsen; -gsen, gcos]
    %procedure is protective of overflow

    v1=pivj; v2=U(i,j);
    if( abs(v2)==0 )
      gcos=1; gsen=0;
    else
      if( abs(v1)>=abs(v2) )
        tau= -v2/v1;
        gcos= (1+tau*tau)^(-1/2);
        gsen= gcos*tau;
      else % abs(v2)>abs(v1)
        tau= -v1/v2;
        gsen= (1+tau*tau)^(-1/2);
        gcos= gsen*tau;
      end %else_abs
    end %else_v20

    % rotate rows U([j,i],j:m) by Q

    pivj= gcos*v1 -gsen*v2;
    U(i,j)=0;
   for k=j+1:m
      v1=U(j,k);
      v2=U(i,k);
      U(j,k)= gcos*v1 -gsen*v2;
      U(i,k)= gsen*v1 +gcos*v2;
    end %fork

  end %fori
U(j,j)=pivj;
end %forj

%*******************************************************
```

```
function update(ip,jp)

global A U b r
[m,n]=size(A);

% r(jp) enters the basis,
% b(ip) exits the basis.

a= A(:,r(jp));
y=qtm(a);
qrup(y,ip);
auxi=b(ip);
for k=ip:m-1
  b(k)=b(k+1);
end
b(m)=r(jp);
r(jp)=auxi;

%*****************************


function  qrup(x,k)

global A U b r
[m,n]= size(A);

%file qrup.m
%Updates the U factor from the Hessemberg matrix obtained
%by deleting column k of U,
%and appending column x as the last column of U.

EPSLN=1E-16;

for i=1:k-1
  for j=k+1:m
    U(i,j-1)=U(i,j);
  end
  U(i,m)=x(i,1);
end


for i=k:m-1

  %compute Givens rotation Q s.t. Q*[v1,v2]=[norm(2,v),0]
  %Q=[gcos, gsen; -gsen, gcos]
  %procedure is protective of overflow

  v1=U(i,i+1); v2=U(i+1,i+1);
  if( abs(v2)<EPSLN )
    gcos=1; gsen=0;
  else
```

```
    if( abs(v1)>=abs(v2) )
      tau= -v2/v1;
      gcos= (1+tau*tau)^(-1/2);
      gsen= gcos*tau;
    else % abs(v2)>abs(v1)
      tau= -v1/v2;
      gsen= (1+tau*tau)^(-1/2);
      gcos= gsen*tau;
    end %else_abs
   end %else_v20

  % rotate rows U(i:i+1,:) by Q


  U(i,i)= gcos*v1 -gsen*v2;
  for j=i+2:m
    v1=U(i,j);
    v2=U(i+1,j);
    U(i,j-1)= gcos*v1 -gsen*v2;
    U(i+1,j)= gsen*v1 +gcos*v2;
  end

  %rotate elements x(i:i+1)
  v1=x(i,1);
  v2=x(i+1,1);
  U(i,m)= gcos*v1 -gsen*v2;
  x(i+1,1)= gsen*v1 +gcos*v2;
end
U(m,m)=x(m);

%*********************************************************


function y = qtm(x)

global A U b r
[m,n]=size(A);

%file qtm.m y =Q'*x =inv(U')*B'*x  , B=QR

y=btm(x);
y=utim(y);

%***************************************


function y = uim(x)

global A U b r
[m,n]=size(A);

%file uim.m    y=inv(U)*x
```

```
y(m,1)=x(m,1)/U(m,m);
for i=m-1:-1:1
  y(i,1)=( x(i,1) - U(i,i+1:m)*y(i+1:m,1) )/U(i,i);
end

%************************************************


function y = utim(x)

global A U b r
[m,n]=size(A)

%file utim.m  x=inv(U')*y

y=x;
for i=1:m-1
  y(i)=y(i)/U(i,i);
  y(i+1:m) = y(i+1:m,1) - y(i,1)*U(i,i+1:m)';
end

y(m)=y(m)/U(m,m);

%*****************************************


function y = btim(x)

global A U b r

%file btim.m  y = inv(B')*x = B*inv(U)*inv(U')*x

y=utim(x);
y=uim(y);
y=bm(y);

%*************************************************


function y = btm(x)

global A U b r
[m,n]= size(A);

%file btm.m   y=B'*x

for i=1:m
  a=A(:,b(i));
  y(i,1)=a'*x;
end

%*****************************************************
```

```
function  y = bim(x)

global A U b r

% file bim.m y= inv(B)*x =inv(U)*inv(U')*B'*x , B=QU

y=btm(x);
y=utim(y);
y=uim(y);

%*************************************************


function y = bm(x)

global A U b r
[m,n]= size(A);

%file bm.m   y=B*x

y=zeros(m,1);
for j=1:m
  a=A(:,b(j));
  y=y+x(j)*a;
end

%*************************************************
```

E.5 depvital.m

```
function fus= rank2(a1,a2)

% F(t) is the components cumulative
% life probability distribution
% F(t) = Pr(l<=t)
% Its complement is the survival probability distribution
% Fc(t) = 1-F(t) = Pr(l>t)
% The failure probability at the next
% period x given the survival up to
% current time t is
% F(x|t) = (F(t+x)-F(t))/Fc(t)
% = 1 -Fc(x|t)
% The failure rate, hazard rate or
% force of mortality at age t is
% h(t) = f(t)/Fc(t)
% Integrating
% I[0:x] h(t)dt = -log(Fc(x))
% Fc(x) = exp(-H(x))
% H(x) = I[0:x] h(t)dt

