

UNIVERSIDADE FEDERAL DE SANTA CATARINA CURSO DE
PÓS-GRADUAÇÃO EM MATEMÁTICA E COMPUTAÇÃO CIENTÍFICA

UM ESTUDO COMPUTACIONAL DE ALGORÍTMOS DE TRAJETÓRIA
CENTRAL PARA PROBLEMAS DE COMPLEMENTARIEDADE LINEAR
MONÓTONA.

MÁRCIO AUGUSTO VILLELA PINTO
FLORIANÓPOLIS - SC
DEZEMBRO DE 1997

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CURSO DE PÓS-GRADUAÇÃO EM MATEMÁTICA E COMPUTAÇÃO CIENTÍFICA

UM ESTUDO COMPUTACIONAL DE ALGORÍTMOS DE TRAJETÓRIA CENTRAL
PARA PROBLEMAS DE COMPLEMENTARIEDADE LINEAR MONÓTONA.

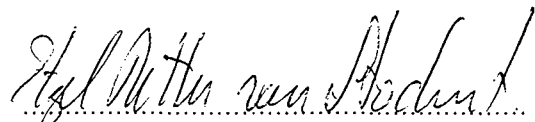
DISSERTAÇÃO APRESENTADA AO CURSO
DE PÓS-GRADUAÇÃO EM MATEMÁTICA E
COMPUTAÇÃO CIENTÍFICA, DO CENTRO
DE CIÊNCIAS EXATAS DA UNIVERSIDADE
FEDERAL DE SANTA CATARINA, PARA
OBTENÇÃO DO GRAU DE MESTRE EM
MATEMÁTICA, COM ÁREA DE
CONCENTRAÇÃO EM OTIMIZAÇÃO.

MÁRCIO AUGUSTO VILLELA PINTO
FLORIANÓPOLIS - SC
DEZEMBRO DE 1997

UNIVERSIDADE FEDERAL DE SANTA CATARINA CURSO DE PÓS-
GRADUAÇÃO EM MATEMÁTICA E COMPUTAÇÃO CIENTÍFICA

MÁRCIO AUGUSTO VILLELA PINTO

Esta Dissertação foi julgada adequada para obtenção do Título de “Mestre”, Área de
Concentração em Otimização, e aprovada em sua forma final pelo Curso de Pós-
Graduação em Matemática e Computação Científica.



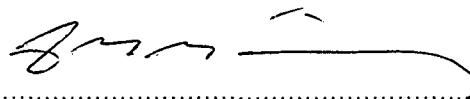
ETZEL RITTER VON STOCKERT

Coordenador

Comissão Examinadora



Prof. Dr. Clóvis Caesar Gonzaga (UFSC - orientador)



Prof. Dr. José Mário Martinez Perez (UNICAMP)



Prof. Dr. Mário César Zambaldi (UFSC)

FLORIANÓPOLIS, 19 DE DEZEMBRO DE 1997.

A DEUS POR TUDO

Agradecimentos

- Ao professor Clóvis Caesar Gonzaga pela excelente orientação; que sem seu auxílio seria impossível a finalização deste trabalho.
- À Capes pelo apoio financeiro.
- A minha família e amigos pela compreensão.
- Ao meu amigo Gilmar por estar sempre próximo.

Resumo

Neste trabalho analisamos, em particular, um algoritmo que segue a trajetória central associado a um problema de complementariedade linear monótona, gerando pontos em vizinhanças grandes da trajetória. Esse algoritmo baseia-se em passos que procuram uma aproximação rápida da face ótima do problema, e, quando houver necessidade, em passos corretores, que provocam uma aproximação à trajetória central.

A trajetória central termina no ponto conhecido como *ótimo central*, que é o centro analítico da face ótima.

Neste trabalho mostramos como este algoritmo gera seqüências que convergem para o ótimo central, e como o algoritmo de centralização é acelerado pela aproximação desse ponto.

Abstract

In this work we analyse an algorithm that follows the central path associated with a monotone linear complementary problem, generating points in the large neighborhoods of the path. This algorithm is based in steps that look for a fast approximation of the optimal face of the problem, and, when necessary, corrector steps that cause an approximation to the central path.

The central path ends in the point known as optimal center, corresponding to the analytic center of the optimal face.

In this work we show how this algorithm generates sequences wich converge to the optimal center, and how the centering algorithm is acelerated when it approaches this point.

Índice

1	Introdução	3
1.1	Convenções	5
2	O Problema de complementariedade linear monótona	7
2.1	Pontos centrais e face ótima	9
2.2	O passo de Newton	12
2.3	Equações escaladas	15
3	Resultados teóricos	17
3.1	Ordens de magnitude para o passo de Newton	17
3.2	Medida de proximidade primal	19
3.3	Distância primal	23
4	Algoritmos	28
4.1	Indicador de Tapia	28
4.2	Os Algoritmos	30
4.2.1	O Algoritmo de Busca	30
4.2.2	O Algoritmo Corretor	31
4.2.3	O algoritmo Bord	32
4.2.4	O algoritmo Front	33
4.2.5	O Algoritmo Intelc	34
4.2.6	O Algoritmo Intel	37
4.2.7	O Algoritmo Largec	39

4.2.8	O Algoritmo Large	41
5	Experimentos numéricos e Conclusões	44
5.1	Experimentos numéricos	44
5.2	Conclusões	56
5.2.1	Eficiência do Indicador de Tapia	56
5.2.2	Desempenho dos algoritmos	59
5.2.3	Propriedades do Teorema 3.3.3	62

Capítulo 1

Introdução

Neste trabalho descrevemos um algoritmo de trajetória central para o *Problema de Complementariedade Linear* (LCP) monótona horizontal, baseado em iteradas viáveis. Este algoritmo começa de um sistema não linear dado pelas condições de otimalidade, e resolve o sistema por aplicações inteligentes do método de Newton. O formato do problema, que será formalmente descrito no próximo capítulo, é o seguinte:

Encontrar $(x, s) \in \mathbb{R}^{2n}$ tal que

$$(P) \quad \begin{aligned} xs &= 0 \\ Qx + Rs &= b \\ x, s &\geq 0, \end{aligned}$$

onde $b \in \mathbb{R}^n$, e $Q, R \in \mathbb{R}^{n \times n}$ são tais que para quaisquer $u, v \in \mathbb{R}^n$,

$$\text{se } Qu + Rv = 0 \text{ então } u^t v \geq 0.$$

Esta formulação trivialmente inclui os *Problemas de Programação Linear* (PL) e de *Programação Quadrática Convexa* expressados por suas condições de otimalidade.

Há muito tempo, a pesquisa em métodos de pontos interiores para programação linear tem sido dominada pelo estudo dos algoritmos Primal-Dual. Estes métodos foram originados a partir da descrição da trajetória central feita por Megiddo [12] e seguidos pelos algoritmos de Kojima, Mizuno e Yoshise [11] e Monteiro e Adler [14].

Algoritmos de trajetória central operam essencialmente da seguinte forma:

Cada iteração começa com um par viável (x, s) e um parâmetro $\mu > 0$ tal que um critério é satisfeito. Este critério é a medida de proximidade, que certifica que (x, s) está próximo do ponto central associado a μ . A iteração consiste em pegar um passo de Newton (com ou sem busca linear) para se aproximar da condição $xs = \mu^+e$, onde $\mu^+ = \gamma\mu$, $\gamma \in (0, 1)$.

A medida de proximidade associada com o par viável (x, s) e um parâmetro $\mu > 0$ usadas neste trabalho, serão dadas na norma Euclidiana e norma Infinito.

Algoritmos baseados em vizinhanças grandes têm maior complexidade, mas são geralmente mais eficientes na prática.

Devemos saber que o passo de Newton, citado acima, é sempre uma combinação de dois passos; de acordo com as escolhas de γ : o passo afim-escala, que tenta alcançar $xs = 0$ em apenas uma iteração (com $\gamma = 0$); e o passo de centralização, que tenta uma busca do ponto central, fazendo $\mu^+ = \mu$ (com $\gamma = 1$).

Atualmente os melhores resultados teóricos relacionando razão de convergência e complexidade computacional são obtidos pelos algoritmos Preditor-Corretor. Esses algoritmos baseiam-se: em passos preditores, que procuram uma aproximação rápida da face ótima do problema com buscas lineares; e isto resulta, em cada iteração, em um ponto na vizinhança da trajetória central. E passos corretores, que provocam uma aproximação à trajetória central, feita por um algoritmo (algoritmo de centralização), baseado no método de Newton.

Resultados teóricos mostram que essa centralização pode requerer um grande número de passos de Newton, quando a vizinhança usada pelo passo preditor é grande.

Por esta razão estudam-se maneiras que realizem essas centralizações apenas quando há necessidade; evitando assim o grande custo computacional.

Uma das maneiras de se fazer isto é criar um novo critério de decisão. E este critério está fortemente relacionado com a medida de proximidade primal, que será estudada com detalhes no capítulo 3.

As iterações combinando esses passos seguem a trajetória central, que nos leva para a face ótima do problema.

A face ótima de um problema de complementariedade linear monótona é caracterizada por uma partição dos índices das variáveis em dois subconjuntos: algumas variáveis (conhecidas como *variáveis pequenas*) são nulas sobre a face ótima, e outras (conhecidas como *variáveis*

grandes) são positivas no interior relativo dessa face. A trajetória central termina no ponto conhecido como *ótimo central*, que é o centro analítico da face ótima.

Neste trabalho descrevemos que o comportamento do algoritmo de centralização é fortemente influenciado pela precisão com que são conhecidas as variáveis grandes: se essas estão próximas de seus valores centrais, então o número de passos de centralização necessários em cada iteração do algoritmo será pequeno.

Fazemos também uma análise computacional dessas propriedades, com problemas aleatórios e o algoritmo programado em Matlab.

Este trabalho está estruturado como segue:

Na seção 2 descrevemos os LCP's e suas principais propriedades. Na seção 3 enunciamos os resultados mais fortes do trabalho. E na seção 4 fazemos uma análise computacional desses resultados.

1.1 Convenções

Dado os vetores x, d , as correspondentes letras maiúsculas denotam as matrizes diagonais X, D definida pelos respectivos vetores. O símbolo e representará o vetor onde todas as componentes são o escalar 1, com a dimensão dada pelo contexto.

Dada a matriz A , o espaço nulo e o espaço coluna são denotados por $\mathcal{N}(A)$ e $\mathcal{R}(A)$ respectivamente. A matriz de projeção sobre $\mathcal{N}(A)$ é P_A , e seu complementar é $\tilde{P}_A = P - P_A$.

Denotaremos a operação componente-a-componente de vetores pela notação usual para números reais. Então, dados dois vetores u, v de mesma dimensão, $uv, u/v$, etc. denotarão os vetores com componentes $u_i v_i, u_i/v_i$, etc. Note que $uv \equiv Uv$ e se A é uma matriz, então $Auv \equiv AUv$.

Frequentemente usaremos a notação $O(\cdot)$ e $\Omega(\cdot)$ para expressar relação entre funções. Nosso uso mais comum será associado com a sequência x^k de vetores e uma sequência μ^k de números reais positivos. Neste caso, $x^k = O(\mu^k)$ significa que existe uma constante K (dependente dos dados do problema) tal que para todo $k \in \mathbb{N}$, $\|x^k\| \leq K\mu^k$. Igualmente, se $x^k > 0$, $x^k = \Omega(\mu^k)$ significa que $(x^k)^{-1} = O(1/\mu^k)$. Finalmente, $x^k \sim \mu^k$ significa que $x^k = O(\mu^k)$ e $x^k = \Omega(\mu^k)$.

Usaremos a mesma notação para relacionarmos um ponto x em um conjunto parametrizado

por μ , digamos E_μ . Dizemos que $x = O(\mu)$ (respectivamente: $x = \Omega(\mu)$, $x \sim \mu$) sempre que existir uma constante K tal que (para μ suficientemente pequeno) para todo $x \in E_\mu$, $\|x\| \leq K\mu$ (respectivamente: $\|x\|^{-1} = O(1/\mu)$, $\|x\| \sim \mu$). Em particular, $x \sim 1$ em E significa que existem constantes $K_2 > K_1 > 0$, tais que qualquer $x \in E$ satisfaz $K_1 \leq x_i \leq K_2, i = 1, \dots, n$.

Capítulo 2

O Problema de complementariedade linear monótona

Obs.: Gostaríamos de ressaltar que os resultados deste capítulo são partes integrantes de [7].

Neste capítulo vamos descrever o Problema de Complementariedade Linear Monótona e destacar suas principais características e propriedades. O Problema de complementariedade linear monótona será estabelecido no seguinte formato: Encontrar $(x, s) \in \mathbb{R}^{2n}$ tal que

$$(P) \quad \begin{aligned} xs &= 0 \\ Qx + Rs &= b \\ x, s &\geq 0, \end{aligned}$$

onde $b \in \mathbb{R}^n$, e $Q, R \in \mathbb{R}^{n \times n}$ são tais que para qualquer $u, v \in \mathbb{R}^n$,

$$\text{se } Qu + Rv = 0 \text{ então } u^t v \geq 0.$$

Esta condição é chamada de: Condição de monotonicidade.

O conjunto viável para (P) e o conjunto de soluções interiores são, respectivamente:

$$\begin{aligned} F &= \{(x, s) \in \mathbb{R}^{2n} \mid Qx + Rs = b, x, s \geq 0\}, \\ F^0 &= \{(x, s) \in F \mid x > 0, s > 0\}. \end{aligned}$$

Dizemos que x é viável se existe s tal que $(x, s) \in F$. Igualmente, s é viável se $(x, s) \in F$ para algum x .

O conjunto de soluções ótimas e o conjunto de soluções ótimas estritamente complementares são, respectivamente:

$$\begin{aligned}\mathcal{F} &= \{(x, s) \in F \mid xs = 0\}, \\ \mathcal{F}^0 &= \{(x, s) \in \mathcal{F} \mid x + s > 0\}.\end{aligned}$$

Este formato é um tanto geral:

Exemplo: O problema de programação quadrática. O problema de programação quadrática convexa é

$$\begin{aligned}\text{minimizar} & \quad c^t x + \frac{1}{2} x^t H x \\ \text{sujeito a} & \quad Ax = b \\ & \quad x \geq 0,\end{aligned}$$

onde $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$, e $H \in \mathbb{R}^{n \times n}$ é uma matriz semi-definida positiva.

A condição necessária e suficiente de otimalidade (Condição de Karush-Khun-Tucker) para este problema é

$$\begin{aligned}x^t s &= 0 \\ -Hx + A^t w + s &= c \\ Ax &= b \\ x, s &\geq 0\end{aligned}$$

Seja B uma matriz cujas linhas geram o espaço nulo de A . Multiplicando a segunda equação por B , obtemos a relação equivalente $-BHx + Bs = Bc$, tal que (x, w, s) satisfaça o sistema de otimalidade de primeira ordem se e só se

$$\begin{aligned}x^t s &= 0 \\ \begin{bmatrix} A & 0 \\ -BH & B \end{bmatrix} \begin{bmatrix} x \\ s \end{bmatrix} &= \begin{bmatrix} b \\ Bc \end{bmatrix} \\ x, s &\geq 0\end{aligned}$$

Agora, seja $u, v \in \mathbb{R}^n$ tal que $Au = 0$ e $-Hu + A^t w + v = 0$. Multiplicando esta equação por u^t , obtemos $u^t v = U^t H u \geq 0$, e concluímos que a condição de otimalidade constitui um problema de complementariedade linear monótona.

Para o caso de programação linear, isto também é trivialmente verdade, com $H = 0$.

2.1 Pontos centrais e face ótima

Pontos centrais. A dificuldade para se resolver (P) é devida ao problema combinatório $xs = 0$.

Um problema mais fácil de se resolver é o seguinte problema perturbado:

$$(P_\mu) \quad \begin{aligned} xs &= \mu e \\ Qx + Rs &= b \\ x, s &> 0 \\ \mu &> 0. \end{aligned}$$

Sabemos de [10] que este sistema tem solução única $(x(\mu), s(\mu))$ que é chamado de ponto central associado ao parâmetro $\mu > 0$. O conjunto de pontos centrais define uma curva diferenciável $\mu > 0 \rightarrow (x(\mu), s(\mu))$ chamada trajetória central. A trajetória central é uma curva que atravessa o conjunto dos pontos interiores, mantendo sempre uma distância razoável das faces não-ótimas de F e termina no centro analítico da face ótima (x^*, s^*) , que será definido a seguir.

Os algoritmos de trajetória central são algoritmos que seguem esta trajetória.

Dado $\mu > 0$, queremos encontrar (x, s) tal que $xs = \mu e$. Podemos então avaliar o desvio desta condição por uma medida de proximidade.

Portanto, dado $(x, s) \in F$ e $\mu > 0$, as proximidades de (x, s) para $(x(\mu), s(\mu))$ são estimadas pelas medidas

$$\begin{aligned} \delta(x, s, \mu) &= \left\| \frac{xs}{\mu} - e \right\| \\ \delta_\infty(x, s, \mu) &= \left\| \frac{xs}{\mu} - e \right\|_\infty. \end{aligned}$$

Estas proximidades definem vizinhanças da trajetória central. Dado $\alpha \in (0, 1)$ e $\mu^0 > 0$; a vizinhança pequena e a vizinhança grande são, respectivamente

$$\begin{aligned} \mathcal{N}_\alpha &= \{(x, s) \in F \mid \delta(x, s, \mu) \leq \alpha \text{ para algum } \mu \leq \mu^0\} \\ \mathcal{N}_\infty &= \{(x, s) \in F \mid \delta_\infty(x, s, \mu) \leq \alpha \text{ para algum } \mu \leq \mu^0\}. \end{aligned}$$

Um par (x, s) é central se e só se ele satisfaz $\delta(x, s, \mu) = 0$ para algum $\mu \in (0, \mu^0]$. Se $\delta(x, s, \mu) = \delta \in (0, 1)$, então

$$\frac{xs}{\mu} = e + \rho, \quad \|\rho\| = \delta.$$

Pré-multiplicando esta expressão por e^t , obtemos:

$$\frac{x^t s}{n} = n\mu(1 + \epsilon), \quad |\epsilon| \leq \frac{\delta}{\sqrt{n}}.$$

De maneira análoga para a proximidade δ_∞ , temos:

$$\frac{x s}{\mu} = e + \rho, \quad \|\rho\|_\infty.$$

Pré-multiplicando esta expressão por e^t , obtemos:

$$\frac{x^t s}{\mu} \leq n + \delta n, \quad \text{pois } |e^t \rho| \leq n\delta.$$

Então:

$$\frac{x^t s}{n} \leq \mu(1 + \delta).$$

Em ambos os casos, temos:

$$\frac{x^t s}{n} \leq 2\mu \Rightarrow \text{gap} \leq 2n\mu.$$

Isto mostra que para um ponto em \mathcal{N}_α ou \mathcal{N}_∞ , μ mede o "gap de complementariedade" $x^t s$; que, no caso dos problemas de programação linear e quadrática, é o gap de dualidade. Em particular, em um ponto central, ou seja, $(x, s) = (x(\mu), s(\mu))$, $\mu = \frac{x^t s}{n}$.

Isto justifica os algoritmos de trajetória central, que geram seqüências (x^k, s^k, μ^k) tal que $\mu^k \rightarrow 0$ e $\delta(x^k, s^k, \mu^k) \leq \alpha \in (0, 1)$.

Ou ainda de (P_μ) , podemos fazer a seguinte interpretação: adicionando $x_i(\mu)s_i(\mu) = n$, vemos que $x(\mu)^t s(\mu) = n\mu$. Tomando $x^t s$ como a função objetivo para minimizar; é suficiente encontrar (aproximadamente) pontos na trajetória central com $\mu \rightarrow 0$.

Face ótima. O conjunto de soluções ótimas de (P) é uma face do poliedro F , denotada por \mathcal{F} .

A face é caracterizada por uma partição $\{B, N, T\}$ do conjunto de índices, chamada partição ótima, tal que $i \in T$ se $x_i = 0$ e $s_i = 0$ para toda solução ótima; $i \in B$ ou $i \in N$, respectivamente, se existe uma solução ótima (x, s) tal que $x_i > 0$ ou $s_i > 0$.

Neste trabalho precisamos de uma análise separada do comportamento das então chamadas *variáveis grandes* x_B, s_N e as *variáveis pequenas* x_N, s_B . Uma grande simplificação é obtida pelas seguintes hipóteses discutidas abaixo:

Notemos que a monotonicidade não é afetada pela reordenação. E a direção de Newton, que será estudada na próxima seção, juntamente com a vizinhança da trajetória central, são invariantes em relação a esta transformação. O que significa que todos os algoritmos baseados no passo de Newton e proximidade da trajetória central são invariantes com relação a esta permutação de variáveis.

É claro que nos algoritmos não se usa o conhecimento desta partição ótima, pois ela é desconhecida. No capítulo 4 descreveremos um procedimento para estimar a partição ótima (Indicador de Tapia), e construiremos um algoritmo baseado nessas estimativas.

Quando for conveniente, suporemos que as variáveis foram ordenadas como mencionado acima, sem alterar sua notação. Obviamente isto somente pode ser feito em estudos teóricos, o que significa que os algoritmos não podem usar esta reordenação.

Neste capítulo assumimos que as variáveis são ordenadas como foi mencionado acima, e então x e s são respectivamente os vetores das variáveis grandes e pequenas.

Obs.: Frequentemente chamaremos x e s respectivamente de variáveis primal e dual; apesar de ser um abuso de linguagem, principalmente pelo fato de termos reordenado as variáveis.

2.2 O passo de Newton

Considere o Problema (P) .

Algoritmos de pontos interiores primais-duais buscam em cada iteração uma solução aproximada para o problema abaixo, que corresponde à busca de um ponto na trajetória central:

$$\begin{aligned} \tilde{x}\tilde{s} &= \mu^+e, \\ Q\tilde{x} + R\tilde{s} &= b, \end{aligned} \tag{2.1}$$

A solução deste problema é impossível em tempo finito.

Dados $(x, s) \in F^0$, estudamos a seguir o Passo de Newton para a busca do ponto central associado a $\mu^+ > 0$, a partir de (x, s) .

$$x^+ = x + u, \quad s^+ = s + v,$$

onde u, v são obtidos resolvendo o seguinte sistema linear:

Hipótese 1: $F^0 \neq \emptyset$.

Hipótese 2: $T = \emptyset$.

A hipótese 1 é necessária para a construção dos algoritmos de pontos interiores viáveis. A hipótese 2 é a hipótese de complementariedade estrita. Sob as hipóteses 1 e 2, a face ótima é limitada e seu interior relativo coincide com $\mathcal{F}^0 \neq \emptyset$.

Então, qualquer par (x, s) no interior relativo da face ótima satisfaz

$$\begin{bmatrix} x_B \\ s_N \end{bmatrix} > 0, \quad \begin{bmatrix} x_N \\ s_B \end{bmatrix} = 0.$$

Uma solução particular tem muito interesse: o ótimo central (x^*, s^*) , o centro analítico da face ótima. Ele é definido como

$$(x^*, s^*) = \operatorname{argmin}_{(x,s) \in \mathcal{F}^0} \left(\sum_{i \in B} \log x_i + \sum_{i \in N} \log s_i \right).$$

Para estudos teóricos, é conveniente uma terceira hipótese; que pode ser feita sem perda de generalidade.

Hipótese 3: $N = \emptyset$.

Assumindo $T = \emptyset$, a hipótese 3 significa que x é o vetor das variáveis grandes e s o vetor das variáveis pequenas; então para qualquer solução ótima estritamente complementar $(x, s) \in \mathcal{F}^0$, $x > 0$ e $s = 0$. Isto certamente não ocorre em problemas práticos, mas isto pode ser obtido por uma reordenação de variáveis, que é feita em [1], e mostramos agora. A restrição $Qx + Rs = b$ pode ser reescrita como:

$$[Q_B \quad R_N] \begin{bmatrix} x_B \\ s_N \end{bmatrix} + [R_B \quad Q_N] \begin{bmatrix} s_B \\ x_N \end{bmatrix} = b.$$

Podemos renomear as variáveis da seguinte forma:

$$Q \leftarrow [Q_B \quad R_N], \quad R \leftarrow [R_B \quad Q_N], \quad x \leftarrow \begin{bmatrix} x_B \\ s_N \end{bmatrix}, \quad s \leftarrow \begin{bmatrix} s_B \\ x_N \end{bmatrix}, \\ N \leftarrow \emptyset, \quad B \leftarrow \{1, \dots, n\}.$$

Com esta reordenação, a face ótima é caracterizada simplesmente por:

$$\mathcal{F} = \{(x, s) \in \mathbb{R}^{2n} \mid s = 0, Qx = b, x \geq 0\}.$$

$$su + xv = -xs + \mu^+ e, \quad Qu + Rv = 0. \quad (2.2)$$

Usualmente a escolha de μ^+ é feita da seguinte forma: dados (x, s) , define-se $\mu(x, s) = \frac{x^+ s}{n}$. Toma-se $\mu^+ = \gamma \mu(x, s)$, onde $\gamma \in [0, 1]$.

Observe que se (x, s) é o ponto central associado a μ , então $\mu(x, s) = \mu$.

Sabemos de [15] que, sob nossas hipóteses, o sistema acima tem solução (u, v) e é única.

Nesta seção estamos interessados em estudar as propriedades da direção de Newton (u, v) descritas acima. A partir da próxima seção descreveremos algoritmos baseados nos passos de Newton, o que significa que iremos olhar para estas direções acompanhadas de buscas lineares.

A proximidade do ponto resultante para um dado $\mu^+ > 0$, $x^+ > 0$ e $s^+ > 0$ é dada por:

$$\delta^+ = \delta(x^+, s^+, \mu^+) = \left\| \frac{x^+ s^+}{\mu^+} - e \right\|. \quad (2.3)$$

O cálculo de δ^+ segue do seguinte argumento: $x^+ s^+ = xs + xv + su + uv$, e do passo de Newton, $xs + xv + su = \mu^+ e$. Subtraindo estas expressões, deduzimos que $x^+ s^+ = uv + \mu^+ e$ e então para $\mu^+ > 0$,

$$\delta^+ = \left\| \frac{uv}{\mu^+} \right\|. \quad (2.4)$$

Definimos $\mu^+ = \gamma \mu$ com $\mu = \frac{x^+ s}{n}$ e $\gamma \in [0, 1]$.

Duas escolhas de γ têm especial interesse:

(i) $\gamma = 0$: o passo afim-escala,

$$x^a = x + u^a, \quad s^a = s + v^a,$$

(ii) $\gamma = 1$: o passo de centralização,

$$x^c = x + u^c, \quad s^c = s + v^c.$$

Por superposição, temos que o passo de Newton 2.2 para $\gamma \in [0, 1]$ satisfaz:

$$(u, v) = \gamma(u^c, v^c) + (1 - \gamma)(u^a, v^a), \quad (2.5)$$

e o ponto resultante será:

$$(x^+, s^+) = \gamma(x^c, s^c) + (1 - \gamma)(x^a, s^a). \quad (2.6)$$

Descrevemos agora um importante resultado em métodos de pontos interiores. É a descrição de uma iteração de Newton para solucionar (P_μ) .

Teorema 2.2.1 *Dado $(x, s) \in F$ e $\mu^+ > 0$ tal que $\delta(x, s, \mu^+) = \delta \leq 0,5$. Seja (x^+, s^+) o resultado de um passo de Newton para 2.1 de (x, s) . Então $\delta(x^+, s^+, \mu^+) \leq \delta^2$.*

Prova. A demonstração está feita com detalhes em [5]. ■

Isto significa que o método de Newton é muito eficiente para resolver (P_μ) no sentido que ele reduz a medida de proximidade quadraticamente na dada região.

Vamos estudar agora uma importante propriedade das direções viáveis.

Definição: Um par $(u, v) \in \mathbb{R}^{2n}$ é uma direção viável se e só se $Qu + Rv = 0$.

As direções viáveis podem ser definidas separadamente para x e s , como segue:

$$\mathcal{U} = \{u \in \mathbb{R}^n \mid Qu + Rv = 0 \text{ para algum } v \in \mathbb{R}^n\},$$

$$\mathcal{V} = \{v \in \mathbb{R}^n \mid Qu + Rv = 0 \text{ para algum } u \in \mathbb{R}^n\}.$$

O lema seguinte nos dá uma boa propriedade geométrica das direções viáveis sob a hipótese da monotonicidade.

Lema 2.2.2 $\mathcal{U} \subset \mathcal{R}(R^t)$ e $\mathcal{V} \subset \mathcal{R}(Q^t)$.

Prova. Considere $u \in \mathcal{U}$. Por definição, para algum $v \in \mathcal{V}$, $Qu + Rv = 0$. Dado $v' \in \mathcal{N}(R)$, e $\lambda \in \mathbb{R}$,

$$Qu + R(v + \lambda v') = 0,$$

e da afirmação de monotonicidade,

$$u^t(v + \lambda v') \geq 0.$$

Isto implica em $u^t v' = 0$. Visto que v' é arbitrário em $\mathcal{N}(R)$, $u \in \mathcal{N}(R^\perp)$. A segunda inclusão do lema é demonstrada igualmente, e com isso completando a prova. ■

2.3 Equações escaladas

Nesta seção trabalharemos com x e s como variáveis grandes e pequenas, respectivamente, ou seja, x e s tomadas após a reordenação citada no final da seção 2.1.

Vamos definir o vetor escala $d = \sqrt{\frac{\mu x}{s}}$, e a mudança de escala das variáveis por

$$\bar{x} = d^{-1}x, \quad \bar{u} = d^{-1}u, \quad \bar{s} = \frac{ds}{\mu}, \quad \bar{v} = \frac{dv}{\mu} \quad (2.7)$$

As outras variáveis como \bar{u}^a , etc. são definidas igualmente.

Nota-se que esta mudança de escala difere da usual, e somente tem sentido quando x e s são respectivamente as variáveis grandes e pequenas. Ela só pode ser usada em estudos analíticos, não em algoritmos.

Esta escala transforma x e s para o mesmo vetor

$$\phi = d^{-1}x = \frac{ds}{\mu} = \sqrt{\frac{xs}{\mu}}, \quad (2.8)$$

e a medida de proximidade é dada por

$$\delta(x, s, \mu) = \|\phi^2 - e\|. \quad (2.9)$$

A equação 2.2 escalada torna-se, depois de dividir por $\mu\phi$,

$$\bar{u} + \bar{v} = -\phi + \gamma\phi^{-1}, \quad (2.10)$$

e $u^t v \geq 0$ sempre que \bar{u}, \bar{v} satisfaz a restrição

$$QD\bar{u} = -\mu R D^{-1}\bar{v}. \quad (2.11)$$

Vamos simplificar a notação e definir $A = QD$.

De acordo com o Lema 2.2.2, sabemos que com a condição de monotonicidade, qualquer direção viável dual \bar{v} satisfaz

$$\bar{v} \in \mathcal{R}(A^t).$$

Usando este fato, e [1], podemos mostrar que \bar{u} é a projeção de $(-\phi + \gamma\phi^{-1})$ sobre a variedade linear definida por $A\bar{u} = -\mu R D^{-1}\bar{v}$, mas $\mu R D^{-1}\bar{v} = O(\mu)$, e então:

$$\bar{u} = P_A(-\phi + \gamma\phi^{-1}) + O(\mu). \quad (2.12)$$

Visto que qualquer solução ótima (x^*, s^*) satisfaz $s^* = 0$, qualquer vetor viável de variáveis pequenas s é também uma direção viável $s - s^*$. Em particular, para o problema escalado, ϕ é uma direção dual viável, e então $\phi \in \mathcal{R}(A^t)$, ou seja,

$$P_A\phi = 0, \quad \tilde{P}_A\phi = \phi, \quad (2.13)$$

onde $\tilde{P}_A = I - P_A$.

Uma propriedade importante do passo de Newton escalado é descrita no seguinte lema.

Lema 2.3.1 *A solução (\bar{u}, \bar{v}) da equação escalada de Newton 2.10 satisfaz:*

$$\begin{aligned} \bar{u} &= P_A(-\phi + \gamma\phi^{-1}) + O(\mu) = \gamma P_A\phi^{-1} + O(\mu), \\ \bar{v} &= \tilde{P}_A(-\phi + \gamma\phi^{-1}) + O(\mu) = -\phi + \gamma\tilde{P}_A\phi^{-1} + O(\mu), \\ \bar{u}^t\bar{v} &= O(\mu). \end{aligned}$$

Prova. Seja (\bar{u}, \bar{v}) a solução da equação escalada. Então $\bar{u} + \bar{v} = -\phi + \gamma\phi^{-1}$. Substituindo 2.12,

$$P_A(-\phi + \gamma\phi^{-1}) + \bar{v} = (-\phi + \gamma\phi^{-1}) + O(\mu).$$

Visto que $\bar{v} \in \mathcal{R}(A^t)$, esta é uma decomposição ortogonal do lado direito da expressão acima, ao longo do $\mathcal{N}(A)$ e $\mathcal{R}(A^t)$. Segue que

$$\bar{v} = \tilde{P}_A(-\phi + \gamma\phi^{-1}) + \tilde{P}_AO(\mu) = \tilde{P}_A(-\phi + \gamma\phi^{-1}) + O(\mu).$$

A outra expressão para a projeção, vem do fato de que $P_A\phi = 0$. Por substituição direta, podemos deduzir que $\bar{u}^t\bar{v} = O(\mu)$, completando a prova. ■

Capítulo 3

Resultados teóricos

O objetivo deste capítulo é estudar as ordens de magnitude para o passo de Newton e o efeito do passo de centralização na proximidade.

Já havíamos definido a medida de proximidade primal-dual $\delta(x, s, \mu)$ associada com x, s, μ , mas agora queremos examinar como as variáveis grandes x desviam-se da centralidade se analisadas separadamente.

Para isso, vamos supor a reordenação feita no final da seção 2.1, ou seja, x é o vetor das variáveis grandes e s é o vetor das variáveis pequenas.

3.1 Ordens de magnitude para o passo de Newton

Obs.: Gostaríamos de ressaltar que os resultados das seções 3.1 e 3.2 são resultados integrantes de [7].

Definição: Dados $\epsilon > 0$ e $\mu_0 > 0$. O conjunto em que todos algoritmos convergentes primal-dual viáveis operam é definido por:

$$F_\epsilon = \{(x, s) \in F \mid \mu \leq \mu_0, \frac{xs}{\mu} \geq \epsilon e\}. \quad (3.1)$$

Obs.: Se considerarmos $1 - \alpha = \epsilon$, então $\mathcal{N}_\infty \subset F_\epsilon$

Lema 3.1.1 *Considere o passo de Newton 2.2 a partir de um ponto $(x, s) \in F_\epsilon$ e (\bar{u}, \bar{v}) dados pela mudança de escala 2.7. Então:*

$$\begin{aligned}
x &\sim 1, \quad s \sim \mu, \quad d \sim 1, \quad \phi \sim 1; \\
\bar{u} &= O(\gamma) + O(\mu), \quad u = O(\gamma) + O(\mu); \\
\bar{v} &= O(1), \quad v = O(\mu); \\
u^t v &= O(\mu^2).
\end{aligned}$$

Prova. A magnitude de x está demonstrada com detalhes em [15]. Usando o fato que $\delta(x, s, \mu) \leq \alpha$, temos que $s \sim \mu$. Logo: $d = \sqrt{\frac{\mu x}{s}} \sim 1$ e $\phi = d^{-1}x \sim 1$. Temos que:

$$\bar{u} + \bar{v} = -\phi + \gamma\phi^{-1} = -\frac{ds}{\mu} + \gamma\frac{d}{x} = \frac{-dxs + d\gamma\mu}{\mu x} = \frac{d}{\mu x}(-xs + \gamma\mu e).$$

Denotemos $f = -xs + \gamma\mu e$, então

$$\bar{u} + \bar{v} = \frac{df}{\mu x}.$$

Tomando a norma, temos:

$$\|\bar{u} + \bar{v}\| = \left\| \frac{df}{\mu x} \right\| \quad \text{e conseqüentemente} \quad \|\bar{v}\| \leq \left\| \frac{df}{\mu x} \right\|,$$

mas $x \sim 1$ e $d \sim 1$; logo $\bar{v} = O(\frac{f}{\mu})$, e como $v = \frac{\mu\bar{v}}{d}$, temos que $v = O(\mu\bar{v}) = O(f)$. Mas: $f = -xs + \gamma\mu e = O(\mu) + O(\gamma\mu) = O(\mu)$. Portanto: $\bar{v} = O(1)$ e $v = O(\mu)$. Para provar o mesmo para as variáveis grandes, usaremos o Lema 2.3.1.

$$\bar{u} = \gamma P_A \phi^{-1} + O(\mu) \quad \Rightarrow \quad \bar{u} = O(\gamma) + O(\mu).$$

Lembrando que $u = d^{-1}\bar{u}$, temos que $u = O(\bar{u})$. Então:

$$u = O(\gamma) + O(\mu).$$

Vamos agora analisar a expressão $u^t v$. De 2.7, temos:

$$u^t v = \mu \bar{u}^t \bar{v}.$$

Novamente pelo Lema 2.3.1, temos: $u^t v = \mu O(\mu) = O(\mu^2)$. Completando a prova. ■

O próximo lema usa o resultado anterior para expor algumas propriedades interessantes com relação ao passo afim-escala.

Lema 3.1.2 *Considere o passo afim-escala (u^a, v^a) de (x, s) , obtido da equação de Newton com $\gamma = 0$. Então:*

$$x^a = x + O(\mu), \quad s^a = O(\mu^2).$$

Prova. Do lema 3.1.1 com $\gamma = 0$, $u^a = O(\mu)$ e portanto $x^a = x + u^a = x + O(\mu)$, provando a primeira igualdade. Temos:

$$s^a = s + v^a;$$

$$s^a = \mu d^{-1} \bar{s} + \mu d^{-1} \bar{v}^a = \mu d^{-1} (\bar{s} + \bar{v}^a) = \mu d^{-1} (\phi + \bar{v}^a).$$

Pelo Lema 2.3.1, $\bar{v} = -\phi + \gamma \tilde{P}_A \phi^{-1} + O(\mu)$, com $\gamma = 0$, $\bar{v} = -\phi + O(\mu)$, e segue que $s^a = \mu d^{-1} (\phi - \phi + O(\mu)) = \mu d^{-1} O(\mu)$. Visto que $d^{-1} \sim 1$, isto completa a prova. ■

3.2 Medida de proximidade primal

Dado um vetor viável de variáveis grandes x , define-se "medida de proximidade primal" que depende somente de x , e estima em um certo sentido a distância ao centro analítico da face ótima, dada por:

$$\delta_p(x) = \min_s \{ \|xs - e\| \mid s \in \mathcal{R}(Q^t) \} = \min_{\bar{s}} \{ \|\phi \bar{s} - e\| \mid \bar{s} \in \mathcal{R}(A^t) \}. \quad (3.2)$$

A expressão acima está estabelecida em ambas, variáveis originais e variáveis escaladas.