% A(:,1)= age
% A(:,2)= h(t)

nx=100;
%maximum age at life table

% generates test assuring h(nx)==1;
% a= 1:nx; h= (1/nx)*a; h=h.^5;  %plot(h);

% f= life density; h= haz.rate;
% a= age; c=complement; u=cumulative
aux=0;
for i=1:nx
  aux= aux +h(i);
  hu(i)= aux;
  fuc(i)= exp(-hu(i));
  fu(i)= 1-fuc(i);
end

% 2 lifelong dependents
% ak= current age of k-th depend
% Xk= surviv. of k-th depend.
% R2= sup{X1,X2} R1= inf{X1,X2}
% Pr(R2<=t|a1,a2).
% = Pr(X1<=t|a1 and X2<=t|a2)
% Pr(R1<=t|a1,a2)
% = Pr(X1<=t|a1 or X2<=t|a2)
% Pr(R1>t|a1,a2)
% = Pr(X1>t|a1 and X2>t|a2)

for t=1:100
  if( (a1+t)>nx )
    fua1(t)=1;
  else      %Pr(X1<=t|a1)
```

```
    fua1(t) = ...
    ((fu(a1+t)-fu(a1))/fuc(a1));
  end
  if( (a2+t)>nx )
    fua2(t)=1;
  else
    fua2(t) = ...
    ((fu(a2+t)-fu(a2))/fuc(a2));
  end
  f2u(t)= fua1(t)*fua2(t);
  f1u(t)= fua1(t) +fua2(t) -f1u(t);
end
fus=[fu;f2u;f1u;fua1;fua2];
plot(a',fu','--b',a',f2u,'-r',a',f1u,
..'-r',a',fua1','--k',a',fua2','--k');
title( ...
..['Order statistics for survival', ...
  int2str(a1),' and ',int2str(a2)]);

%************************************************
```

E.6 GRG

```
% Gradiente Reduzido Generalizado
%
% Eduardo Oda   BMACC  2004
%
% Fazem parte desse pacote duas funcoes:
% 1. grg.m
%   Implementa o metodo em questao
%
% 2. buscalinear.m
%   Implementa uma busca linear simples
%
% Adicionalmente, seguem diretorios nomeados teste#,
% onde '#' eh um numero, com exemplos para teste.
%
% Para utilizar o exemplo, faca:
%
%   $ cp teste1/* .
%   $ octave teste.m
%
% Para definir um problema diferente, edite os arquivos de um dos
% exemplos, copie-os para o mesmo diretorio dos arquivos principais
% (grg.m e buscalinear.m) e execute:
%
%  $ octave teste.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%  Esta eh uma funcao em octave versao 2.0.16.92 que encontra um
%  ponto x do R^n que seja uma solucao otima do problema
%  min  f(x)
%  s.a.  h(x)=0
%    l<=x<=u
%
%  com o metodo dos gradientes reduzidos generalizados (GRG),
%  recebendo como parametro:
%    - f(x) , f:R^n->R
%    - h(x), h:R^n->R^m
%    - l (n x 1)
%    - u (n x 1)
%
%  Adicionalmente a funcao recebe a variavel trace, que controla a
%  exibicao das iteracoes, mostrando cada iteracao se valer 1 ou
%  ocultando-as se valer 0.
%
%  As funcoes f e h, bem como seu gradiente (gradf) e jacobiano (jach),
%  respectivamente, devem ser definidas no octave. Preferencialmente devem
%  ser criados quatro arquivos no mesmo diretorio desta funcao:
%  a) f.m
```

```
%    define uma funcao que recebe um vetor x de dimensao [nx1] e
%    retorna o valor da funcao f calculada nesse ponto, um real.
%
%  b) gradf.m
%    define uma funcao que recebe um vetor x de dimensao [nx1] e
%    retorna o valor do gradiente da funcao f calculado nesse ponto,
%    um vetor de dimensao [nx1].
%
%  c) h.m
%    define uma funcao que recebe um vetor x de dimensao [nx1] e
%    retorna o valor da funcao h calculada nesse ponto, um vetor de
%    dimensao [mx1].
%
%  c) jach.m
%    define uma funcao que recebe um vetor x de dimensao [nx1] e
%    retorna o valor do jacobiano da funcao h calculado nesse ponto,
%    uma matriz de dimensao [mxn].
%
%  Junto com esta funcao seguem exemplos desses arquivos, use-os como base
%  para suas funcoes.
%
%  O retorno desta funcao eh um vetor (1 x 2) cujas componentes sao, na
%  ordem, o x otimo e o valor otimo.
%
%  Para utilizar essa funcao, inicie o octave, defina todos os parametros
%  descritos acima e execute o comando:
%
%  octave:> [x_otimo,v_otimo]=grg(x,l,u,trace)

function[x_otimo,v_otimo]=grg(x,l,u,trace)

  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  % Inicializando as variaveis %
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

  EPS=0.00000001;
  Jx=jach(x);
  [m,n]=size(Jx);
  count=1;

  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  % Implementacao do metodo GRG %
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

  while 1
    gradfx=gradf(x);

    % Procuramos uma matriz nao singular para ser base, dando
    % preferencia a indices cujas variaveis estejam mais distantes
    % dos bordos da caixa

    dif=abs(x-(u-l)/2);
```

```
    [odif i_dif]=sort(dif);
    i_base=[i_dif(1:m)];
    i_r=[i_dif(m+1:n)];
    i=m;  % indice do indice basico a ser trocado
    j=1;  % indice do indice residual a ser trocado
    while(abs(det(Jx(:,i_base)))<EPS)
      disp("");
      %desfaz
      if(i+1>m && j>1)
        aux=i_base(1);
        i_base(1)=i_r(j-1);
        i_r(j-1)=aux;
        % nao faz nada
      elseif(i+1<=m)
        aux=i_base(i+1);
        i_base(i+1)=i_r(j);
        i_r(j)=aux;
      end
      aux=i_base(i);
      i_base(i)=i_r(j);
      i_r(j)=aux;
      i=i-1;
      if(i==0)
        i=m;
        j=j+1;
        if(j>n-m)
          disp("Nao foi possivel encontrar uma base boa")
          break;
        end
      end
    end