A proximidade pode ser calculada da seguinte forma: fixando $y = xs$, obtemos $\delta_p(x) = \min_y \{ \|y - e\| \mid y \in \mathcal{R}(XQ^t) \}$. Usando a definição de projeção, vemos que o valor deste mínimo é (veja [6]):

$$\delta_p(x) = \|P_{QX} e\| = \|P_{A\Phi} e\|. \quad (3.3)$$

A última igualdade vem de $QX = A\Phi$, e isto se dá por substituição direta.

Então, temos a medida de proximidade primal-dual $\delta(x, s, \mu)$ e a medida de proximidade primal $\delta_p(x)$. Por definição, visto que s e então $\frac{s}{\mu}$ estão em $\mathcal{R}(Q^t)$, podemos ter sempre que $\delta_p(x) \leq \delta(x, s, \mu)$. Nossa próxima tarefa é descrever o efeito do passo de Newton sobre ambas as medidas, e inter relacioná-las.

Porém, antes vamos enunciar dois resultados importantes de álgebra linear.

Lema 3.2.1 Seja $p, q \in \mathbb{R}^n$ tais que $p^t q \geq 0$. Então

$$\|pq\| \leq \frac{1}{\sqrt{8}} \|p + q\|^2.$$

Prova. Este resultado é devido a Mizuno (veja [13]). ■

Lema 3.2.2 Seja $q \in \mathbb{R}^n$ tal que $\|q - e\|_\infty \leq \alpha$, onde $\alpha \in (0, 1)$. Considere as projeções $\hat{h} = P_A \rho$, $h = q P_{A Q} q \rho$, onde $A \in \mathbb{R}^{m \times n}$ e $\rho \in \mathbb{R}^n$. Então

$$\|h - \hat{h}\| \leq \alpha(1 + \alpha) \frac{2 - \alpha}{1 - \alpha} \|\hat{h}\|.$$

Em particular:

$$\|q - e\|_\infty \leq 0,155 \Rightarrow \|h - \hat{h}\| \leq 0,4 \|\hat{h}\|,$$

$$\text{e consequentemente } \|h\| \leq 1,4 \|\hat{h}\|.$$

Prova. Veja a demonstração em [9]. ■

Na seção anterior fizemos algumas afirmações a respeito das propriedades do passo afim-escala; vamos agora olhar um pouco para o passo de centralização.

Dados x, s, μ , o passo de Newton de centralização é dado por:

$$xv^c + su^c = -xs + \mu e, \quad Qu^c + Rv^c = 0. \quad (3.4)$$

De 2.10 e do lema 2.3.1 temos:

$$\bar{u}^c + \bar{v}^c = -\phi + \phi^{-1}. \quad (3.5)$$

E então

$$\bar{u}^c = P_{QD}(-\phi + \phi^{-1}) + O(\mu) = P_{QD}\left(-\frac{ds}{\mu} + dx^{-1}\right) + O(\mu);$$

Logo:

$$u^c = DP_{QD}\left(-\frac{ds}{\mu} + dx^{-1}\right) + O(\mu) = DP_{QD}D\left(-\frac{s}{\mu} + x^{-1}\right) + O(\mu). \quad (3.6)$$

Definindo $\psi = \sqrt{\frac{\mu}{xs}}$, então $\psi = d^{\frac{1}{x}}$, e portanto $d = x\psi$, ou seja: $D = X\Psi$.

Portanto 3.6 torna-se:

$$u^c = DP_{QD}D\left(-\frac{s}{\mu} + x^{-1}\right) + O(\mu) = X\Psi P_{QX}\Psi X\Psi\left(-\frac{s}{\mu} + x^{-1}\right) + O(\mu). \quad (3.7)$$

Agora provaremos alguns lemas sobre o efeito do passo de centralização de Newton (u^c, v^c) em ambas as medidas de proximidade.

O primeiro deles mostra-nos que movendo ao longo da direção de centralização, reduz-se a proximidade primal continuamente.

Lema 3.2.3 *Se $\delta(x, s, \mu) = \delta \leq 0,25$, então para todo $\lambda \in [0, 1]$,*

$$\delta(x + \lambda u^c, s + \lambda v^c, \mu) \leq (1 - \lambda)\delta + \lambda^2 \frac{\delta^2}{2,12}.$$

Em particular para $\lambda = 0,7$,

$$\delta_p(x + \lambda u^c) \leq \delta(x + \lambda u^c, s + \lambda v^c, \mu) \leq 0,09.$$

Prova. Temos, por definição, para $\lambda \in [0, 1]$,

$$\delta(x + \lambda u^c, s + \lambda v^c, \mu) = \left\| \frac{(x + \lambda u^c)(s + \lambda v^c)}{\mu} - e \right\|.$$

Desenvolvendo o numerador,

$$(x + \lambda u^c)(s + \lambda v^c) = xs + \lambda(xv^c + su^c) + \lambda^2 u^c v^c.$$

Subtraindo $\mu e = \lambda \mu e + (1 - \lambda)\mu e$ desta equação,

$$\begin{aligned} (x + \lambda u^c)(s + \lambda v^c) - \mu e &= (1 - \lambda)(xs - \mu e) \\ &+ \lambda(xv^c + su^c + xs - \mu e) + \lambda^2 u^c v^c. \end{aligned}$$

Do passo de Newton 3.4, $xv^c + su^c + xs - \mu e = 0$, dividindo por μ e então tomando a norma,

$$\begin{aligned} \delta(x + \lambda u^c, s + \lambda v^c, \mu) &= \left\| (1 - \lambda) \left[\frac{xs}{\mu} - e \right] + \lambda^2 \frac{u^c v^c}{\mu} \right\| \\ &\leq (1 - \lambda)\delta(x, s, \mu) + \lambda^2 \left\| \frac{u^c v^c}{\mu} \right\|. \end{aligned}$$

O último termo pode ser calculado usando as variáveis escaladas. De 3.5,

$$\bar{u}^c + \bar{v}^c = -\phi + \phi^{-1} = \phi^{-1}(-\phi^2 + e).$$

Pela condição de monotonicidade $(\bar{u}^c)^t \bar{v} \geq 0$, o que significa que podemos usar o lema 3.2.1, e portanto

$$\left\| \frac{u^c v^c}{\mu} \right\| = \|\bar{u}^c \bar{v}^c\| \leq \frac{1}{\sqrt{8}} \|\phi^{-1}(\phi^2 - e)\|^2 \leq \frac{1}{\sqrt{8}} \|\phi^{-2}\|_{\infty} \|\phi^2 - e\|^2.$$

Mas $\|\phi^2 - e\| = \delta$ por 2.9, e portanto para $i = 1, \dots, n$,

$$-\delta \leq \phi_i^2 - 1 \leq \delta \quad \text{e consequentemente} \quad \phi_i^2 \geq 1 - \delta \Rightarrow \frac{1}{\phi_i^2} \leq \frac{1}{1 - \delta} \leq \frac{1}{0,75}.$$

Substituindo, obtemos

$$\left\| \frac{u^c v^c}{\mu} \right\| \leq \frac{\delta^2}{2,12},$$

que nos guia para o resultado desejado, completando a prova. \blacksquare

Vamos ver agora o efeito do passo primal-dual na proximidade primal e perceber que uma centralização primal limitada também será obtida.

Lema 3.2.4 *Sejam $\delta(x, s, \mu) = \delta \leq 0,25$ e $\delta_p = \delta_p(x)$. Considere o passo de centralização (u^c, v^c) . Então para qualquer $\lambda \in [0, 1]$,*

$$\delta_p(x + \lambda u^c) \leq \delta_p - \lambda \frac{\delta_p}{2} + \lambda \delta_p^2 + O(\mu) \leq \delta_p - \frac{\lambda}{4} \delta_p + O(\mu).$$

Prova. Veja a demonstração com detalhes em [7]. \blacksquare

O próximo lema nos fornece um dos resultados mais importantes sobre o efeito da proximidade primal no comportamento do passo de centralização primal-dual.

Lema 3.2.5 *Seja $\delta(x, s, \mu) = \delta \leq 0,25$. Considere o passo de centralização (u^c, v^c) . Se $\delta_p \leq 0,1$, então*

$$\left\| \frac{x s^c}{\mu} - e \right\| \leq 0,175 + O(\mu).$$

Prova. Assumimos que $\delta_p(x) \leq 0,1$. Por 3.3, $\|P_{A\Phi} e\| \leq 0,1$. De 3.4,

$$x s + x v^c - \mu e = -s u^c, \quad \text{logo} \quad x(s + v^c) - \mu e = -s u^c, \quad \text{e portanto} \quad x s^c - \mu e = -s u^c,$$

dividindo por μ , temos

$$\frac{x s^c}{\mu} - e = -\frac{s u^c}{\mu} = -\bar{s} \bar{u}^c.$$

Portanto:

$$\frac{x s^c}{\mu} - e = -\frac{s u^c}{\mu} = -\phi \bar{u}^c.$$

O que significa que deveremos desenvolver a expressão $\|\phi\bar{u}^c\|$. Do lema 2.3.1 com $\gamma = 1$ e $\phi = O(1)$,

$$\phi\bar{u}^c = \phi P_A \phi^{-1} + O(\mu) = \phi^2 \phi^{-1} P_A \phi^{-1} + O(\mu)$$

e então

$$\left\| \frac{x s^c}{\mu} - e \right\| = \|\phi\bar{u}^c\| = \|\phi^2 \phi^{-1} P_A \phi^{-1}\| + O(\mu)$$

E portanto

$$\left\| \frac{x s^c}{\mu} - e \right\| \leq \|\phi^2\|_\infty \|\phi^{-1} P_A \phi^{-1}\| + O(\mu) \leq 1,25 \|\phi^{-1} P_A \phi^{-1}\| + O(\mu),$$

porque para $i = 1, \dots, n$, $|\phi_i^2 - 1| \leq \|\phi^2 - e\| = \delta \leq 0,25$. Usando o lema 3.2.2, concluímos que

$$\|\phi^{-1} P_A \phi^{-1}\| \leq 1,4 \|P_{A\Phi} e\| \leq 0,14.$$

Mergulhando isto na desigualdade acima, obtemos

$$\left\| \frac{x s^c}{\mu} - e \right\| \leq 0,175 + O(\mu),$$

completando a prova. ■

3.3 Distância primal

Obs.: Esta seção foi baseada nos manuscritos em preparação [3].

Definimos agora uma outra medida de proximidade que depende de x e o ponto central $x(\mu)$ associado com o parâmetro $\mu > 0$.

Esta medida nos fornece o desvio das variáveis grandes com relação aos seus valores centrais, o que significa que temos a distância real de x com relação a $x(\mu)$, ou seja, a distância primal.

Definição: Dados $(x, s) \in F$ e $\mu > 0$, definimos:

$$dist(x, \mu) = \left\| \frac{x - x(\mu)}{x(\mu)} \right\|. \quad (3.8)$$

Obs.: O mesmo pode-se definir para as variáveis pequenas. Fornecendo assim o desvio das variáveis pequenas com relação aos seus valores centrais:

$$dist(s, \mu) = \left\| \frac{s - s(\mu)}{s(\mu)} \right\| \quad (3.9)$$

O lema a seguir nos mostra uma relação importante:

Lema 3.3.1 Seja $\delta(x, s, \mu) = \delta \leq 0,5$. Então

$$\begin{aligned} \text{dist}(x, \mu) &\leq \sqrt{1 - \sqrt{1 - \delta^2}} + O(\mu); \\ \text{dist}(s, \mu) &\leq \sqrt{1 - \sqrt{1 - \delta^2}} + O(\mu). \end{aligned}$$

Em particular:

$$\begin{aligned} \delta = 0,25 &\Rightarrow \text{dist}(x, \mu) \leq 0,2541 \\ \delta = 0,5 &\Rightarrow \text{dist}(x, \mu) \leq 0,541 \\ \delta = 0,1 &\Rightarrow \text{dist}(x, \mu) \leq 0,1003 \end{aligned}$$

Prova. Veja a demonstração em [8] e [16]. ■

Obs.: Com isso vemos que, para valores pequenos, a medida de proximidade primal-dual δ é muito boa.

O próximo lema relaciona o efeito da distância $\text{dist}(x, \mu)$ no comportamento do passo de centralização primal-dual.

Lema 3.3.2 Seja $\delta(x, s, \mu) = \delta \leq 0,25$. Considere o passo de centralização (u^c, v^c) . Se $\text{dist}(x, \mu) < 0,1$, então

$$\delta(x, s^c, \mu) \leq 0,136 + O(\mu).$$

Prova. Pelo lema 3.2.3 temos que $\delta(x^c, s^c, \mu) \leq 0,03$. Aplicando o lema 3.3.1, obtemos:

$$\text{dist}(s^c, \mu) \leq \sqrt{1 - \sqrt{1 - 2(0,03)^2}} + O(\mu) = 0,03 + O(\mu).$$

Seja $x = x(\mu)(e + \Theta)$ e $s^c = s(\mu)(e + \Lambda)$, então:

$$\begin{aligned} x &= x(\mu) + \Theta x(\mu) \Rightarrow \Theta = \frac{x - x(\mu)}{x(\mu)} \\ s^c &= s(\mu) + \Lambda s(\mu) \Rightarrow \Lambda = \frac{s^c - s(\mu)}{s(\mu)} \end{aligned}$$

Então:

$$\delta(x, s^c, \mu) = \left\| \frac{x s^c}{\mu} - e \right\| = \left\| \frac{x(\mu)(e + \Theta)s(\mu)(e + \Lambda)}{\mu} - e \right\|;$$

lembrando do fato que $\frac{x(\mu)s(\mu)}{\mu} = e$, temos,

$$\delta(x, s^c, \mu) = \|(e + \Theta)(e + \Lambda) - e\| = \|\Theta + \Lambda + \Theta\Lambda\|,$$

aplicando a desigualdade triangular, vem

$$\delta(x, s^c, \mu) \leq \|\Theta\| + \|\Lambda\| + \|\Theta\Lambda\|.$$

Mas:

$$\begin{aligned}\|\Theta\| &= \left\| \frac{x-x(\mu)}{x(\mu)} \right\| = \text{dist}(x, \mu) \leq 0,1; \\ \|\Lambda\| &= \left\| \frac{s^c-s(\mu)}{s(\mu)} \right\| = \text{dist}(s^c, \mu) \leq 0,03 + O(\mu); \\ \|\Theta\Lambda\| &\leq \frac{1}{\sqrt{8}} \|\Theta + \Lambda\|^2 \leq \frac{1}{\sqrt{8}} (0,1 + 0,03 + O(\mu))^2.\end{aligned}$$

Obs: A última desigualdade segue do lema 3.2.1. Portanto:

$$\delta(x, s^c, \mu) \leq 0,136 + O(\mu),$$

completando a prova. ■

O resultado principal desta seção está descrito no próximo teorema. Ele nos mostra o efeito do passo de centralização em \mathcal{N}_∞ para o caso em que as variáveis grandes estejam bem centradas.

Teorema 3.3.3 *Seja $\delta_\infty(x, s, \mu) \leq 0,25$ e $\text{dist}(x, \mu) \leq 0,1$. Considere o passo de centralização (x^c, v^c) . Então o passo de Newton exato para a centralização de (x, s) é viável e*

$$\begin{aligned}\delta(x^c, s^c, \mu) &\leq 0,2 + O(\mu), \\ \text{dist}(x^c, \mu) &\leq 0,05 + O(\mu).\end{aligned}$$

Prova. O passo de centralização é dado por 3.4. Fazendo a mudança das variáveis $\bar{x} = \frac{x}{x(\mu)}$, e $\bar{s} = \mu \frac{s}{s(\mu)}$, temos:

$$\bar{x}\bar{s} = \frac{x}{x(\mu)} \mu \frac{s}{s(\mu)} = xs \frac{\mu}{x(\mu)s(\mu)} = xs.$$

Esta mudança de variáveis acarreta em $\bar{x}(\mu) = e$ e $\bar{s}(\mu) = \mu e$.

Usando, se necessário, esta mudança de variáveis, podemos assumir, sem perda de generalidade, que $x(\mu) = e$ e $s(\mu) = \mu e$. Vejamos:

$$\begin{aligned}\text{dist}(x, \mu) &= \left\| \frac{x-x(\mu)}{x(\mu)} \right\| = \left\| \frac{x-e}{e} \right\| = \|x - e\|, \\ \text{dist}(s, \mu) &= \left\| \frac{s-s(\mu)}{s(\mu)} \right\| = \left\| \frac{s-\mu e}{\mu e} \right\| = \left\| \frac{s}{\mu} - e \right\|.\end{aligned}$$

Tomando a equação de Newton 3.4, podemos reescrevê-la como segue:

$$xv^c + xs - \mu e = -su^c \Rightarrow x(s + v^c) - \mu e = -su^c \Rightarrow xs^c - \mu e = -su^c;$$

dividindo por μ e depois tomando a norma, temos:

$$\frac{xs^c}{\mu} - e = -\frac{su^c}{\mu} \Rightarrow \left\| \frac{xs^c}{\mu} - e \right\| = \left\| \frac{su^c}{\mu} \right\|.$$

Mas $\left\| \frac{xs^c}{\mu} - e \right\| = \delta(x, s^c, \mu)$. E sabemos que

$$\delta(x^c, s^c, \mu) = \left\| \frac{x^c s^c}{\mu} - e \right\| \leq \left\| \frac{xs^c}{\mu} - e \right\| = \delta(x, s^c, \mu) = \left\| \frac{su^c}{\mu} \right\|.$$

Logo, devemos limitar a expressão

$$\left\| \frac{su^c}{\mu} \right\|.$$

Notação: $u \equiv u^c$.

a) Limitando $\|u\|$: Considere a seguinte "direção de centralização primal" (veja [6]):

$$u^p = x\bar{u}^p \quad e \quad \bar{u}^p = P_{QX} X \left(-\frac{s}{\mu} + x^{-1} \right).$$

Pelas propriedades de projeção, temos:

$$\|\bar{u}^p\| = \min \left\{ \left\| \frac{xs}{\mu} - e - (QX)^t y \right\| \mid y \in \mathbb{R}^n \right\}.$$

Em particular, (veja [7]), $s - s(\mu)$ é viável, então pelo lema 2.2.2 $(s - s(\mu)) \in \mathcal{R}(Q^t)$. Isto significa que existe $\tilde{y} \in \mathbb{R}^n$ tal que

$$\begin{aligned} s - s(\mu) &= \mu Q^t \tilde{y} \Rightarrow s = s(\mu) + \mu Q^t \tilde{y} \\ \frac{xs}{\mu} &= \frac{xs(\mu)}{\mu} + (QX)^t \tilde{y} \Rightarrow \frac{xs}{\mu} - (QX)^t \tilde{y} = \frac{xs(\mu)}{\mu}. \end{aligned}$$

Sabemos que

$$\begin{aligned} \|\bar{u}^p\| &= \min \left\{ \left\| \frac{xs}{\mu} - e - (QX)^t y \right\| \mid y \in \mathbb{R}^n \right\} \leq \left\| \frac{xs}{\mu} - e - (QX)^t \tilde{y} \right\| = \\ &= \left\| \frac{xs}{\mu} - (QX)^t \tilde{y} - e \right\| = \left\| \frac{xs(\mu)}{\mu} - e \right\| = \|x - e\| = \text{dist}(x, \mu) \leq 0,1. \end{aligned}$$