    % Calculamos o vetor de gradientes reduzidos
%    z=gradfx(i_r)'-gradfx(i_base)'*inv(Jx(:,i_base))*Jx(:,i_r);
    z=gradfx(i_r)-gradfx(i_base)*inv(Jx(:,i_base))*Jx(:,i_r);

    % Definimos uma direcao para andar com a solucao viavel atual
    v=zeros(n,1);
    i_neg=find(z<-EPS);
    if(size(i_neg)>0)
      i=find(x(i_r(i_neg))<u(i_r(i_neg)));
      v(i_r(i_neg(i)))=-z(i_neg(i))*diag(u(i_r(i_neg(i)))-x(i_r(i_neg(i))));
    end
    i_pos=find(z>EPS);
    if(size(i_pos)>0)
      i=find(x(i_r(i_pos))>l(i_r(i_pos)));
      v(i_r(i_pos(i)))=-z(i_pos(i))*diag(x(i_r(i_pos(i)))-l(i_r(i_pos(i))));
    end
    v(i_base)=-inv(Jx(:,i_base))*Jx(:,i_r)*v(i_r);
    v(find(abs(v)<EPS))=0;

    % Se a direcao for toda nula, o metodo termina, pois nao
```

```
    % podemos caminhar numa direcao que melhore a funcao objetivo
    if(abs(v)<EPS)
      break;
    end

    % Descobrimos o quanto devemos (e podemos) andar na direcao v
    eta(1)=0;
    e3n=inf;
    e3p=inf;
    i_neg=find(v<-EPS);
    if(size(i_neg)>0)
      e3n=min( (l(i_neg)-x(i_neg))./v(i_neg) );
    end
    i_pos=find(v>EPS);
    if(size(i_pos)>0)
      e3p=min( (u(i_pos)-x(i_pos))./v(i_pos) );
    end
    eta(3)=min(e3n,e3p);
    eta(2)=eta(3)/2;
    eta=buscalinear(eta,x,v,EPS);
    eta=eta(2);

    if(trace)
      printf("%%%%%%% Iteracao %d %%%%%%%%%% \n",count);
      count++;
      disp("Ponto atual:")
      disp(x')
      disp("Indices da base:")
      disp(i_base')
      disp("Gradiente reduzido:")
      disp(v')
      disp("Eta:")
      disp(eta)
    end

    % Retornamos para a restricao h(x)=0 utilizando Newton-Raphson
    while eta>EPS
      x2=x+eta*v;
      Jx2=jach(x2);

      % Tentamos retornar para a restricao em, no maximo,
      % seis iteracoes do Newton-Raphson.
      % Segundo Lasdon, a experiencia mostra que chamadas
      % subsequentes ao Newton-Raphson geralmente nao
      % convergem.
      % Observe que isso nao interfere na generalidade do
      % metodo.
      for j=1:6
%        Dx=-inv(Jx2(:,i_base))*h(x2);
        Dx=-inv(Jx2(:,i_base))*h(x2)';
        if(abs(Dx)<EPS)
          break;
```

```
        end
        aux=x2(i_base)+Dx;
        % Verificamos se alguma restricao de caixa
        % nao seria honrada no retorno a restricao
        % h(x)=0
        if(any(aux<l(i_base)) || any(aux>u(i_base)))
          break;
        else
          x2(i_base)=aux;
          Jx2=jach(x2);
        end
      end
      % Se nao foi possivel retornar a restricao h(x)=0
      % diminuimos o tamanho do passo e fazemos uma nova
      % tentativa
      if(any(abs(Dx)>EPS))
        eta=eta/2;
      else
        x=x2;
        Jx=Jx2;
        break;
      end
    end

    % O tamanho do passo igual a zero indica que nao podemos
    % melhorar a funcao objetivo caminhando nessa direcao,
    % entao o metodo termina
    if(eta<EPS)
      break;
    end
    x(find(abs(x)<EPS))=0;

  end
  x_otimo=x;
  v_otimo=f(x_otimo);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%  Esta eh uma funcao em octave versao 2.0.16.92 que minimiza f(x+e*v),
%  com x e v fixos, fazendo uma busca linear.
%
%  Recebe como parametro os vetores x e v, ambos de dimensao [nx1], o
%  vetor eta, de dimensao [3x1], e um valor, abaixo do qual um numero
%  em modulo eh considerado zero.
%
%  O vetor eta deve ser da forma eta(1)<=eta(2)<=eta(3).
%
%  Retorna o vetor eta de forma que min(f(x+e*v))=f(x+eta(2)*v)

function eta=buscalinear(eta,x,v,EPS)
  vf=[
```

```
  f(x+eta(1)*v);
  f(x+eta(2)*v);
  f(x+eta(3)*v);
];
sf=vf(1)+vf(2)+vf(3);

while 1
  if(vf(2)-vf(1)<=EPS)
    if(vf(2)-vf(3)<=EPS)
      if(abs(vf(2)-vf(1))<EPS && abs(vf(2)-vf(3))<EPS)
        eta(2)=eta(3);
        break;
      end
      numerador=(
        vf(1)*(eta(3)^2-eta(2)^2)+
        vf(2)*(eta(1)^2-eta(3)^2)+
        vf(3)*(eta(2)^2-eta(1)^2)
        );
      denominador=(
        vf(1)*(eta(3)-eta(2))+
        vf(2)*(eta(1)-eta(3))+
        vf(3)*(eta(2)-eta(1))
        );
      eta4=numerador/(2*denominador);
      if(abs(eta4-eta(2))<EPS)
        break;
      end
      vf4=f(x+eta4*v);
      if(eta4<eta(2))
        eta(3)=eta(2);
        vf(3)=vf(2);
        eta(2)=eta4;
        vf(2)=vf4;
      else
        eta(1)=eta(2);
        vf(1)=vf(2);
        eta(2)=eta4;
        vf(2)=vf4;
      end
    else
      eta4=(eta(3)+eta(2))/2;
      if(abs(eta4-eta(2))<EPS)
        break;
      end
      vf4=f(x+eta4*v);
      eta(1)=eta(2);
      vf(1)=vf(2);
      eta(2)=eta4;
      vf(2)=vf4;
    end
  else
    eta4=(eta(1)+eta(2))/2;
```

```
      if(abs(eta4-eta(2))<EPS)
        break;
      end
      vf4=f(x+eta4*v);
      eta(3)=eta(2);
      vf(3)=vf(2);
      eta(2)=eta4;
      vf(2)=vf4;
    end
    sfn=vf(1)+vf(2)+vf(3);
    if((sf-sfn)<EPS)
      break;
    else
      sf=sfn;
    end
  end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function y=f(x)
  y=-2*x(1)-4*x(2)+x(1)^2+x(2)^2+5;
end

function y=gradf(x)
  y(1)=-2+2*x(1);
  y(2)=-4+2*x(2);
  y(3)=0;
  y(4)=0;
end

function y=h(x)
  y(1)=-x(1)+2*x(2)+x(3)-2;
  y(2)=x(1)+x(2)+x(4)-4;
end

function y=jach(x)
  y=[
  -1 2 1 0;
   1 1 0 1
  ];
end

l=[0;  0;  0;  0];
x=[0;  0;  2;  4];
u=[10;  10;  10;  10];