Como $\|x - e\| \leq 0,1$ temos que $|x_i - 1| \leq 0,1, \forall i$. Então $0,9 \leq x_i \leq 1,1$, e, conseqüentemente, $\|x\|_\infty \leq 1,1$. Logo: $\|u^p\| = \|x\bar{u}^p\| \leq \|x\|_\infty \|\bar{u}^p\| \leq 1,1 \times 0,1 = 0,11$. Agora vamos relacionar $\|u\|$ e $\|u^p\|$ pelo lema 3.2.2: Temos:

$$\begin{aligned} u &= X\Psi P_{QX\Psi} X\Psi \left(-\frac{s}{\mu} + x^{-1} \right) + O(\mu) \\ u^p &= X P_{QX} X \left(-\frac{s}{\mu} + x^{-1} \right). \end{aligned}$$

Note que :

$$\psi = \sqrt{\frac{\mu}{xs}} \Rightarrow \psi_i = \sqrt{\frac{\mu}{x_i s_i}} \Rightarrow \psi_i^2 = \frac{\mu}{x_i s_i} \Rightarrow \frac{1}{\psi_i^2} = \frac{x_i s_i}{\mu}.$$

Mas por hipótese $\delta_\infty(x, s, \mu) = \left\| \frac{xs}{\mu} - e \right\|_\infty \leq 0,25$. Ou seja:

$$\left| \frac{x_i s_i}{\mu} - 1 \right| \leq 0,25 \Rightarrow 0,75 \leq \frac{x_i s_i}{\mu} \leq 1,25.$$

Então:

$$\begin{aligned} \frac{1}{\psi_i^2} \geq 0,75 &\Rightarrow \psi_i^2 \leq 1,333 \Rightarrow \\ \psi_i &\leq 1,155 \Rightarrow \psi_i - 1 \leq 0,155 \Rightarrow \\ |\psi_i - 1| &\leq 0,155 \Rightarrow \|\psi - e\|_\infty \leq 0,155. \end{aligned}$$

Já estamos aptos para aplicar o lema 3.2.2:

$$\|u\| \leq 1,4 \|u^p\| + O(\mu) \leq 1,4 \times 0,11 + O(\mu) = 0,154 + O(\mu). \square$$

b) Limitando $\left\| \frac{su}{\mu} \right\|$

Por hipótese

$$\left\| \frac{xs}{\mu} - e \right\|_\infty \leq 0,25 \Rightarrow \left| \frac{x_i s_i}{\mu} - 1 \right| \leq 0,25 \Rightarrow 0,75 \leq \frac{x_i s_i}{\mu} \leq 1,25.$$

Mas

$$\|x - e\| \leq 0,1 \Rightarrow x_i \geq 0,9 \Rightarrow \frac{0,9s_i}{\mu} \leq \frac{s_i}{\mu} \leq 1,39 \Rightarrow \left\| \frac{s}{\mu} \right\|_\infty \leq 1,39.$$

Finalmente:

$$\left\| \frac{su}{\mu} \right\| \leq \left\| \frac{s}{\mu} \right\|_\infty \|u\| \leq 1,39 \times [0,154 + O(\mu)] \leq 0,22 + O(\mu). \square$$

c) O resultado em $dist(x^c, \mu)$ segue de $\|u - u^p\| \leq 0,4 \|u^p\|$, e propriedades de centralização de u^p (direção primal de centralização). Veja [6]. \square

Completando a prova. ■

Capítulo 4

Algoritmos

4.1 Indicador de Tapia

Considere o problema de complementariedade linear monótona (P) com a notação descrita nos capítulos anteriores.

O Indicador de Tapia, que faz uma estimativa das restrições ativas e inativas (o que é equivalente a identificar as variáveis grandes e pequenas), é uma função definida por $(x, s) \in F^0 \mapsto I(x, s) \in \mathbb{R}^{2n}$ com $I(x, s) = \left[\frac{u^a}{x}, \frac{v^a}{s} \right]$, onde (u^a, v^a) é a solução de $xv + su = -xs$, $Qu + Rv = 0$.

A seguir apresentaremos, sem demonstração, um Teorema que estabelece algumas propriedades de interesse para o Indicador de Tapia.

Considere o problema de complementariedade linear com a hipótese de complementariedade estrita.

Teorema 4.1.1 *Seja $(z^k)_{k \in \mathbb{N}}$ uma seqüência $z^k = (x^k, s^k)$ (não necessariamente gerada por um processo iterativo) tal que $z^k \rightarrow z^* \in \mathcal{F}^0$. Seja $(u^k, v^k) = (u_a^k, v_a^k)$ a direção afim-escala associada a z^k . Então:*

$$\begin{aligned} \frac{u_N^k}{x_N^k} \rightarrow -1 \quad , \quad \frac{v_N^k}{s_N^k} \rightarrow 0 \\ \frac{u_B^k}{x_B^k} \rightarrow 0 \quad , \quad \frac{v_B^k}{s_B^k} \rightarrow -1; \end{aligned}$$

conseqüentemente, $I(x^k, s^k) \rightarrow (I^x, I^s)$, onde

$$\begin{aligned} I_N^x &= -e \quad , \quad I_N^s = 0 \\ I_B^x &= 0 \quad , \quad I_B^s = -e. \end{aligned}$$

Do Teorema acima podemos concluir que $\frac{u_i^k}{x_i^k} \rightarrow 0$ para $i \in B$, e $\frac{u_i^k}{x_i^k} \rightarrow -1$ para $i \in N$. Portanto, a taxa de variação $\frac{u_i^k}{x_i^k}$ pode servir como indicador.

Corolário 4.1.2 *Considere k suficientemente grande. Então tem-se:*

$$\begin{aligned} \frac{v_i^k}{s_i^k} &> \frac{u_i^k}{x_i^k} \quad \text{se } i \in N \\ \frac{v_i^k}{s_i^k} &< \frac{u_i^k}{x_i^k} \quad \text{se } i \in B. \end{aligned}$$

Este fato é usado para estimar a partição ótima.

O algoritmo abaixo constrói uma estimativa $\{\tilde{B}, \tilde{N}\}$ da partição ótima $\{B, N\}$. Pelo Teorema acima, se x e s estão suficientemente próximos da face ótima, então esta estimativa coincide com a partição ótima.

Algoritmo 4.1.3 *Dados: $(x, s) \in F^0$.*

Calcule a direção afim-escala (u, v) resolvendo

$$\begin{aligned} su + xv &= -xs \\ Qu + Rv &= 0. \end{aligned}$$

Calcule: $x^a = x + u$, e $s^a = s + v$

Construa a partição $\{\tilde{B}, \tilde{N}\}$ de $\{1, \dots, n\}$.

Para $i=1, \dots, n$

Se $\frac{v_i^k}{s_i^k} > \frac{u_i^k}{x_i^k}$, então $i \in \tilde{N}$.

Senão $i \in \tilde{B}$.

Construção de $\{\tilde{B}, \tilde{N}\}$.

4.2 Os Algoritmos

Os algoritmos que nós trabalharemos estarão na vizinhança \mathcal{N}_∞ , definida com um raio fixo $\alpha \in (0, 1)$. Os nossos procedimentos trabalharão com as triplas (x, s, μ) , onde (x, s) é um par primal-dual interior viável e $\mu > 0$.

Teremos os algoritmos principais:

1. Algoritmo Bord;
2. Algoritmo Front;
3. Algoritmo Intelc;
4. Algoritmo Intel;
5. Algoritmo Largec;
6. Algoritmo Large.

E os algoritmos secundários:

1. Algoritmo de Busca.
2. Algoritmo Corretor.

4.2.1 O Algoritmo de Busca

Este Algoritmo é composto de apenas uma iteração e efetua uma Busca Linear ao longo de uma direção dada.

Algoritmo 4.2.1 *Dados:* $x, s, \mu > 0, (u, v)$ e $\alpha > 0$.

Considere as aplicações para $\theta \in \mathbb{R}$

$$x(\theta) = x + \theta u, s(\theta) = s + \theta v, \mu(\theta) = \frac{x(\theta)^t s(\theta)}{n}.$$

Tamanho do passo: Calcule

$$\lambda = \min\{1, \inf\{\theta > 0 \mid \delta_\infty(x(\theta), s(\theta), \mu(\theta)) \geq \alpha\}\}.$$

Resultado: $(x_+, s_+, \mu_+) := (x(\lambda), s(\lambda), \mu(\lambda))$.

4.2.2 O Algoritmo Corretor

Cada iteração do algoritmo corretor começa de um dado (x, s, μ) em \mathcal{N}_∞ . Pega-se uma seqüência dos passos de centralização de Newton, isto é, uma seqüência de passos de Newton, cada uma consistindo da solução de 2.2 associada com o valor de $\gamma = 1$, seguido de uma busca linear.

Devemos analisar dois aspectos importantes desse algoritmo: a busca linear e a regra de parada.

A medida de proximidade δ_∞ não nos fornece uma boa função mérito para o procedimento de busca, devido à sua não-diferenciabilidade. Para isto usaremos como função mérito a medida de proximidade Euclidiana.

E como regra de parada, estabeleceremos o seguinte algoritmo : dado um J fixo e um $\beta < \alpha$, pare em (x, s, μ) sempre que o número de iterações for igual a J ou $\delta(x, s, \mu) \leq \beta$.

Assim, a vizinhança grande pode ser interpretada como uma região de confiança para os passos de Newton.

Algoritmo 4.2.2 *Dados: (x^0, s^0, μ^0) tal que $\delta_\infty(x^0, s^0, \mu^0) = \alpha$, $\alpha > 0$, $\beta > 0$ e $J \geq 1$ ($J = +\infty$ é aceitável.)*

$k := 0$

REPITA

$(x, s, \mu) := (x^k, s^k, \mu^k)$

Newton: calcule a direção de centralização (u, v) resolvendo

$$\begin{aligned}su + xv &= -xs + \mu e \\Qu + Rv &= 0.\end{aligned}$$

Tamanho máximo do passo: calcule

$$\lambda_{max} = \min\{1, \inf\{\theta > 0 \mid \delta_\infty(x + \theta u, s + \theta v, \mu) \geq \alpha\}\}.$$

Busca linear: calcule

$$\lambda = \operatorname{argmin}\{\delta(x + \theta u, s + \theta v, \mu) \mid \theta \in [0, \lambda_{max}]\}.$$

Resultado: $(x^{k+1}, s^{k+1}, \mu^{k+1}) := (x + \lambda u, s + \lambda v, \mu)$.

k:=k+1.

ATÉ $k = J$ ou $\delta(x, s, \mu) \leq \beta$.

Resultado: $(\hat{x}, \hat{s}, \hat{\mu}) := (x^k, s^k, \mu)$.

Obs.: Na prática, usaremos o algoritmo 4.2.2 de duas formas distintas:

Correção: Neste caso daremos apenas um passo de centralização seguido de uma busca linear, ou seja, considerar $J = 1$. Esta é a forma que será utilizada no algoritmo 4.2.6, que será formalmente descrito mais à frente.

Centralização: Neste caso o critério de parada $\delta(x, s, \mu) \leq \beta$, para β suficientemente pequeno, será utilizado. Desta forma estaremos fazendo uma aproximação da trajetória central. Esta é a forma que será utilizada nos algoritmos 4.2.5 e 4.2.7.

4.2.3 O algoritmo Bord

Cada iteração do algoritmo começa de um dado $(x, s, \mu) \mathcal{N}_\infty$. Pega-se um passo ao longo da direção de Newton 2.2 associada com $\gamma \in (0, 1)$. Esta direção está intermediária entre a direção afim-escala e a direção de centralização.

O tamanho do passo é tal que o ponto resultante esteja na vizinhança de \mathcal{N}_∞ .

Algoritmo 4.2.3 Dados: (x^0, s^0, μ^0) tal que $\delta_\infty(x^0, s^0, \mu^0) \leq \alpha$, $\gamma \in (0, 1)$ e $\epsilon > 0$.

k:=0.

REPITA

$(x, s, \mu) := (x^k, s^k, \mu^k)$.

Calcule a direção (u, v) resolvendo as equações de Newton

$$\begin{aligned} su + xv &= -xs + \gamma\mu\epsilon \\ Qu + Rv &= 0. \end{aligned}$$

Busca: Calcule (x_+, s_+, μ_+) usando o Algoritmo de Busca 4.2.1 com $x, s, \mu, (u, v)$ e α dados.

Resultado: $(x^{k+1}, s^{k+1}, \mu^{k+1}) := (x_+, s_+, \mu_+)$.

$k := k + 1$.

ATÉ $\mu^k \leq \epsilon$.

Resultado: $(\hat{x}, \hat{s}, \hat{\mu}) := (x^k, s^k, \mu^k)$

Obs.: Este é um tipo de algoritmo que não faz uso do algoritmo 4.2.2. Ele apenas caminha em \mathcal{N}_∞ , normalmente por pontos na sua fronteira. Pode ocorrer que sejam gerados pontos (x, s, μ) com $\delta_\infty(x, s, \mu) < \alpha$; e isto ocorre quando o passo de Newton for muito eficiente.

4.2.4 O algoritmo Front

Cada iteração do algoritmo começa de um dado $(x, s, \mu) \in F^0$. Pega-se um passo ao longo da direção de Newton 2.2 associada com $\gamma \in (0, 1)$. Esta direção está intermediária entre a direção afim-escala e a direção de centralização.

O tamanho do passo é tal que o ponto resultante esteja em F^0 , próximo da fronteira do conjunto viável, ou seja, a fronteira de F .

Algoritmo 4.2.4 Dados: $(x^0, s^0, \mu^0) \in F^0$, $\gamma \in (0, 1)$ e $\tau \in (0, 1)$.

k:=0.

REPITA

$(x, s, \mu) := (x^k, s^k, \mu^k)$

Calcule a direção (u, v) resolvendo as equações de Newton

$$\begin{aligned}su + xv &= -xs + \gamma\mu e \\Qu + Rv &= 0.\end{aligned}$$

Tamanho do Passo: (Teste da razão) Calcule

$$\begin{aligned}\lambda_1 &= \min\left\{\frac{-x_i}{u_i} \mid u_i < 0, i = 1, \dots, n\right\} \\ \lambda_2 &= \min\left\{\frac{-s_i}{v_i} \mid v_i < 0, i = 1, \dots, n\right\} \\ \lambda &= \min\{1, \tau\lambda_1, \tau\lambda_2\}.\end{aligned}$$

$$\text{Resultado: } x^{k+1} = x + \lambda u, \quad s^{k+1} = s + \lambda v, \quad \mu^{k+1} = \frac{(x^{k+1})^t (s^{k+1})}{n}.$$

$$k := k + 1.$$

$$\text{ATÉ } \mu^k \leq \epsilon.$$

$$\text{Resultado: } (\hat{x}, \hat{s}, \hat{\mu}) := (x^k, s^k, \mu^k).$$

Obs.: Este é outro algoritmo, além do 4.2.3, que não faz uso de 4.2.2. Porém, ele caminha perto da fronteira de F .

4.2.5 O Algoritmo Intelc

Cada iteração do algoritmo começa de um dado $(x, s, \mu) \in \mathcal{N}_\infty$. Verifica-se se uma dada medida de proximidade $\delta^*(.) \leq \xi$ é atendida. Em caso positivo, pega-se um passo ao longo de uma dada direção (u, v) , como no algoritmo *Bord* 4.2.3. Caso contrário, pega-se uma seqüência dos passos de centralização de Newton.

Alguns aspectos desse algoritmo devem ser discutidos:

Usaremos o Indicador de Tapia 4.1.3 para estimar a partição ótima $\{\tilde{B}, \tilde{N}\}$, e com isso estimamos as variáveis grandes g e pequenas p , inclusive as variáveis pequenas centralizadas p^c .

Usaremos $\left\| \frac{gp^c}{\mu} - e \right\|$ como medida de proximidade, que é a medida descrita no Teorema 3.2.5 e 3.3.2. Esta está relacionada com a medida de proximidade primal das variáveis grandes δ_p dada por 3.2, a medida $\text{dist}(g, \mu)$ dada por 3.8 e o passo de centralização (u^c, v^c) .

Algoritmo 4.2.5 *Dados:* (x^0, s^0, μ^0) tal que $\delta_\infty(x^0, s^0, \mu^0) \leq \alpha$, $\gamma \in (0, 1)$, $\alpha > 0$, $\beta > 0$, $\epsilon > 0$ e $\xi > 0$.

$$k := 0.$$

$$\text{cent} := 0.$$

REPITA

$$(x, s, \mu) := (x^k, s^k, \mu^k).$$

$$\text{SE cent} == 0$$

Calcule a direção afim-escala (u^a, v^a) e a direção de centralização (u^c, v^c) resolvendo as equações de Newton

$$\begin{aligned} su + xv &= -xs + \gamma\mu e \\ Qu + Rv &= 0; \end{aligned}$$

para $\gamma = 0$ e $\gamma = 1$, respectivamente.

Usando o Indicador de Tapia 4.1.3, determine a partição $\{\tilde{B}, \tilde{N}\}$ (estimativa de $\{B, N\}$).

Defina $g = (x_{\tilde{B}}, s_{\tilde{N}})$.

Definimos:

$$x^c = x + u^c, \quad s^c = s + v^c.$$

Defina $p^c = (x_{\tilde{N}}^c, s_{\tilde{B}}^c)$.

SE $p^c > 0$ e $\left\| \frac{gp^c}{\mu} - e \right\| \leq \xi$;

Calcule a direção $(u, v) = \gamma(u^c, v^c) + (1 - \gamma)(u^a, v^a)$

Busca: Calcule (x_+, s_+, μ_+) usando o Algoritmo de Busca 4.2.1 com $x, s, \mu, (u, v)$ e α dados.

Resultado: $(x^{k+1}, s^{k+1}, \mu^{k+1}) := (x_+, s_+, \mu_+)$.

$cent := 0$.

SENÃO

Definimos as aplicações para $\theta \in \mathbb{R}$:

$$x(\theta) = x + \theta u^c, \quad s(\theta) = s + \theta v^c.$$

Tamanho do Passo: Calcule

$$\lambda = \min\{1, \inf\{\theta > 0 \mid \delta_\infty(x(\theta), s(\theta), \mu) \geq \alpha\}\}.$$

Resultado: $x^c = x(\lambda), \quad s^c = s(\lambda)$.

Corretor: Use o algoritmo 4.2.2 com os dados: (x^c, s^c, μ) , α e β dados, e $J = +\infty$, para calcular $(x^{k+1}, s^{k+1}, \mu^{k+1})$.

$cent := 1$.

SENÃO (O que significa x e s centrados)

Calcule a direção (u, v) resolvendo as equações de Newton

$$\begin{aligned}su + xv &= -xs + \gamma\mu e \\Qu + Rv &= 0;\end{aligned}$$

Busca: Calcule (x_+, s_+, μ_+) usando o Algoritmo de Busca 4.2.1 com $x, s, \mu, (u, v)$ e α dados.

Resultado: $(x^{k+1}, s^{k+1}, \mu^{k+1}) := (x_+, s_+, \mu_+)$.

$cent := 0$.

$k := k + 1$.

ATÉ $\mu^k \leq \epsilon$.

Resultado: $(\hat{x}, \hat{s}, \hat{\mu}) := (x^k, s^k, \mu^k)$.