[x_otimo,v_otimo]=grg(x,l,u,0)
```

Bibliografia

[Ait 91] M Aitkin (1991) Posterior Bayes Factors. *J R Statist Soc B* 1:111-142

[Alb 93] U.N.de Alba J.V.Asensio. *Matemtica Actuarial.* Instituto de Ciencias del Seguro, Editorial MAPFRE, Madrid, 1993.

[Ale 86] G.J.Alexander and J.C.Francis. *Portfolio Analysis.* Prentice-Hall, Englewood Cliffs, New Jersey, 1986.

[Ale 88] S.T.Alexander C.T.Pan R.J.Plemmons. *Analysis of a Recursive Least Squares Hyperbolic Rotation Algorithm for Signal Processing.* Linear Algebra and its Applications, 98, pp-3-40, 1988.

[Azo 94] E. M. Azoff. *Neural network time series forecasting of financial markets.* John Wiley, 1994.

[Bae 94] D. W. D. Baestaens and W. M. V. D. Bergh. *Neural network solutions for trading in financial markets.* Pitman, 1994.

[Bar 84] A.R.Barron. *Predicted Squared Error: A Criterion for Automatic Model Selection.* in [Far 84].

[Baz 90] M.Bazaraa J.J.Jarvis H.D.Sherali. Linear Programming and Network Flows. Wiley 1990.

[Baz 93] M.Bazaraa H.D.Sherali C.M.Shetty. Nonlinear Programming. Wiley, 1993.

[Bec 64] E.F.Beckenbach. Applid Combinatorial mathematics. Wiley, 1964.

[Bel 96] A.Beltratti S.Margarita P.Terna. Neural Networks and Financial Modeling. Thomson, 1996.

[Ber 94] J.Beran Statistics for Long Memory Processes. Chapman Hall, 1994.

[Ber 76] D.P.Bertsekas. *Dynamic Programming and Stochastic Control.* Academic Press, 1976.

[Ber 89] D.P.Bertsekas J.N.Tsitsiklis. paralel and Distributed Computation: Numerical Methods. Prentice hall, 1984.

[Ber 87] JO Berger M Delampady (1987) Testing precise hypothesis. *Statistical Science* 3:315-352

[Ber 97] JO Berger, B Boukai, Y Wang (1997) Unified frequentist and Bayesian testing of a precise hypothesis. *Statistical Science* 3:315-352

[Beu 94] A.Beutelspacher. Cryptology.

[Bic 77] P.J.Bickel K.A.Doksum. mathematical Statistics. Holden-Day, 1997.

[Bil 86] P.Bilingsley. *Probability and Measure*. Wiley-Interscience, 1986.

[Bir 97] J.Birge F.Louveaux. Introduction to Stochastic Programming. Sringer, 1997

[Bor 92] K.H.Borch. *Economics of Insurance*. North-Holland, 1992.

[Bow 97] N.L.Bowers Jr. H.U.Gerber J.C.Hickman D.A.Jones C.J.Nesbitt. *Actuarial Mathematics*. The Society of Actuaries, USA, 1997.

[Bra 87] P.Bratley, B.L.Fox, L.Schrage. A Guide to Simulation. Springer-Verlag, 1987.

[Bre 84] L.Breiman, J.H.Friedman, C.J.Stone. "Classification and Regression Trees." Chapman Hall, London, 1984.

[Bro 91] P.J.Brockwell and R.A.Davis. *Time Series: Theory and Models.* Springer, 1991.

[Bue 64] R.J.Buehler, B.V.Shah, O.Kempthorne. Iowa State Univ. Statist. Lab. Tech. Repts. for the Office of Naval Res. 1) Some Problems of Steepest Ascent and Related Procedures for Finding Optimum Conditions. April 1961. 2) Methods of Parallel Tangents (PARTAN). April 1961, rev. August 1962. 3) Some Further Properties of the Methods of Parallel Tangents and Conjugate Gradients. September, 1961.

[Cab 96] A.N.Cabot. Cassino Gamming: Policy, Economics and Regulation. UNLV International Gaming Institute, 1996.

[Che 98] V.Cherkassky F.Mulier. Learning from Data: Concepts, Theory, and Methods.

[Chi 95] J.Y.Ching, A.K.C.Wong, K.C.C.Chan. "Class-Dependent Discretization for Inductive Learning from Continuous and Mixed-Mode Data." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17 n.7, pp.641-651, 1995.

[Cho 83] G.Chow. Econometrics. McGraw-Hill, 1983.

[Chr 87] R.Christensen. *Plane Answers to Complex Questions - The Theory of Linear Models.* Springer, 1987.

[Chv 83] V.Chvátal. *Linear Programming.* W.H.Freeman, 1983.

[Clo 97] W.F.Clocksin. Clause and Effect: Prolog for the Working Programmer. Springer, 1997.

[Coh 86] K.J.Cohen et all. *The Microstructure of Securities Markets.* Prentice-Hall, Englewood Cliffs, New Jersey, 1986.

[Coh 67] K.J.Cohen J.A.Pogue. *An Empirical Evaluation of Alternative Portfolio Selection Models.* Journal of Business, 1967.

[Col88] R.W.Colby, A.Mayers. "The Encyclopedia of Technical Market Indicators." Dow Jones - Irwin, Homewood, Illinois, 1988.

[Col 90] T.F.Coleman and C.F.van Loan. *A Matrix Computation Handbook.* SIAM Publications, Philadelphia.

[Cox 77] DR Cox (1977) The role of significance tests. *Scand J Statist* 4:49-70

[Cra 45] H.Cramer. Mathematical Methods of Statistics. Princeton, 1945.

[Cut 91] T.van Cutsem. "Decision Trees for Detecting Emergency Voltage Conditions." *Proc. Second International Workshop on Bulk Power System Voltage Phenomena*, pp.229-240, McHenry, USA, 1991.

[Cyb 89] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control Signals Systems,* 2:303-314, 1989.

[Dat 88] B.N.Datta et all (Edts.) *Linear Algebra in Signal, Systems and Control.* Siam, Philadelphia, 1988.

[Dav 69] F.N.David. Games Gods and Gambling: A History of Probability and Statistical Ideas. Griffin, 1969.

[Dav 85] M.H.A.Davis R.B.Vinter. *Stochastic Modelling and Control.* Chapman and Hall, London, 1985.