Obs.: Se considerarmos parâmetros apropriados, podemos notar que este algoritmo atende as condições teóricas dadas pelo Teorema 3.3.3. Vejamos:

Dado $(x, s, \mu) \in \mathcal{N}_\infty$, onde $\alpha = 0.25$. Usando o Indicador de Tapia 4.1.3, podemos estimar as variáveis grandes g e pequenas p , inclusive as variáveis pequenas centradas p^c . Com esta informação podemos calcular a medida de proximidade primal $\delta(g, p^c, \mu)$ definida nos Teoremas 3.2.5 e 3.3.2 e desta forma obter uma aproximação da distância primal $dist(g, \mu)$ definida em 3.8. Tomamos $\xi = 0.136$ e usamos o Lema 3.3.2. Com isso podemos verificar a centralidade das variáveis grandes.

Consideremos, então, como primeiro caso, que elas estejam bem centradas, ou seja, $dist(g, \mu) < 0.1$. Segundo o Teorema 3.3.3, o passo de centralização de Newton é viável, isto é, $(x^c, s^c) \in \mathcal{N}_\infty$, onde $x^c = x + u^c$ e $s^c = s + v^c$, e haverá uma melhora significativa em $\delta(x^c, s^c, \mu)$ e também em $dist(g^c, \mu)$. O passo de centralização (u^c, v^c) neste caso é muito eficiente, tão eficiente quanto seria se $\delta(x, s, \mu)$ fosse pequeno (observe que $\delta_\infty(x, s, \mu)$ é pequeno, mas $\delta(x, s, \mu)$ pode ser grande). Logo, o passo longo de Newton na vizinhança grande terá todas as propriedades que teria na vizinhança pequena, não sendo necessário um passo corretor.

Suponhamos agora, como segundo caso, que as variáveis grandes não estejam bem centradas. Neste caso nada podemos dizer a respeito da eficiência do passo de Newton; pois não podemos

aplicar o Teorema 3.3.3. Isto significa que devemos realizar um passo corretor para amenizar este problema. Se realizamos uma centralização completa com base na medida de proximidade primal-dual, ou seja, $\delta(x, s, \mu)$, estamos melhorando todas as variáveis. Pelo Lema 3.3.1 vemos que há uma melhora significativa na distância primal das variáveis grandes $dist(g, \mu)$.

Resumindo: Com as hipóteses do Teorema 3.3.3 satisfeitas, ou seja, um ponto dentro da vizinhança grande ($\delta_\infty(x, s, \mu) \leq 0.25$) e as variáveis grandes centradas ($dist(g, \mu) \leq 0.1$), temos bons passos de centralização com um decréscimo em $\delta(x^c, s^c, \mu)$ e também em $dist(g^c, \mu)$. Isto significa que a cada iteração o passo de Newton ficará mais eficiente. Exatamente como foi provado pelo Teorema.

Mas é interessante saber se esses resultados também podem ser obtidos mesmo com o não-cumprimento das hipóteses?

Veremos mais adiante na seção *Experimentos Numéricos* do próximo capítulo que, mesmo relaxando as hipóteses do Teorema, é possível obter bons resultados no que diz respeito às suas teses.

4.2.6 O Algoritmo Intel

Cada iteração do algoritmo começa de um dado $(x, s, \mu) \in \mathcal{N}_\infty$. Verifica-se se uma dada medida de proximidade $\delta^*(.) \leq \xi$ é atendida. Em caso positivo, pega-se um passo ao longo de uma dada direção (u, v) como no algoritmo *Bord* 4.2.3. Caso contrário, pega-se um único passo de centralização de Newton, em contraste com o algoritmo *Intelc* 4.2.5, que pega uma seqüência dos passos de centralização de Newton.

A medida $\delta^*(.)$ tem o mesmo aspecto discutido para o algoritmo 4.2.5.

Algoritmo 4.2.6 *Dados:* (x^0, s^0, μ^0) tal que $\delta_\infty(x^0, s^0, \mu^0) \leq \alpha$, $\gamma \in (0, 1)$, $\alpha > 0$, $\beta > 0$, $\epsilon > 0$ e $\xi > 0$.

$k := 0$.

$cent := 0$.

REPITA

$(x, s, \mu) := (x^k, s^k, \mu^k)$.

SE $cent == 0$

Calcule a direção afim-escala (u^a, v^a) e a direção de centralização (u^c, v^c) resolvendo as equações de Newton

$$\begin{aligned}su + xv &= -xs + \gamma\mu e \\Qu + Rv &= 0;\end{aligned}$$

para $\gamma = 0$ e $\gamma = 1$, respectivamente.

Usando o Indicador de Tapia 4.1.3 determine a partição $\{\tilde{B}, \tilde{N}\}$ (estimativa de $\{B, N\}$).

Defina $g = (x_{\tilde{B}}, s_{\tilde{N}})$.

Definimos:

$$x^c = x + u^c, \quad s^c = s + v^c.$$

Defina $p^c = (x_{\tilde{N}}^c, s_{\tilde{B}}^c)$.

SE $p^c > 0$ e $\left\| \frac{gp^c}{\mu} - e \right\| \leq \xi$;

Calcule a direção $(u, v) = \gamma(u^c, v^c) + (1 - \gamma)(u^a, v^a)$

Busca: Calcule (x_+, s_+, μ_+) usando o Algoritmo de Busca 4.2.1 com $x, s, \mu, (u, v)$ e α dados.

Resultado: $(x^{k+1}, s^{k+1}, \mu^{k+1}) := (x_+, s_+, \mu_+)$.

$cent := 0$.

SENÃO

Definimos as aplicações para $\theta \in \mathbb{R}$:

$$x(\theta) = x + \theta u^c, \quad s(\theta) = s + \theta v^c.$$

Tamanho do Passo: Calcule

$$\lambda = \min\{1, \inf\{\theta > 0 \mid \delta_\infty(x(\theta), s(\theta), \mu) \geq \alpha\}\}.$$

Defina: $x^c = x(\lambda), \quad s^c = s(\lambda)$.

Resultado: $(x^{k+1}, s^{k+1}, \mu^{k+1}) := (x^c, s^c, \mu)$.

$cent := 1$.

SENÃO (O que significa x e s centrados)

Calcule a direção (u, v) resolvendo as equações de Newton

$$su + xv = -xs + \gamma\mu e$$

$$Qu + Rv = 0;$$

Busca: Calcule (x_+, s_+, μ_+) usando o Algoritmo de Busca 4.2.1 com $x, s, \mu, (u, v)$ e α dados.

Resultado: $(x^{k+1}, s^{k+1}, \mu^{k+1}) := (x_+, s_+, \mu_+)$.

$cent := 0$.

$k := k + 1$.

ATÉ $\mu^k \leq \epsilon$.

Resultado: $(\hat{x}, \hat{s}, \hat{\mu}) := (x^k, s^k, \mu^k)$.

Obs.: Neste algoritmo temos apenas uma correção, ou seja, apenas um passo no processo de centralização; o que significa que nem sempre se obtém uma centralização adequada. Logo, não se tem a garantia de pontos bem centrados, e, conseqüentemente, variáveis grandes bem centradas.

Veremos mais adiante na seção *Experimentos Numéricos* do próximo capítulo que, mesmo não havendo a garantia da boa centralização das variáveis grandes, ainda assim obtemos bons resultados; o que significa que o efeito da centralização é acumulativo. Pois a centralização não piora, ou piora pouco, as variáveis grandes, conforme vimos no início deste capítulo com os *Indicadores de Tapia*.

4.2.7 O Algoritmo Largec

Cada iteração do algoritmo começa de um dado $(x, s, \mu) \in \mathcal{N}_\infty$. Verifica-se se uma dada medida de proximidade $\delta^*(.) \leq \xi$ é atendida. Em caso positivo, pega-se um passo ao longo de uma dada direção (u, v) tal que o ponto resultante esteja em \mathcal{N}_∞ . Caso contrário, pega-se uma seqüência dos passos de centralização de Newton.

Alguns aspectos desse algoritmo devem ser discutidos:

A estimativa da partição ótima e, necessariamente, a estimativa das variáveis grandes g e pequenas p , ocorrerá de maneira idêntica aos algoritmos anteriores, ou seja, fazendo uso do Indicador de Tapia 4.1.3.

A medida $\delta^*(.)$ tem o mesmo aspecto discutido para o algoritmo *Intelc* 4.2.5.

Um aspecto interessante, talvez o mais importante, é a forma de tomar a direção (u, v) .

Dada a direção afim-escala (u^a, v^a) , obtemos o ponto (x^a, s^a) , onde $x^a = x + u^a$ e $s^a = s + v^a$. Obtemos de maneira análoga (x^c, s^c) , ou seja, $x^c = x + u^c$ e $s^c = s + v^c$, onde (u^c, v^c) é a direção de centralização. Com isso criamos a direção $(u, v) = (x^a - x^c, s^a - s^c)$ que nos fornece, a partir de (x^c, s^c) , o algoritmo com o maior passo.

Algoritmo 4.2.7 *Dados:* (x^0, s^0, μ^0) tal que $\delta_\infty(x^0, s^0, \mu^0) \leq \alpha$, $\alpha > 0$, $\beta > 0$, $\epsilon > 0$ e $\xi > 0$.

$k := 0$.

REPITA

$(x, s, \mu) := (x^k, s^k, \mu^k)$.

Calcule a direção afim-escala (u^a, v^a) e a direção de centralização (u^c, v^c) , resolvendo as equações de Newton

$$su + xv = -xs + \gamma\mu e$$

$$Qu + Rv = 0;$$

para $\gamma = 0$ e $\gamma = 1$, respectivamente.

Definimos:

$$x^a = x + u^a, \quad s^a = s + v^a;$$

$$x^c = x + u^c, \quad s^c = s + v^c.$$

Usando o Indicador de Tapia 4.1.3 determine a partição $\{\tilde{B}, \tilde{N}\}$ (estimativa de $\{B, N\}$).

Defina $g = (x_{\tilde{B}}, s_{\tilde{N}})$ e $p^c = (x_{\tilde{N}}^c, s_{\tilde{B}}^c)$.

SE $p^c > 0$ e $\left\| \frac{gp^c}{\mu} - e \right\| \leq \xi$;

Calcule a direção $(u, v) = (x^a - x^c, s^a - s^c)$

Busca: Calcule (x_+, s_+, μ_+) usando o Algoritmo de Busca 4.2.1 com $x^c, s^c, \mu, (u, v)$ e α dados.

Resultado: $(x^{k+1}, s^{k+1}, \mu^{k+1}) := (x_+, s_+, \mu_+)$.

SENÃO

Definimos as aplicações para $\theta \in \mathbb{R}$:

$$x(\theta) = x + \theta u^c, \quad s(\theta) = s + \theta v^c.$$

Tamanho do Passo: Calcule

$$\lambda = \min\{1, \inf\{\theta > 0 \mid \delta_\infty(x(\theta), s(\theta), \mu) \geq \alpha\}\}.$$

Resultado: $x^c = x(\lambda)$, $s^c = s(\lambda)$.

Corretor: Use o algoritmo 4.2.2 com os dados: (x^c, s^c, μ) , α e β dados, e $J = +\infty$, para calcular $(x^{k+1}, s^{k+1}, \mu^{k+1})$.

$k := k + 1$.

ATÉ $\mu^k \leq \epsilon$.

Resultado: $(\hat{x}, \hat{s}, \hat{\mu}) := (x^k, s^k, \mu^k)$.

Obs.: Se considerarmos parâmetros apropriados, podemos notar que este algoritmo, de modo análogo ao algoritmo *Intelc* 4.2.5, atende as condições teróricas dadas pelo Teorema 3.3.3.

Mas será que esses resultados também são obtidos se as hipóteses não se verificam?

Na seção *Experimentos Numéricos* do próximo capítulo, veremos que a resposta é sim, isto é, mesmo relaxando as hipóteses do Teorema, consegue-se bons resultados.

4.2.8 O Algoritmo Large

Cada iteração do algoritmo começa de um dado $(x, s, \mu) \in \mathcal{N}_\infty$.

Dada a direção afim-escala (u^a, v^a) , obtemos o ponto (x^a, s^a) , onde $x^a = x + u^a$ e $s^a = s + v^a$. De forma análoga ao algoritmo *Intel* 4.2.6, podemos obter (x^c, s^c) com apenas um passo de centralização seguido de uma busca linear. Com isso criamos a direção $(u, v) = (x^a - x^c, s^a - s^c)$. O tamanho do passo, a partir de (x^c, s^c) , é tal que o ponto resultante esteja na vizinhança de \mathcal{N}_∞ . E isto nos fornece o algoritmo com o maior passo.

Algoritmo 4.2.8 Dados: (x^0, s^0, μ^0) tal que $\delta_\infty(x^0, s^0, \mu^0) \leq \alpha$, $\alpha > 0$, $\beta > 0$ e $\epsilon > 0$.

$k := 0$.

REPITA

$(x, s, \mu) := (x^k, s^k, \mu^k)$.

Calcule a direção afim-escala (u^a, v^a) e a direção de centralização (u^c, v^c) , resolvendo as equações de Newton

$$\begin{aligned} su + xv &= -xs + \gamma\mu e \\ Qu + Rv &= 0; \end{aligned}$$

para $\gamma = 0$ e $\gamma = 1$, respectivamente.

Definimos:

$$x^a = x + u^a, \quad s^a = s + v^a.$$

Definimos as aplicações para $\theta \in \mathbb{R}$:

$$x(\theta) = x + \theta u^c, \quad s(\theta) = s + \theta v^c.$$

Tamanho do Passo: Calcule

$$\lambda = \min\{1, \inf\{\theta > 0 \mid \delta_\infty(x(\theta), s(\theta), \mu) \geq \alpha\}\}.$$

Resultado: $x^c = x(\lambda)$, $s^c = s(\lambda)$.

Calcule a direção $(u, v) = (x^a - x^c, s^a - s^c)$

Busca: Calcule (x_+, s_+, μ_+) usando o Algoritmo de Busca 4.2.1 com $x^c, s^c, \mu, (u, v)$ e α dados.

Resultado: $(x^{k+1}, s^{k+1}, \mu^{k+1}) := (x_+, s_+, \mu_+)$.

$k := k + 1$.

ATÉ $\mu^k \leq \epsilon$.

Resultado: $(\hat{x}, \hat{s}, \hat{\mu}) := (x^k, s^k, \mu^k)$.

Obs.: Neste algoritmo não temos passos de correção. Logo, não se tem a garantia de pontos bem centrados e, conseqüentemente, variáveis grandes bem centradas.

Na seção *Experimentos Numéricos* do capítulo que vem logo a seguir, veremos que mesmo não havendo a garantia da boa centralização das variáveis grandes, ainda assim obtemos bons resultados, conforme descrito também para o Algoritmo *Intel 4.2.6*; confirmando que o efeito da centralização é acumulativo, não piorando, ou piorando pouco, as variáveis grandes.

Capítulo 5

Experimentos numéricos e Conclusões

5.1 Experimentos numéricos

Experimentos. Para analisar o comportamento dos algoritmos foram efetuados vários experimentos numéricos. Os testes foram realizados em um PENTIUM 133 MHz com sistema operacional WINDOWS 95 com 16 Mb RAM. Os programas foram implementados em MATLAB.

O problema teste utilizado, que descreveremos mais adiante, foi obtido pelo gerador desenvolvido por C. C. Gonzaga.

Consideremos então os programas, ambos programados em MATLAB, como **Caixa Preta**: LPDATA e GENLP.

O primeiro, ou seja, LPDATA, fornece dados de uma família de problemas, tais como: dimensão do problema (variáveis e restrições), dimensão ou degeneração das faces ótimas primal e dual, viabilidade do ponto inicial, etc. Com esses dados, o segundo programa, isto é, GENLP, gera aleatoriamente um problema que possui estas características e fornece um ponto viável inicial.

Este ponto (x, s) é sempre (e, e) , onde $e = (1, \dots, 1)$, com a dimensão dada de acordo com o problema.

Em todos os testes, consideramos o raio da vizinhança como sendo $\alpha = 0.9$. E para efeito de centralização $\beta = 0.5$. No que diz respeito à aproximação da trajetória central, o critério de

parada para a centralização é $\beta_1 = 10^{-5}$.

Na implementação dos algoritmos, outros parâmetros foram definidos:

$$\epsilon = 10^{-5}, \quad \gamma = 0.2, \quad \xi = 0.5 \quad \text{e} \quad \tau = 0.99.$$

O critério principal de parada dos algoritmos é que o "gap de complementariedade" seja menor ou igual ao valor de ϵ , ou seja, $x^t s \leq \epsilon$.

Problemas testes. Nesta seção vamos analisar, em particular, um problema que será resolvido com os algoritmos estudados; e este problema foi gerado pelo GENLP. Para isto temos que fornecer suas características principais:

- i) Restrições: 20;
- ii) Variáveis: 40;
- iii) Dimensão da face ótima primal: 3;
- iv) Dimensão da face ótima dual: 1;

Primeiramente vamos aplicar os algoritmos *Front* e *Intel*, ou seja, algoritmos que não atendem às condições teóricas dadas pelo Teorema 3.3.3, e assim, obter os seguintes resultados:

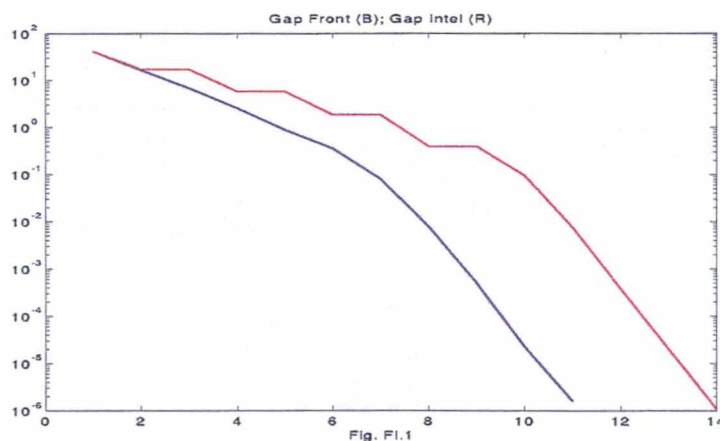
Número necessário de iterações para se resolver o problema usando o algoritmo *Front*: 10

Número necessário de iterações para se resolver o problema usando o algoritmo *Intel*: 13

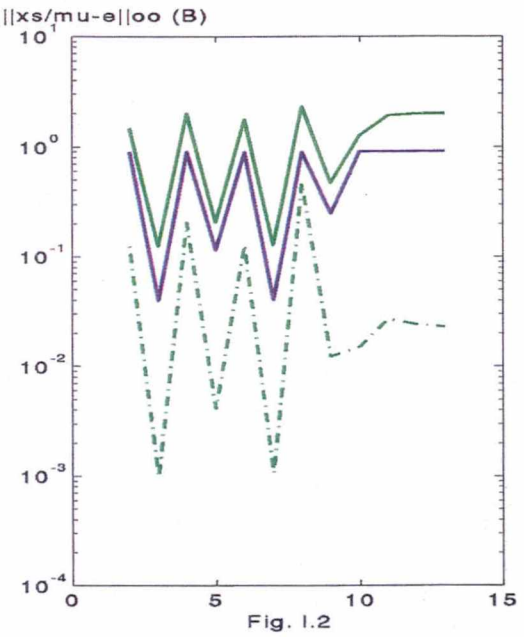
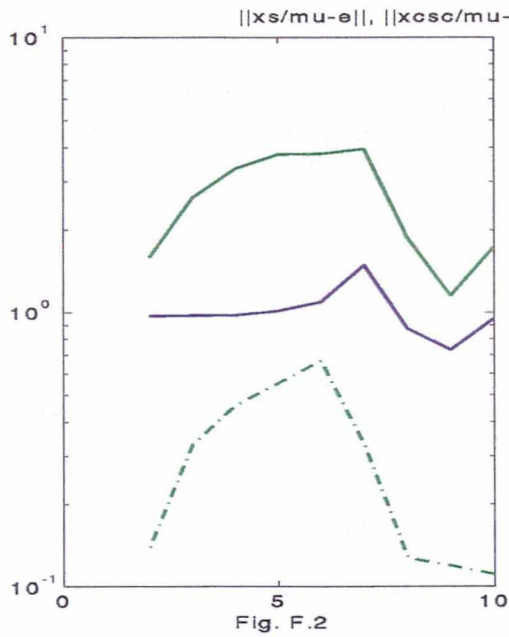
E com as saídas gráficas que veremos logo a seguir.

Obs.: Os gráficos da esquerda (*F.**) são referentes ao algoritmo *Front*; e os da direita (*I.**) são referentes ao algoritmo *Intel*.

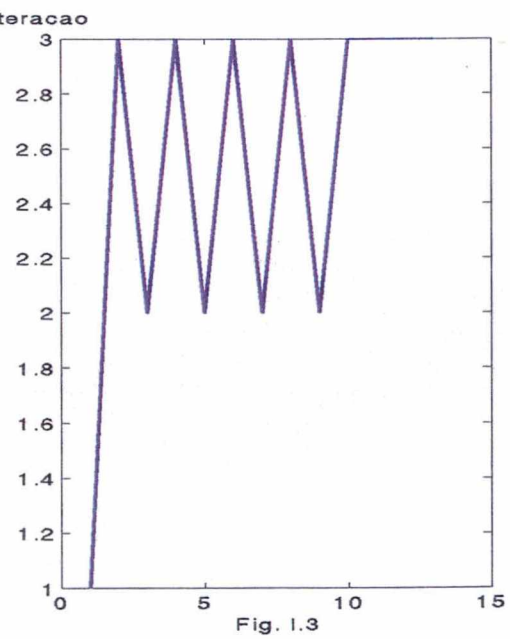
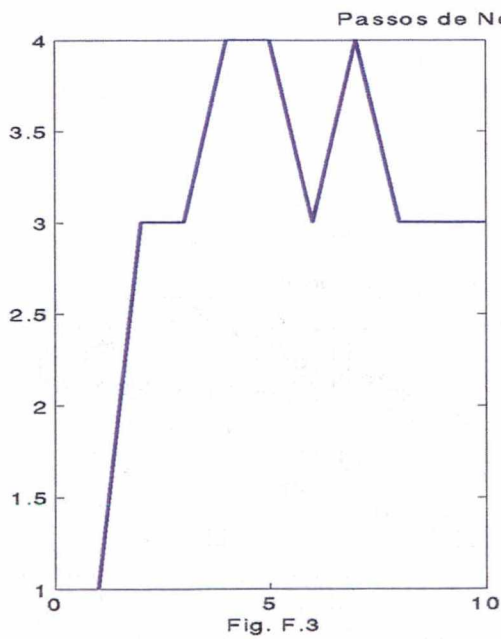
Na figura 1 temos o comportamento do gap de complementariedade, ou seja, $x^t s$. Em linha azul para o algoritmo *Front* e em linha vermelha para o *Intel*.



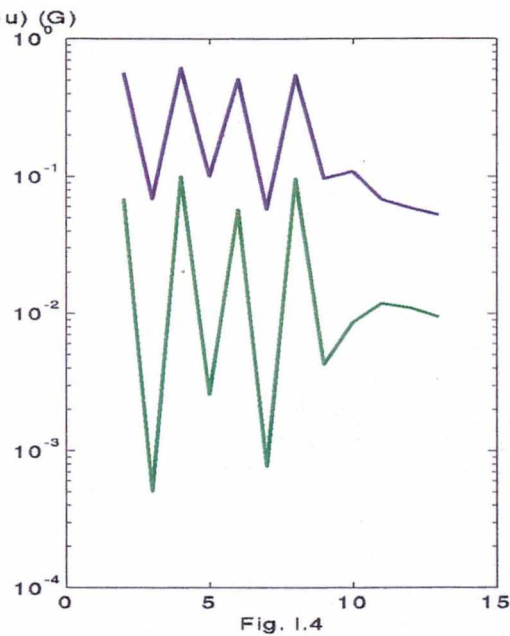
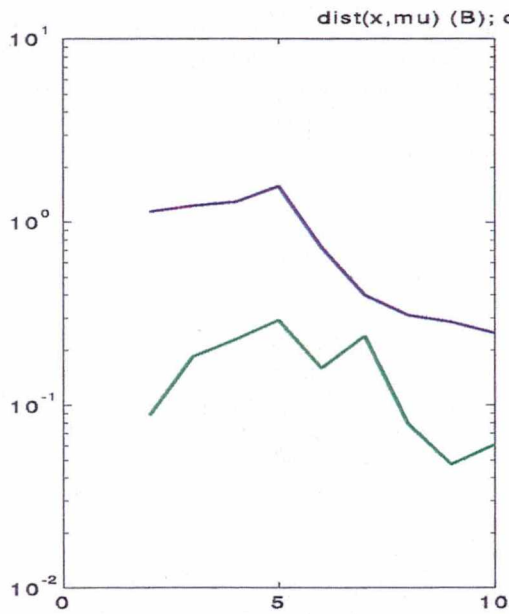
Na figura 2 temos em linha azul $\delta_\infty(x, s, \mu) = \left\| \frac{xs}{\mu} - e \right\|_\infty$. Em linha verde temos $\delta(x, s, \mu) = \left\| \frac{xs}{\mu} - e \right\|$, e em linha verde pontilhada $\delta(x^c, s^c, \mu) = \left\| \frac{x^c s^c}{\mu} - e \right\|$.



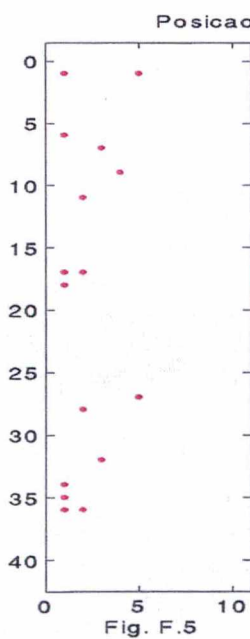
Na figura 3 temos os números de passos de Newton que seriam necessários para encontrar a aproximação da trajetória central, isto é, uma centralização cujo critério de parada é $\delta(x, s, \mu) \leq \beta_1$, que é um bom critério.



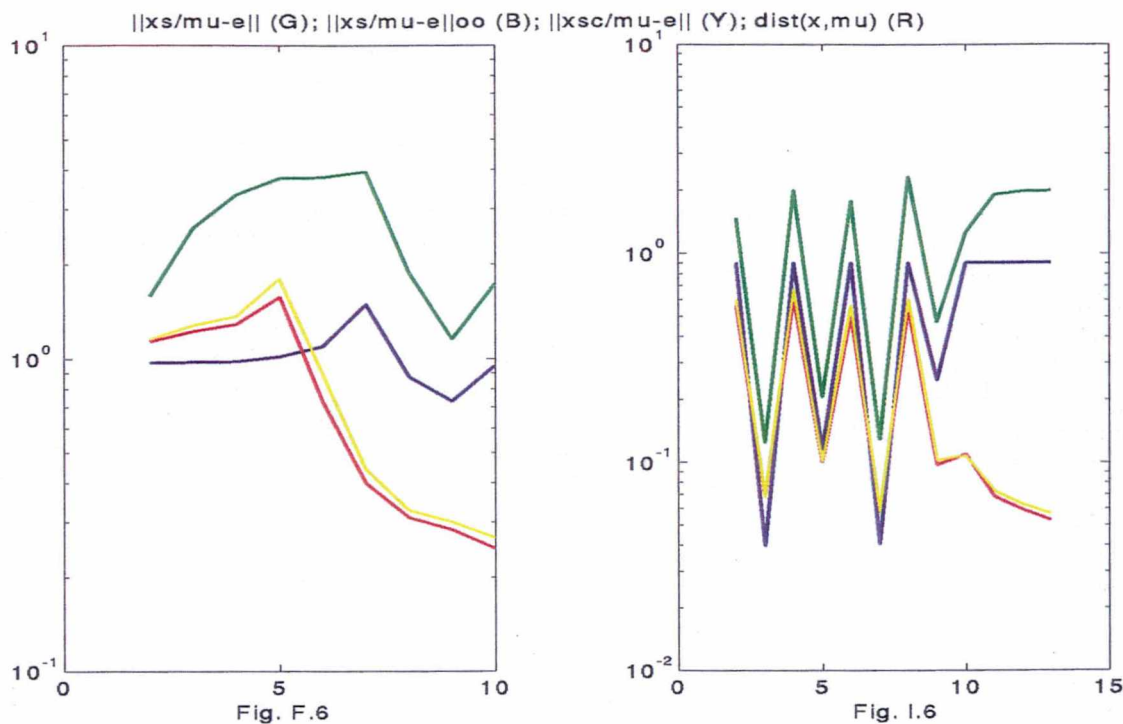
Na figura 4 temos em linha azul $dist(x, \mu) = \left\| \frac{x-x(\mu)}{x(\mu)} \right\|$, que é a distância primal definida em 3.8; e em linha verde $dist(x^c, \mu) = \left\| \frac{x^c-x(\mu)}{x(\mu)} \right\|$, que é a mesma distância, porém para as variáveis grandes centradas.



Na figura 5 temos a disposição dos erros cometidos em utilizar o Indicador de Tapia como forma de identificação das restrições ativas e inativas na solução ótima, ou seja, as variáveis grandes e pequenas.



Na figura 6 temos um comparativo de todas as medidas utilizadas nos algoritmos. Temos em linha azul $\delta_\infty(x, s, \mu) = \left\| \frac{xs}{\mu} - e \right\|_\infty$. Em linha verde temos $\delta(x, s, \mu) = \left\| \frac{xs}{\mu} \right\|$. Em linha amarela $\delta(x, s^c, \mu) = \left\| \frac{xs^c}{\mu} - e \right\|$, que é a medida de proximidade primal definida pelo Lema 3.2.5; e finalmente em linha vermelha $dist(x, \mu) = \left\| \frac{x-x(\mu)}{x(\mu)} \right\|$.



Vamos resolver o mesmo problema aplicando, agora, os algoritmos *Bord* e *Intelc*. O primeiro não cumpre as hipóteses do Teorema 3.3.3; e o segundo efetua eventuais centralizações mediante o teste $\left\| \frac{xs^c}{\mu} - e \right\| \leq \xi$, e conseqüentemente atendendo às condições do respectivo Teorema, e assim obter o seguinte resultado:

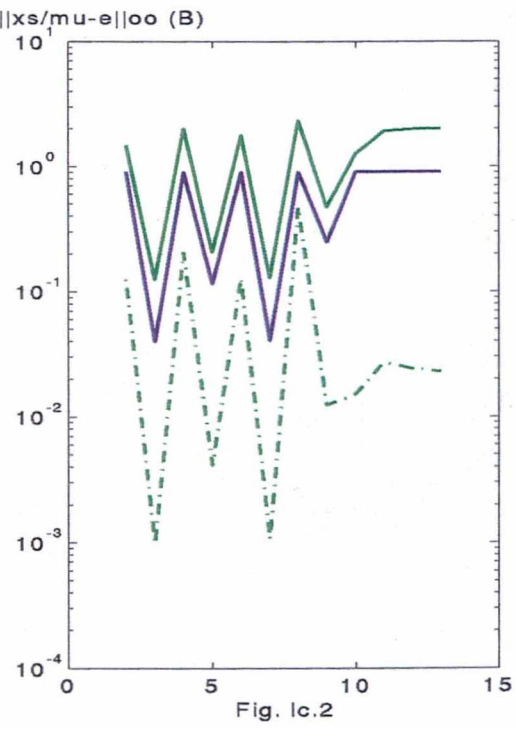
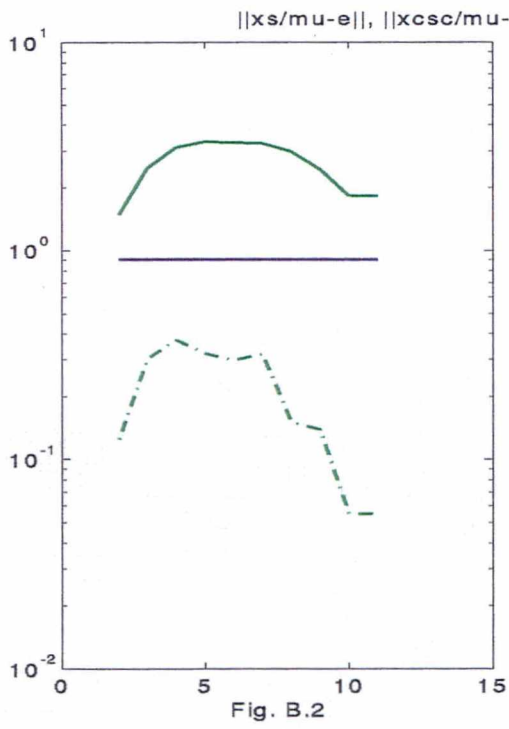
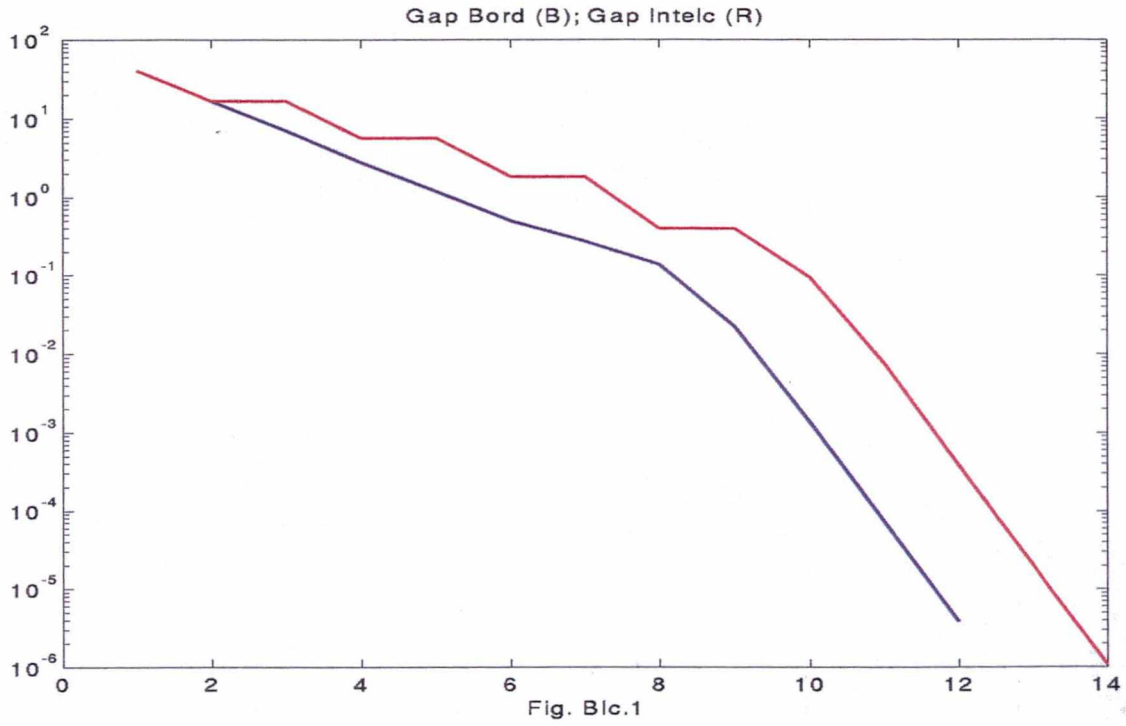
Número necessário de iterações para o algoritmo *Bord*: 11

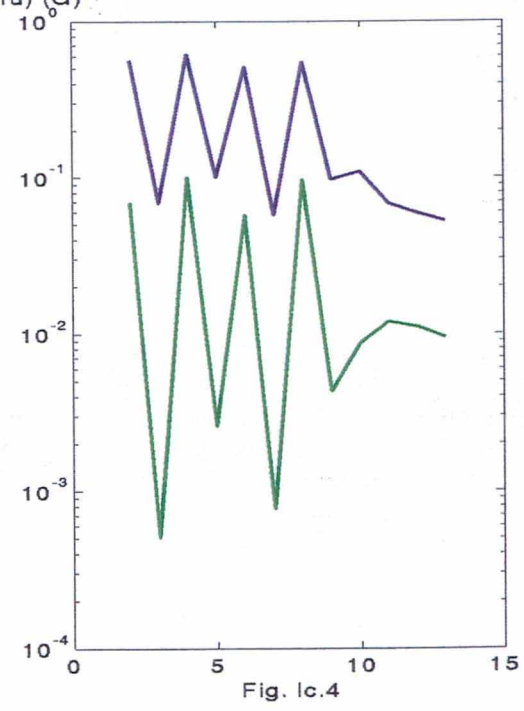
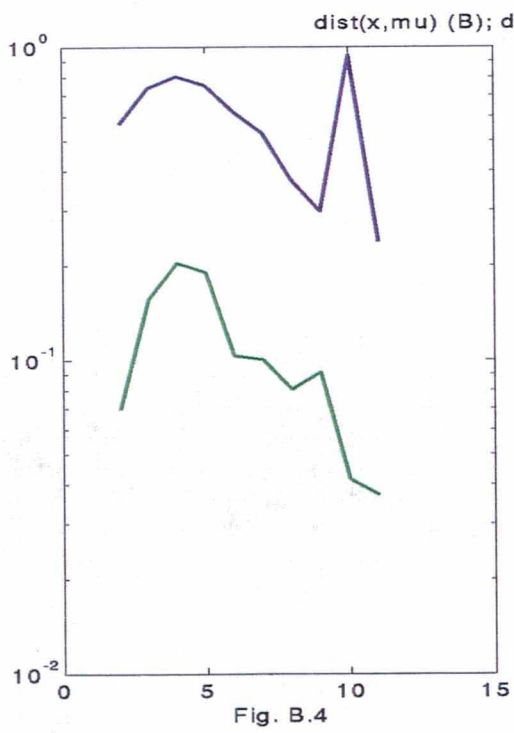
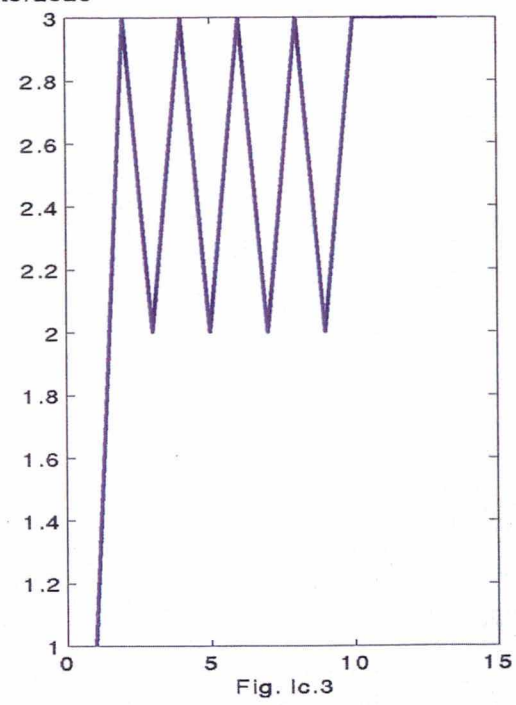
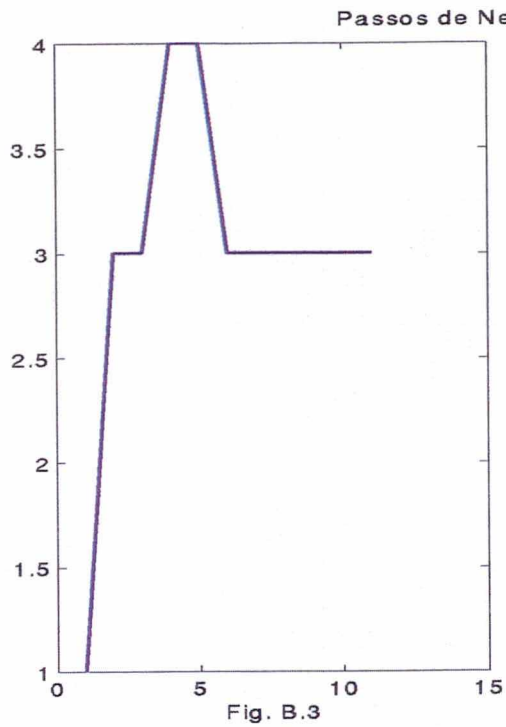
Número necessário de iterações para o algoritmo *Intelc*: 13

E as seguintes saídas gráficas com as mesmas interpretações dadas para o problema, quando este foi resolvido pelos algoritmos *Front* e *Intel*.