[Day 94] C.D.Daykin T.Pentikinen M.Pesonen. *Practical Risk Theory for Actuaries.* Chapman & Hall, 1994.

[DeF 91] B. de Finetti. *Scritti.* Pitagora Editrice, Boligna, 1991.

[DeG 86] M. DeGroot, Probability and Statistics, Addison Wesley 1986.

[Den 82] E.Denardo. Dynamic Programming. Prentice Hall, 1982.

[Dia 96] K.I.Diamataras S.Y.Kung. Principal Component Neural Networks. Wiley, 1996.

[Dix 92] P.B.Dixon, B.R.Parmenter and A.A.Powell. *Notes and Problems in Applied General Equilibrium Economics*. North-Holland, Amsterdam, 1992.

[Don 00] J.J.Dongarra, J.R.Bunch, C.B.Moler, G.W.Stewart. *LINPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia.

[Dop 96] K.Dopfer. The Global Dimension of Economic Evolution. Springer, 1996.

[Eat 00] J.W.Eaton. *Octave Manual*. Chemical Engineering Department, University of Texas at Austin. Austin, Texas.

[Elt 91] E.J.Elton M.J.Gruber. Modern Portfolio and Investment Analysis. Wiley, 1991.

[Esp 76] B. d'Espagnat. Conceptual Fundations of Quantum Mechanics. W.A.Benjamin, 1976.

[Far 84] S.J.Farlow (editor). Self-Organizing Methods in Modeling. Marcel Deckker, 1984.

[Far 97] J.L.Farrell. Portfolio Management. McGraw-Hill, 1997.

[Fis 96] G.S.Fishman. Monte Carlo Methods. Springer-Verlag, 1996.

[Fle 87] R Fletcher (1987) *Practical Methods of Optimization*. Essex: J Wiley, Pp 436

[Flo 94] R.P.Flood P.M.Garber. Speculative Bubbles, Speculative Attacks, and Policy Switching. MIT Press. 1994.

[Fu 94] L. Fu. *Neural networks in computer intelligence*. McGraw-Hill, 1994.

[Gar 00] B.S.Garbow, J.M.Boyle, J.J.Dongarra, C.B.Moler. *Matrix Eigensystem Routines: EISPACK Guide Extension*. Lecture Notes in Computer Science, volume 51, Springer-Verlag.

[Gol 89] G.H.Golub and C.F.van Loan. *Matrix Computations*. John Hopkins, 1989.

[Gom 99] C.Gomez. Engineering and Scientific Computing with Scilab. Birkhäuser, 1999.

[Gol 96] A.Golan G.Judge D.Miller. Maximum Entropy Econometrics. Wiley, 1996.

[Goo 83] IJ Good (1983) *Good thinking: The foundations of probability and its applications*. University of Minnesota Press. Pp 332

[Goo 84] G.C.Goodwin and K.S.Sin. *Adaptive Filtering Prediction and Control* Prentice-Hall, 1984.

[Goo 99] P.I.Good. *Resapling Methods: A Practical guide to Data Analysis*. Birkhauser, 1999.

[Had 64] G. Hadley. Nonlinear and Dynamic Programming. Addison-Wesley, 1964.

[Han 98] D.J.Hand S.D.Jacka. Statistics in Finance. Arnold, 1998.

[Har 81] A.C.Harvey *Time Series Models.* Philip Allan, New York, 1981.

[Har 90] A.C.Harvey. *Forecasting, Structural Time Series, and the Kalman Filter.* Cambridge Univ. Press, Cambridge, 1990.

[Hea 91] M.J.R.Healy. *Matrices for Statistics.* Oxford Science Publications, 1991.

[Hjo 94] J. S. U. Hjorth. *Computer intensive statistical methods.* Chapman and Hall, 1994.

[Hoc 85] R.R.Hocking. *The Analysis of Linear Models.* Brooks-Cole, 1985.

[Hon 94] J.Honerkamp. Stochastic Dynamical Systems: Concepts, Numerical Methods, Data Analysis. VCH Verlagsgesellschaft, 1994.

[Hor 95] R Horst, PM Pardalos NV Thoai (1995) *Introduction to Global Optimization.* Boston: Kluwer Academic Publishers.

[Hul 91] J. Hull. *Introduction to futures and options markets.* Prentice-Hall, 1991.

[Inf 94] G.Infanger. Planing Under Uncertainty: Solving Large Scale Stochastic Linear Programs. Boyd Fraser, 1994

[Ing 87] J.E.Ingersoll. *Theory of Financial Decision Making.* Studies in Financial Economics, Rowman and Littlefield, Savage, Maryland, 1987.

[Iro 86] TZ Irony CAB Pereira (1986) Exact test for equality of two proportions: Fisher×Bayes. *J Statist Comp Simulation* 25:93-114

[Iro 95] TZ Irony CAB Pereira (1986) Bayesian Hypothesis test: Using surface integrals to distribute prior information among hypotheses. *Resenhas* 2:27-46

[Jaz 70] A.H.Jazwinski. *Stochastic Processes and Filtering Theory.* Academic Press, New York, 1970.

[Kal 94] P.Kall S.Wallace. *Stochastic Programming.* Wiley, 1994.

[Kar 93] T.Kariya. Quantitative Methods for Portfolio Analysis. Kluwer, 1993.

[Ker 88] B.W.Kernighan D.M.Rittchie. The C Programming Language (ANSI-C). Prentice Hall, 1988.

[Knu 96] D.E. Knuth. The Art of Computer Programming, vol 2 - Seminumerical Algorithms, Addison Wesley, 1996.

[Koe 89] A.Koenig. C Traps and Pitfalls. AT&T Bell Lab, 1989.

[Kro 98] AR Krommer CW Ueberhuber (1998) *Computational Integration*. Philadelphia: SIAM, Pp 445

[Kum 86] P.R.Kumar and P.Varaiya. *Stochastic Systems: Estimation, Identification, and Adaptive Control.* Prentice-Hall, 1986.

[Las 70] L.S.Lasdon. *Optimization Theory for Large Systems.* MacMillan, 1970.

[Law 76] E.L.Lawler. *Cominatorial Optimization: Networks and Matroids.* Holt Rinehart and Winston, 1976.

[Law 82] A.M.Law W.D.Kelton. Simulation Modeling and Analysis. McGraw-Hill, 1982.

[LEc 88] P. L'Ecuyer. Efficient and Portable Combined Pseudo-Random Number Generators. Commun. ACM, 1988.