Obs.: Os gráficos da esquerda (*B.**) são referentes ao algoritmo *Bord*; e os da direita (*Ic.**)

são referentes ao algoritmo *Intelc*.





Posicao dos Erros: Indicador de Tapia

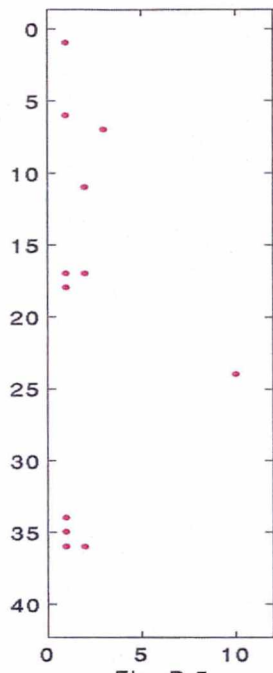


Fig. B.5



Fig. lc.5

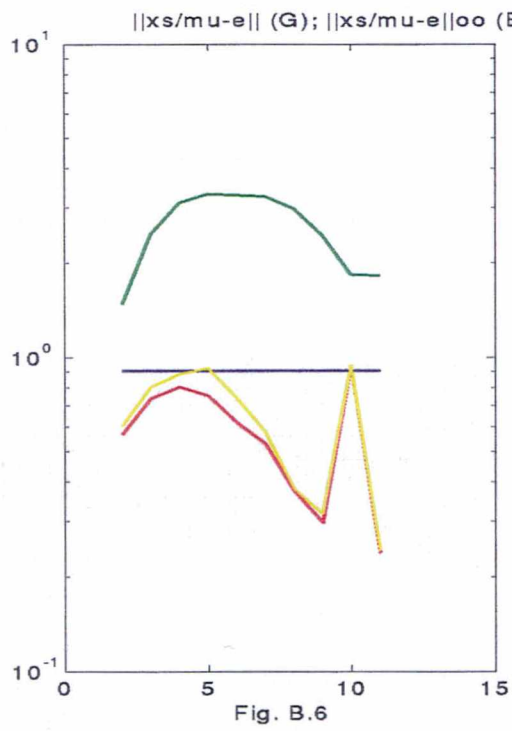


Fig. B.6

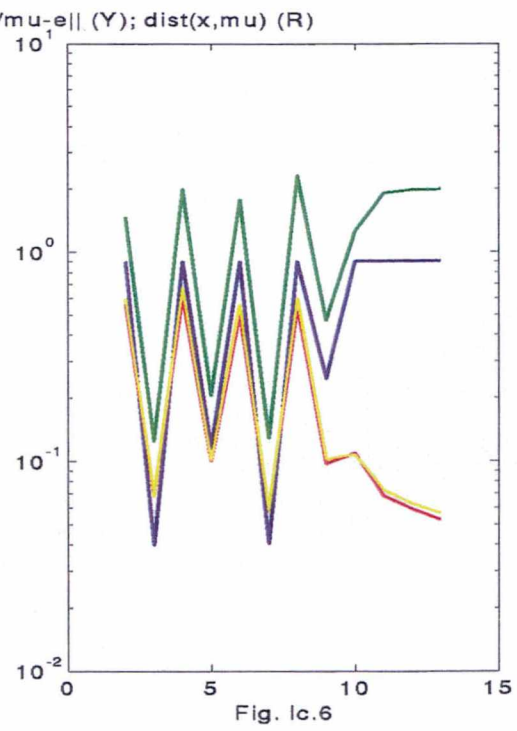


Fig. lc.6

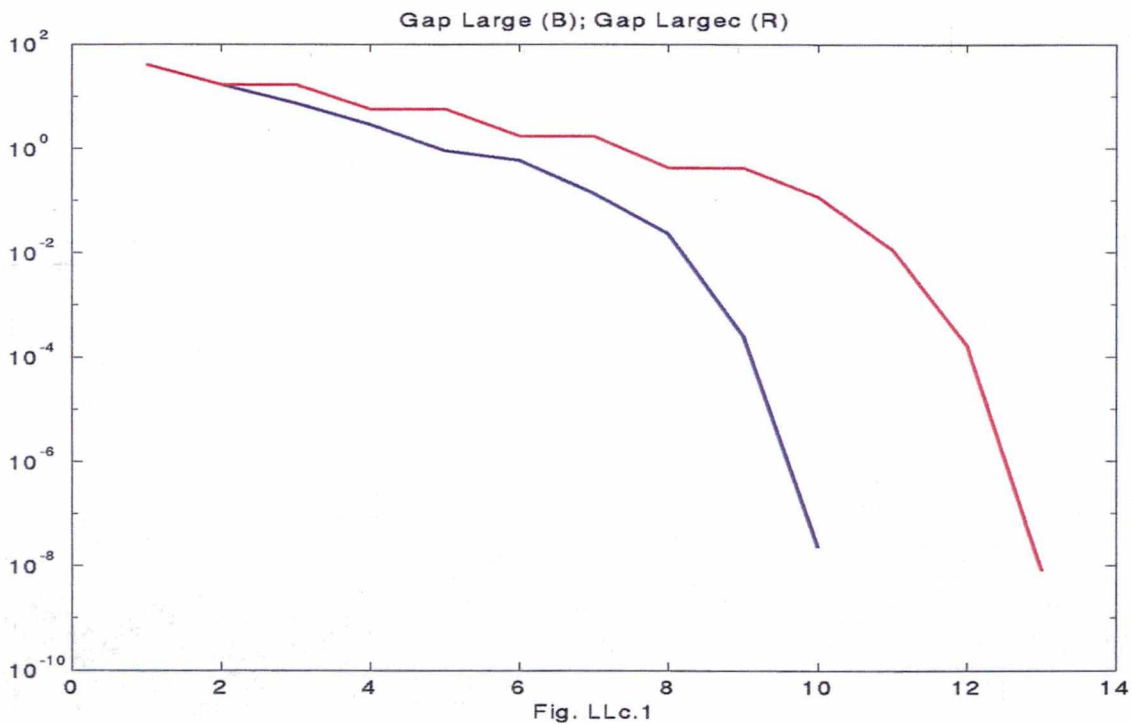
Finalmente vamos resolver o mesmo problema, porém aplicando os algoritmos *Large* e *Largec*. Ambos algoritmos realizam o maior passo, ou seja, ganho computacional. Sendo que o primeiro não realiza centralizações e conseqüentemente não atende às condições do Teorema 3.3.3, e o segundo efetua eventuais centralizações mediante o teste $\left\| \frac{xs^c}{\mu} - e \right\| \leq \xi$, atendendo assim às hipóteses do respectivo Teorema; e obtendo o seguinte resultado:

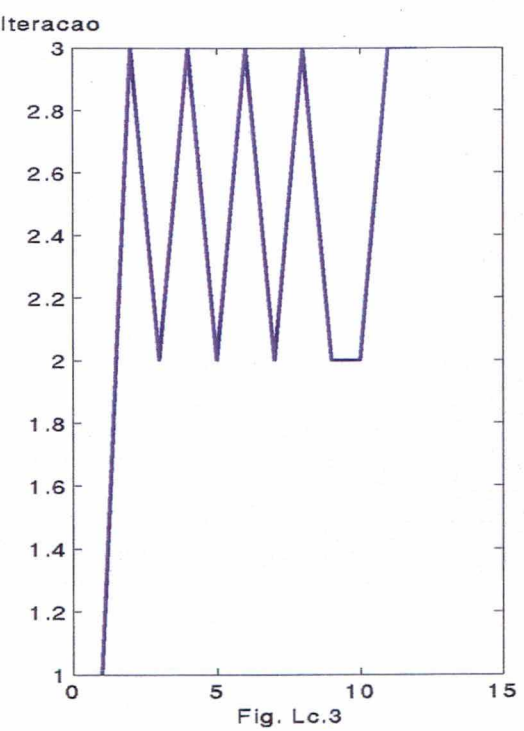
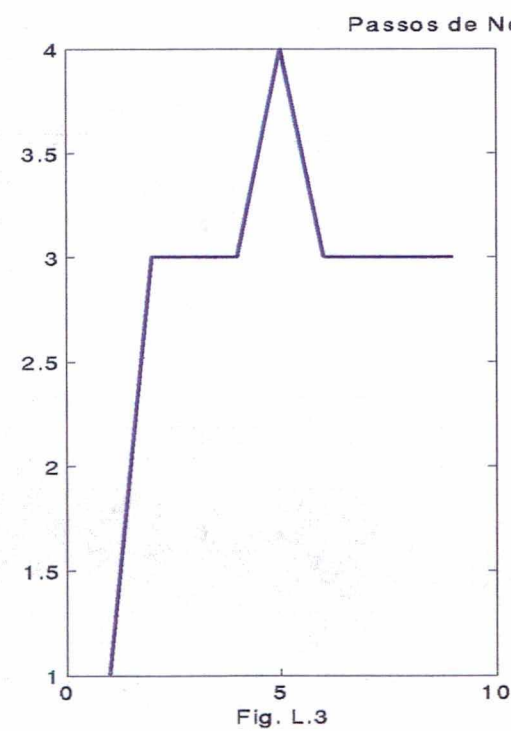
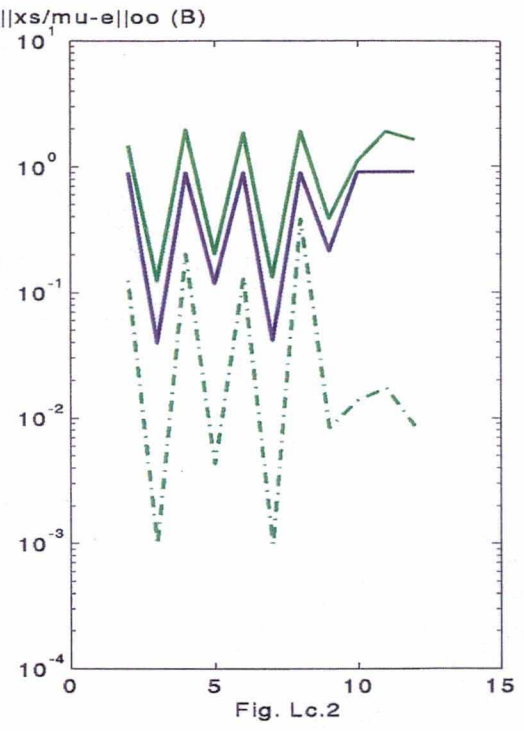
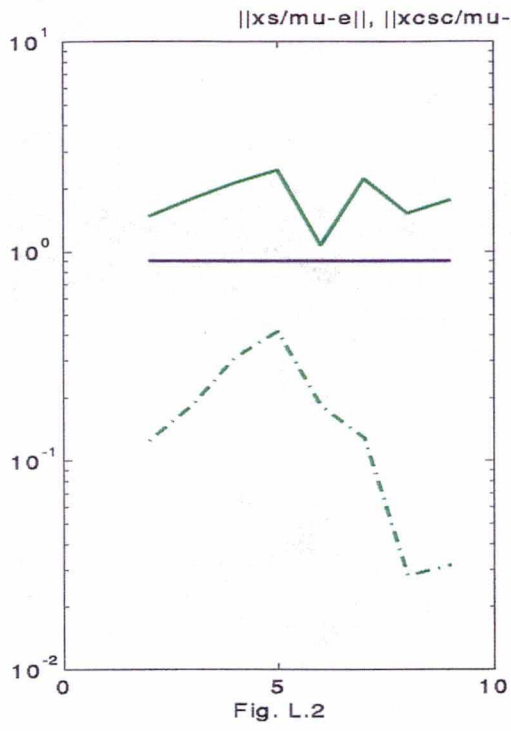
Número necessário de iterações para o algoritmo *Large*: 9

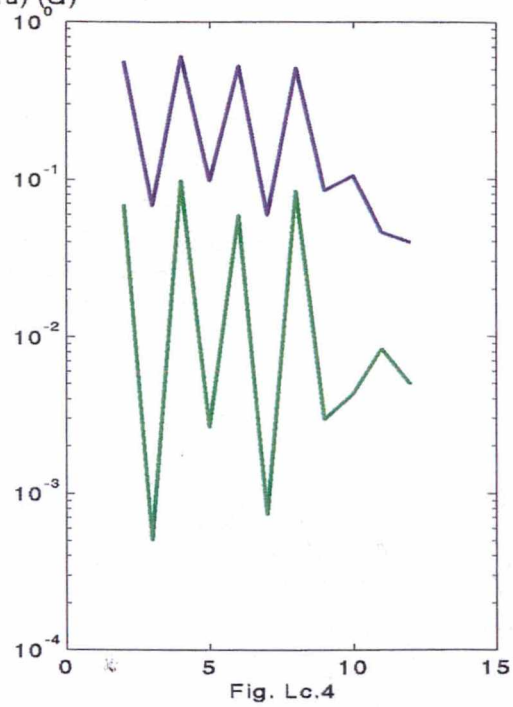
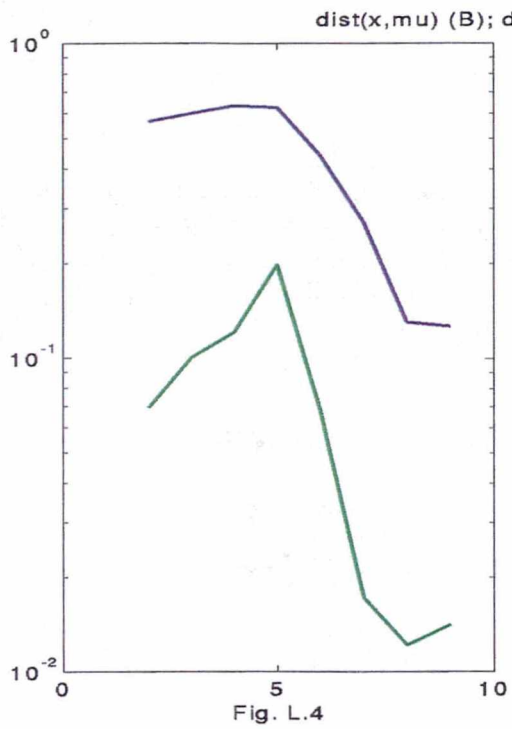
Número necessário de iterações para o algoritmo *Largec*: 12

E as seguintes saídas gráficas com as mesmas interpretações dadas para o problema, quando este foi resolvido pelos quatro algoritmos anteriores.

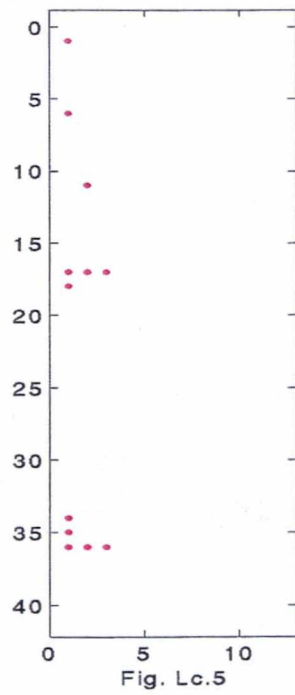
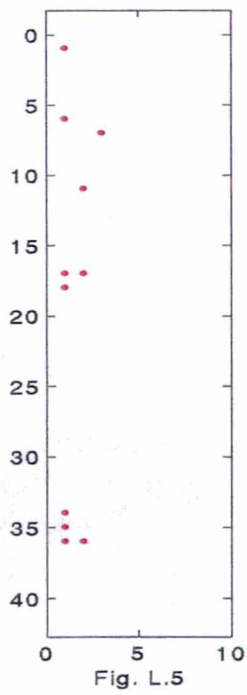
Obs.: Os gráficos da esquerda (*L.**) são referentes ao algoritmo *Large*; e os da direita (*Lc.**) são referentes ao algoritmo *Largec*.

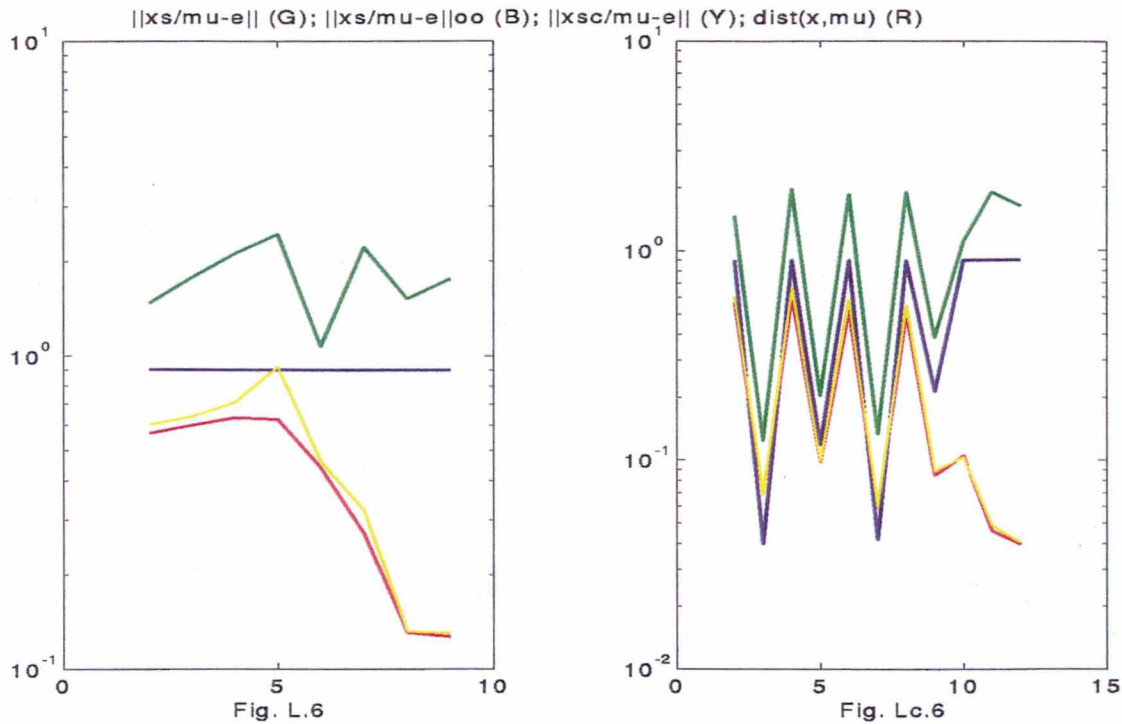






Posicao dos Erros: Indicador de Tapia





Nota: Na realização dos experimentos numéricos agrupamos os algoritmos de acordo com alguns critérios para facilitar nossa análise.