[Lei 93] J.G. Leite, C.A.B. Pereira, F.W. Rodrigues. Waiting Time to Exaust Lottery Numbers. Commun. in Statist. 22, pp. 301-310, 1993.

[Lep 77] G.P.Lepage. A New Algorithm for Adaptive Multidimencional Integration. Stanford Linear Accelerator Center, SLAC-PUB-1839, 1977.

[Lew 89] P.A.W.Lewis, E.J.Orlav. Simulation Methodology. Wadsworth and Brooks Cole, 1989.

[Lin 57] DV Lindley (1957) A statistical paradox. *Biometrika* 44:187-192

[Lin 78] DV Lindley (1978) The Bayesian approach. *Scand J Statist* 5:1-26

[Lue 84] D.G.Luenberger. *Linear and Nonlinear Programming*. Addison Wesley, 1984.

[Lue 98] D.G.Luenberger. *Investment Science*. Oxford, 1998.

[Mar 52] H.M.Markowitz. *Portfolio Selection*. The Jounal of Finance, 7(1), pp-77-91, 1952.

[Mar 56] H.M.Markowitz. *The optimization of a Quadratic Function Subject to Linear Constraints.* Naval Research Logistics Quarterly, 3, 111-133.

[Mar 87] H.M.Markowitz. *Mean-variance Analisys in Portfolio Choice and Capital Markets*. Basil Blackwell, Cambridge, Massachusetts, 1987.

[Mar 72] A Marshall F Prochan (1972) Classes of distributions applicable in replacement, with renewal theory implications. *Proc. 6th Berkeley Symp. Math. Statist. Prob.* pp 395-415.

[Mar 99] R.K.Martin. *Large Scale Linear and Integer Programming.* Kluwer, 1999.

[McC 83] G.P.McCormick. *Nonlinear Programming.* Willey, 1983.

[Mer 90] R.C.Merton. Continous Time Finance. Blackwell, 1990.

[Mic 94] D.Michie, D.J.Spiegelhalter, C.C.Taylor. "Machine Learning, Neural and Statistical Classification." Ellis Horwood, New York, 1994.

[Mic 98] R.O.Michaud. *Efficient Asset management.* Harvard Business School Press, 1998.

[Miz 94] P.Mizen. Buffer Stock Models and the Demand for Money. St.Martin 1994.

[Mol 81] C.B.Moler. *MATLAB Manual.* Department of Computer Science, University of New Mexico, 1981.

[Mol 00] C.B.Moler, J.N.Litte and S.Bangert. *PC-Matlab User's Guide.* The MathWorks Inc. Sherborn, Massachusetts.

[Mon 98] LE Montoya-Delgado, TZ Irony, CAB Pereira M Whittle (1998) Unconditional exact test for the Hardy-Weinberg law. *Submitted for publication*

[Moo 74] A.M.Mood F.A.Graybill D.C.Boes. Introduction to the Theory of Statistics. McGraw-Hill, 1974.

[Moo 78] D.S.Moore. Chi-Square Tests. in R.V.Hogg (editor) Studies in Statistics. MAA, 1978.

[Moo 85] R.E.Moore. Computational Functional Analysis. Ellis Horwood, 1985.

[Mue 94] W.Mueller, F.Wysotzki. "Automatic Construction of Decision Trees for Classification." *Annals of Operations Research* 52, pp.231-247, 1994.

[Mur 96] J. J. Murphy. *Technical analisys of the futures markets.* Prentice-Hall, 1996.

[Mur 81] B.A.Murtagh. Advanced Linear Programming. McGraw Hill, 1981.

[Nem 89] GL Nemhauser, AHG Rinnooy Kan MJ Todd editors (1989) *Optimization, Handbooks in Operations Research Vol 1.* Amsterdam: North-Holland, Pp 709

[New 97] D.Newton. Encyclopedia of Cryptology. Abc-Clio, 1997.

[Nil 90] R. H. Nielsen. *Neurocomputing.* Addison-Wesley, 1990.

[Pai 77] C.C.Paige M.A.Saunders. *Least Squares Estimation of Discrete Linear Dynamic Systems using Orthogonal Transformations.* Siam J. Numer. Anal. 14 (2), pp-180-193, 1977.

[Pan 92] H.H.Panjer G.E.Willmot. *Insurance Risk Models*. Society of Actuaries, 1992.

[Pas 90] R.C.Pascual. *Previsin Tecnolgica de la Demanda*. Marcombo Boixareu, Barcelona, 1990.

[Per 84] CAB Pereira A Rogatko (1984) The Hardy-Weinberg equilibrium under a Bayesian perspective. *Braz J Genet* 7:689-707

[Per 87] CAB Pereira DV Lindley (1987) Examples questioning the use of partial likelihood. *The Statistician,* 36:15-20

[Per 93] C.A.B. Pereira, S. Wechsler. On the concept of P-Value. Brazilian Journal of Probability and Statistics 7, pp. 159-177, 1993.

[Per 99a] C.A.B.Pereira J.M.Stern. *A Dynamic Software Certification and Verification Procedure.* Proc. ISAS-99 - International Conference on Information Systems Analysis and Syntesis. V-II, pp. 426-435, 1999.

[Per 99b] C.A.B.Pereira J.M.Stern. *Evidence and Credibility: Full bayesian Significance Test for Precise Hypoteses.* Entropy, V.1, pp.69-80, 1999.

[Pet 91] E.E.Peter. Chaos and Order in the Capital Markets.

[Pfl 96] G.C.Pflug. Optimization of Stochastic Models. Kluwer 1996.

[Pet 94] E.E.Peter. Fractal Market Analysys.

[Phi 95] L.Philips. Competition Policy: A Game-Theoretic Perspective. Cambridge, 1995.

[Pin 96] JD Pintér. *Global Optimization in Action. Continous and Lipschitz Optimization: Algorithms, Implementations ans Applications.* Boston: Kluwer Academic Publishers, 1996.

[Pir 93] M. J. S. S. Piramuthu and C. Kuan. Learning algorithms for neural-net decision support. *ORSA Journal on Computing,* 5:381-373, 1993.

[Pli 99] S.Pliska. Introduction to Mathematical Finance: Discrete Time Models. Blackwell, 1999.

[Qui 86] J.R.Quinlan. "Induction of Decision Trees." *Machine Learning* 1, pp.221-234, 1986.

[Rho 71] I.B.Rhodes. *A Tutorial Introduction to Estimation and Filtering.* IEEE Trans. on Autom. Contr., 16, (6), pp-688-707, 1971.

[Rip 87] B.D.Ripley. Stochastic Simulation. John Wiley, 1987.

[Rit 96] P.Ritchen. Derivative Markets: Theory, Strategy, and Aplications. Harper Colins, 1996.

[Ros 97] S.M.Ross. *Probability Models*. Academic Press, 1997.

[Roy 97] R Royall (1997) *Statistical Evidence: A Likelihood Paradigm*. London: Chapman Hall, Pp 191

[Sac 71] J.E.Sacks, H.W.Sorenson. *Nonlinear Extensions of the Fading Memory Filter.* IEEE Trans. on Autom. Contr., Oct. 1971, pp-506-507.

[Sca 74] J.Scarne. Scarne's New Complete Guide to Gambling. Simon Schuster, 1974.

[Sch 92] R.W.Schmittberger. New Rules for Classic games. Wiley, 1992.

[Sch 95] L.W.Schruben. Graphical Simulation Modeling and Analysis. Boyd Fraser, 1995.

[Sha 64] B.V.Shah, R.J.Buehler, O.Kempthorne. Some Algorithms for minimizing a Function of Several Variables. *J. Soc. Indust. Appl. Math.* 12, 74–92.

[Sha 93] W. F. Sharpe. A linear programming algorithm for mutual fund portfolio selection. *Managment Science*, 13:499-510, 1993.

[Sha 70] W.F.Sharpe. *Portfolio Theory and Capital Markets*. McGraw-Hill, New York, 1970.

[Sha 82] W.F.Sharpe C.M.Cootner. *Finantial Economics*. Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[Slo 94] IH Sloan S Joe (1994) *Latice Methods for Multiple Integration*. Oxford: Oxford University Press, Pp 239

[Smi 00] B.T.Smith, J.M.Boyle, J.J.Dongarra, B.S.Garbow, Y.Ikebe, V.C.Klema, C.B.Moler. *Matrix Eigensystem Routines: EISPACK Guide*. Lecture Notes in Computer Science, volume 6, second edition, Springer-Verlag.

[Smi 91] D.K.Smith. Dynamic Programming: A Practical Introduction. Ellis Horwood, 1991.

[Ste 86] L.Sterling E.Shapiro. The Art of Prolog. MIT, 1986.

[Ste 83] J.M.Stern. Análise Numérica. XIII Curso de Verão, mimeo. MAP-IME-USP, 1983.

[Ste 94a] J.M.Stern. Esparsidade, Estrutura, Estabilidade e Escalonamento em Algebra Linear Computacional. IX Escola de Computação, Recife, 123 pps, 1994.

[Ste 94b] J.M.Stern S.Vavasis. Active Set Algorithms for Problems in Block Angular Form. Computational and Applied Mathematics, (13) 1994.

[Ste 95] J.M.Stern M.E.da Silva. Efficient Portfolios at Sao Paulo Stock Exchange. Anais do XVII Encontro Brasileiro de Econometria, V.2, pp.995-1013, Salvador, 1995.

[Ste 96a] J.M.Stern, C.O.Ribeiro. Mtodos de Otimização em Finanças. XIX Congresso Nacional de Matemática Aplicada e Computacional, Goiânia, 140 pps, 1996.

[Ste 96b] J.M.Stern R.Terada. Vade-MeCum, Um Guia para Programação em C. MAC-IME-USP, 42 pps, 1996.

[Ste 98a] J.M.Stern M.Lauretto. REAL: Real Attribute Learning Algorithm. In: World Multiconference on Systemics, Cybernetics and Informatics - SCI'98 Proceedings, V. 2, pp.315-321, 1998.

[Ste 98b] J.M.Stern M.Lauretto. REAL: Algoritmo de Aprendizagem para Atributos Reais e Estratégias de Operaçao em Mercado. In: Proceedings of IBERAMIA 98 - Sixth Iberoamerican Conference on Artificial Intelligence, Lisboa, 1998.

[Ste 00] J.M.Stern. Web page, www.ime.usp.br/∼jstern

[Ste 73] G.W.Stewart. Introduction to Matrix Computations. Academic Press, 1973.

[Sti 86] S.M.Stigler. The History of Statistics. Harvard Univ. Press, 1986.

[Str 86] G.Strang. Introduction to Applied Mathematics. Cambridge Press, 1986.

[Str 88] G.Strang. *Linear Algebra and its Applications.* HBJ, 1988.

[Tan 93] Z. Tang and P.A. Fisher. Feedfoward neural nets as models for time series forecasting. *ORSA Journal on Computing,* 5:374-385, 1993.

[Tod 65] I.Todhunter. A History of the Matematical Theory of Probability. Macmillan, 1965.

[Tho 86] L.C.Thomas. Games, Theory and Applications. Ellis Horwood, 1986.

[Tri 96] L.Trigeorgis. Real Options. MIT, 1996.

[Ung 81] S.Unger, F.Wysotzki. Lernfaehige Klassifizierungssysteme. Akademie Verlag, Berlin, 1981.

[Urb 94] J.S.Urban-Hjorth. *Computer Intensive Statistical Methods: Validation, Model Selec. and Bootstrap.* Chapman Hall, 1994.

[Vie 98] VJ Vieland SE Hodge (1998) Book Reviews: *Statistical Evidence* by R Royall (1997). *Am J Hum Genet* 63:283-289

[Wav 97] L.Waverman W.S.Comanor A.Goto. Competition Policy in the Global Economy. Routledge, 1997.

[Whi 78] D.J.White. *Finite Dynamic Programming.* Whiley, 1978.

[Whi 90] P. Whittle. *Risk–Sensitive Optimal Control.* Wiley, 1990.

[Whi 83] P.Whittle. *Optimization over Time.* vol. I e II, John Wiley, 1983.

[Wic 82] B.A. Wichmann I.D. Hill. An Efficient and Portable Pseudo-Random Number Generator. Appl. Stat. 31, pp. 188-190, 1982.

[Wil 98] C.P.Williams S.H.Clearwater. Explorations in Quantum Computing. Springer, 1998.

[Wol 59] The Simplex Method for Quadratic Programming. *Econometrica*, 27, 383–398, 1959.

[Wyk 89] C.J.van Wyk. Data Structures and C Programs. AT&T Bell Lab, 1989.

[Zen 93] S.Zenios. Financial Optimization. Cambridge, 1993.