A primeira bateria de testes foi feita com os algoritmos *Front* e *Intel*. O primeiro algoritmo não cumpre às hipóteses do Teorema 3.3.3, ou seja, não atende a nenhuma condição teórica. O segundo algoritmo realiza apenas um passo corretor quando há necessidade (segundo o critério de proximidade primal), mas esse passo não garante boa centralização; o que significa que ele atende parcialmente às condições teóricas.

A segunda bateria de testes foi feita com os algoritmos *Bord* e *Intelc*. Analogamente à primeira bateria, o primeiro algoritmo não atende às condições teóricas dadas pelo Teorema 3.3.3. O segundo realiza passos corretores quando há necessidade, e esses passos corretores realizam uma aproximação à trajetória com um bom critério de parada; o que significa que atende às condições teóricas.

A terceira e última bateria de testes foi feita com os algoritmos *Large* e *Largec*. Novamente temos o primeiro algoritmo que não atende às condições teóricas e o segundo sim.

5.2 Conclusões

Os objetivos dos testes da seção anterior foram de verificar:

I) Eficiência do Indicador de Tapia na identificação das variáveis grandes e pequenas; já que alguns algoritmos fazem uso destes dados.

II) Desempenho dos Algoritmos para resolver problemas com características distintas.

III) Propriedades do Teorema 3.3.3.

Para a realização de nossa análise, vamos agrupar os algoritmos de acordo com outro critério distinto ao adotado para os *Experimentos Numéricos*.

Vamos agrupá-los por perfil, ou seja, algoritmos que possuem o mesmo perfil são algoritmos que realizam as iterações semelhantemente no que diz respeito à atualização do μ , forma de tomar a direção (u, v) , etc.

Assim teremos como primeiro grupo apenas o algoritmo *Front*, que é um algoritmo diferenciado dos demais.

Constituindo o segundo grupo temos os algoritmos *Intel*, *Bord* e *Intelc*.

E, por fim, temos os algoritmos *Large* e *Largec* constituindo o terceiro grupo.

Pelos testes realizados, podemos chegar às seguintes conclusões:

5.2.1 Eficiência do Indicador de Tapia

Temos visto em [2] a eficiência comprovada do Indicador de Tapia para Problemas de Programação Linear e sua extensão para Métodos de Pontos Interiores.

Temos visto também na seção 4.1 como identificar as variáveis grandes e pequenas usando esses Indicadores.

Pelo Teorema 4.1.1 a direção afim-escala (u^a, v^a) é a direção fundamental para essa identificação. Então, quando avaliamos as iterações onde o Indicador de Tapia foi eficiente, estamos apenas analisando as iterações onde foi realizado o passo preditor.

Agora vamos fazer essa análise para o nosso problema concreto.

A Fig. 5 (*F.5* e *I.5*) mostra a posição dos erros do Indicador de Tapia com o objetivo de identificar as variáveis grandes e pequenas para os algoritmos *Front* e *Intel*.

Temos 16 erros num total de 10 iterações para o algoritmo *Front*; o que acarreta em 4.0 % de erros na identificação.

Obs.: Note que todas as iterações deste algoritmo são iterações preditoras; o que significa que todos os erros do Indicador de Tapia cometidos no algoritmo devem ser considerados.

Para o algoritmo *Intel* temos 9 erros num total de 9 iterações, ; o que acarreta em 2.5 % de erros na identificação.

Obs.: Note que neste algoritmo temos 13 iterações, e das 13, 4 delas eram iterações corretoras; o que significa que os erros do Indicador de Tapia cometidos nestas iterações devem ser desconsiderados, já que estas iterações não fazem uso da direção (u^a, v^a) .

Devemos observar que os erros são cometidos nas iterações iniciais do problema; o que significa que o Indicador de Tapia torna-se mais eficiente à medida que evoluem as iterações. Neste exemplo, em particular, o último erro foi cometido na iteração 5 para o algoritmo *Front* e na iteração 3 para o algoritmo *Intel*. A partir daí, o Indicador tornou-se muito eficiente.

Podemos dizer, então, que o Indicador de Tapia foi ineficiente apenas nos 50.0 % iniciais do desenvolvimento total do problema para o primeiro algoritmo e nos 23.1 % iniciais para o segundo algoritmo.

Essa mesma análise também pode ser feita para os demais algoritmos.

A Fig. 5 (*B.5* e *Ic.5*) mostra a posição dos erros para os algoritmos *Bord* e *Intelc*. Para o algoritmo *Bord* temos 11 iterações e 12 erros, o que acarreta em 2.8 % de erros. E a ineficiência do Indicador de Tapia se deu nos 90.9 % iniciais do problema. E para o algoritmo *Intelc* temos 9 iterações (total de 13 iterações, sendo 4 delas corretoras) e 9 erros, o que acarreta em 2.5 % de erros. E a ineficiência do Indicador se deu nos 23.1 % iniciais do problema.

A Fig. 5 (*L.5* e *Lc.5*) mostra a posição dos erros para os algoritmos *Large* e *Largec*. Para o algoritmo *Large* temos 9 iterações e 11 erros, o que acarreta em 3.1 % de erros. E a ineficiência do Indicador de Tapia se deu nos 33.3 % iniciais do problema. E para o algoritmo *Largec* temos 8 iterações (total de 12 iterações, sendo 4 delas corretoras) e 9 erros, o que acarreta em 2.8 % de erros. E a ineficiência do Indicador se deu nos 25.0 % iniciais do problema.

Abaixo apresentam-se duas tabelas estatísticas que mostram a confiabilidade do Indicador de Tapia para os algoritmos estudados em alguns problemas aleatórios .

A tabela IT.1 mostra a porcentagem de erros cometidos nos problemas em questão; e a

tabela IT.2 a porcentagem do problema onde o Indicador foi ineficiente. Resta lembrar que esta ineficiência ocorre sempre na parte inicial de cada problema.

As tabelas IT.1 e IT.2 têm, como linhas, os problemas aleatórios; e, como colunas, os algoritmos utilizados.

Info	Front	Intel	Bord	Intelc	Large	Largec
PROBL.1	4.0	2.5	2.8	2.5	3.1	2.8
PROBL.2	2.5	0.9	1.0	0.9	1.6	1.1
PROBL.3	3.8	2.2	2.2	2.2	3.1	2.5
PROBL.4	5.7	2.5	3.2	2.5	4.2	3.1
PROBL.5	6.3	2.6	3.2	2.6	4.7	3.6
Média	4.5	2.1	2.5	2.1	3.3	2.6

Tabela IT.1: Percentual de Erros

Info	Front	Intel	Bord	Intelc	Large	Largec
PROBL.1	50.0	23.1	90.9	23.1	33.3	25.0
PROBL.2	44.4	45.5	30.8	45.5	50.0	50.0
PROBL.3	70.0	50.0	86.7	50.0	55.6	50.0
PROBL.4	54.5	46.2	42.9	46.2	44.4	54.5
PROBL.5	80.0	38.5	93.3	38.5	44.4	41.7
Média	59.8	40.7	68.9	40.7	45.5	44.2

Tabela IT.2: Percentual de Iterações com Indicador Errado

Obs.: Os dados das tabelas são em %.

Pela tabela IT.1 podemos perceber que:

Na média, os algoritmos onde o Indicador de Tapia cometem maior percentual de erros são os algoritmos do primeiro grupo (que, no nosso caso, é somente o algoritmo *Front*) e depois

vêm os algoritmos do terceiro grupo (que são os algoritmos *Large* e *Largec*). O segundo grupo de algoritmos (que são *Intel*, *Bord* e *Intelec*) foi o grupo onde o Indicador cometeu o menor percentual de erros.

Vamos observar a mesma tabela, porém por um outro aspecto; pelo critério adotado para os *Experimentos Numéricos*, ou seja, se atende ou não às condições do Teorema 3.3.3.

Tanto para os algoritmos que pertencem ao segundo grupo, como para os que pertencem ao terceiro grupo, atender condições teóricas implica em Indicador de Tapia mais eficiente.

Uma análise isolada dos algoritmos também deve ser feita. Esta análise nos indica o algoritmo *Front* como o algoritmo que teve o maior percentual de erros no Indicador de Tapia, por outro lado temos os algoritmos *Intel* e *Intelec* com o menor percentual.

Pela tabela FT.2 podemos concluir que:

Na média, para os algoritmos do primeiro grupo, houve uma dificuldade maior para o Indicador de Tapia identificar as variáveis grandes e pequenas, ou seja, houve uma persistência nas iterações com erros, implicando em uma maior ineficiência do Indicador no desenvolvimento inicial do algoritmo. Depois vem o segundo grupo e por fim o terceiro grupo, onde o Indicador de Tapia se tornou mais eficiente.

Se observarmos agora a mesma tabela, porém com o critério adotado para os *Experimentos Numéricos*, temos a seguinte análise:

Tanto para os algoritmos do segundo grupo como para os do terceiro grupo, estar de acordo com os aspectos teóricos implica em percentual menor da ineficiência do Indicador de Tapia. É bom lembrar que essa ineficiência ocorre em geral no desenvolvimento inicial dos algoritmos.

Uma análise isolada dos algoritmos nos indica o algoritmo *Bord* como o algoritmo com o maior percentual de ineficiência do Indicador de Tapia, ou seja, persistência em iterações com o Indicador errado; e os algoritmos *Intel* e *Intelec* com o menor percentual.

5.2.2 Desempenho dos algoritmos

Os algoritmos, como já era previsto, se mostraram bastante eficientes para resolver os problemas testados.

Uma das formas de avaliar o desempenho de um algoritmo é saber o número de iterações para se atingir a solução de um dado problema. As iterações representam cálculos das direções (u, v) .

Se esse número é baixo, isso é considerado um bom resultado; já que o custo computacional para fazer tal processo é alto, pois envolve a resolução de Sistemas Lineares grandes.

Agora vamos analisar esses aspectos para os nossos problemas concretos.

A Fig. 2 (F.2 e I.2) mostra o comportamento dos algoritmos *Front* e *Intel*.

Temos 10 iterações para o algoritmo *Front* e todas elas são iterações preditoras, isto é, iterações onde houve a atualização do μ , da forma $\mu_+ = \frac{x_+^t s_+}{n}$. Por outro lado, para o algoritmo *Intel* temos 13 iterações, onde 4 delas não foram iterações preditoras, ou seja, foram iterações de correção.

Essa mesma análise também pode ser feita para os demais algoritmos.

A Fig. 2 (B.2 e Ic.2) mostra o comportamento para os algoritmos *Bord* e *Intelc*. Para o algoritmo *Bord* temos 11 iterações, e todas elas preditoras. E para o algoritmo *Intelc* temos 13 iterações, com 4 delas sendo corretoras.

A Fig. 2 (L.2 e Lc.2) mostra o mesmo, só que para os algoritmos *Large* e *Largec*. Para o algoritmo *Large* temos 9 iterações, onde todas são preditoras. E para o algoritmo *Largec* temos 12 iterações, onde 4 delas são corretoras.

Abaixo apresentam-se duas tabelas estatísticas que mostram dois aspectos diferentes para o número de iterações dos algoritmos estudados em alguns problemas aleatórios.

A tabela DA.1 mostra o número total de iterações para a resolução dos problemas em questão; e a tabela DA.2 o número de iterações do algoritmo onde foram realizados apenas passos preditores, ou seja, apenas onde houve a atualização do μ .

As tabelas DA.1 e DA.2 têm, como linhas, os problemas aleatórios; e, como colunas, os algoritmos utilizados.

Info	<i>Front</i>	<i>Intel</i>	<i>Bord</i>	<i>Intelc</i>	<i>Large</i>	<i>Largec</i>
<i>PROBL.1</i>	10	13	11	13	9	12
<i>PROBL.2</i>	9	11	13	11	8	10
<i>PROBL.3</i>	10	12	15	12	9	11
<i>PROBL.4</i>	11	13	14	13	9	11
<i>PROBL.5</i>	9	10	10	10	8	9
Média	9.8	11.8	12.6	11.8	8.6	10.6

Tabela DA.1: Total de Iterações

Info	<i>Front</i>	<i>Intel</i>	<i>Bord</i>	<i>Intelc</i>	<i>Large</i>	<i>Largec</i>
<i>PROBL.1</i>	10	9	11	9	9	8
<i>PROBL.2</i>	9	8	13	8	8	7
<i>PROBL.3</i>	10	9	15	9	9	8
<i>PROBL.4</i>	11	10	14	10	9	8
<i>PROBL.5</i>	9	8	10	8	8	7
Média	9.8	8.8	12.6	8.8	8.6	7.6

Tabela DA.2: Total de Iterações Predictoras

Pela tabela DA.1 podemos perceber que:

Na média, os algoritmos que realizam maior número de iterações são os algoritmos do segundo grupo (que são os algoritmos *Intel*, *Bord* e *Intelc*), e depois vêm os algoritmos do primeiro grupo (que, no nosso caso, é somente o algoritmo *Front*). O terceiro grupo de algoritmos (que são os algoritmos *Large* e *Largec*) foi o grupo que realizou o menor número de iterações.

Vamos observar a mesma tabela, porém por um outro aspecto; pelo critério adotado para os *Experimentos Numéricos*, ou seja, se atende ou não às condições do Teorema 3.3.3.

Um fato curioso ocorre neste tipo de análise:

Para os algoritmos que pertencem ao segundo grupo, atender condições teóricas implica em número menor de iterações; já para os algoritmos do terceiro grupo, ocorre o inverso.

Uma análise isolada dos algoritmos também deve ser feita. Esta análise nos indica o algoritmo *Bord* como o algoritmo que teve o maior número de iterações, por outro lado temos o algoritmo *Large* com o menor número.

Pela tabela DA.2 podemos concluir que:

Na média, os algoritmos do primeiro grupo (somente o algoritmo *Front*) foram os algoritmos que mais realizaram iterações do tipo predictoras, ou seja, iterações onde o parâmetro μ é atualizado. Depois vem o segundo grupo e por fim o terceiro grupo, onde os algoritmos realizaram menor número de iterações com atualização do μ .

Se observarmos agora a mesma tabela, porém com o critério adotado para os *Experimentos*

Numéricos, temos a seguinte análise:

Tanto para os algoritmos do segundo grupo como para os do terceiro grupo, estar de acordo com os aspectos teóricos implica em redução no número de iterações que exigem a atualização do μ .

Uma análise isolada dos algoritmos nos indica o algoritmo *Bord* como o algoritmo com o maior número de iterações desse tipo, ou seja, iterações predictoras; e os algoritmos *Intel* e *Intelc* com o menor número.

Obs.: Os problemas gerados pelo programa GENLP foram problemas pequenos e com baixo índice de complexidade. Com isso não se conseguiu gerar problemas com condições próximas à do pior caso, previstas pela teoria.

Por exemplo: O número de passos de Newton previsto pela teoria para realizar uma centralização é da ordem de n . Nos nossos casos particulares essas centralizações sempre foram conseguidas com um número bastante baixo.

Podemos constatar esse fato analisando as figuras *F.3*, *I.3*, *B.3*, e as demais *.3, onde esse número não passou de 4.

5.2.3 Propriedades do Teorema 3.3.3

Devemos lembrar as hipóteses do Teorema 3.3.3:

"Seja (x, s) primal-dual viável, $\mu > 0$ tal que $\delta_\infty(x, s, \mu) \leq \alpha$ e $dist(g, \mu) \leq \xi$, onde $\alpha = 0.25$, $\xi = 0.136$ e g são as variáveis grandes".

Podemos notar que na seção *Experimentos Numéricos* tivemos as hipóteses do Teorema relaxadas. Tomamos $\alpha = 0.9$, criando assim uma vizinhança bem maior da trajetória central; e $\xi = 0.5$, inibindo a ação dos passos corretores, e estimamos g usando o Indicador de Tapia.

Podemos notar que, pela natureza dos algoritmos, exceto para o algoritmo *Front*, os pontos gerados estão na vizinhança $\mathcal{N}_\infty = \{(x, s) \in F \mid \delta_\infty(x, s, \mu) \leq 0.9 \text{ para algum } \mu \leq \mu^0, \mu^0 > 0\}$, ou seja, a hipótese relaxada $\delta_\infty(x, s, \mu) \leq 0.9$ é sempre atendida.

Temos que analisar então, a partir de qual iteração a hipótese $dist(g, \mu) \leq 0.5$ é satisfeita. E, em particular para o algoritmo *Front*, a hipótese $\delta_\infty(x, s, \mu) \leq 0.9$.

Vamos analisar, então, o algoritmo *Front* isoladamente.

Da fusão das figuras *F.2* e *F.4* podemos notar que a partir da iteração 8 as hipóteses (relaxadas) do Teorema 3.3.3 são válidas, isto é, as variáveis grandes estão bem centradas.

Observando a figura *F.2*, podemos verificar que a partir dessa iteração realmente um passo de Newton de centralização produz uma grande redução da proximidade em relação ao ponto central correspondente, isto é, a redução significativa de $\delta(x^c, s^c, \mu)$. Verifica-se também pela figura *F.4* o decréscimo de $dist(g^c, \mu)$.

Finalmente, observando a figura *F.6*, onde está a comparação de todas as medidas utilizadas pelo algoritmo, podemos notar que no final do algoritmo, exatamente a partir da iteração 8, onde a proximidade é baixa, a medida de proximidade $\delta(g, p^c, \mu) = \left\| \frac{gp^c}{\mu} - e \right\|$ é uma boa aproximação para a distância $dist(g, \mu)$.

Essa mesma análise também pode ser feita para os demais algoritmos.

Da fusão das figuras *I.2* e *I.4* notamos que a partir da iteração 9 as hipóteses (relaxadas) do Teorema 3.3.3 são válidas para o algoritmo *Intel*, isto é, as variáveis grandes estão bem centradas.

Observando a figura *I.2*, verificamos que a partir dessa iteração o passo de Newton de centralização é muito eficiente, ou seja, a reduz significativamente $\delta(x^c, s^c, \mu)$. Verifica-se também pela figura *F.4* o decréscimo de $dist(g^c, \mu)$.

Finalmente, pela figura *I.6*, podemos notar que exatamente a partir da iteração 9, a medida de proximidade $\delta(g, p^c, \mu)$ é uma boa aproximação para a distância $dist(g, \mu)$.

A análise para os outros algoritmos é bem semelhante, lembrando apenas que a hipótese $\delta_\infty(x, s, \mu) \leq 0.9$ é sempre atendida. Então basta verificar a partir de qual iteração a hipótese $dist(g, \mu) \leq 0.5$ ocorre.

Todos estes fatos nos mostram que, mesmo relaxando as hipóteses do Teorema 3.3.3, podemos obter bons resultados.

Pela análise destes algoritmos podemos concluir que:

”O comportamento do algoritmo de centralização é fortemente influenciado pela precisão com que são conhecidas as variáveis grandes. Se estas estão próximas de seus valores centrais, então o número de passos de centralização necessário em cada iteração do algoritmo será pequeno”.

Bibliografia

- [1] J. F. Bonnans e C. C. Gonzaga, *Convergence of interior point algorithms for the monotone linear complementary problem*, Mathematics of Operations Research, 21 (1996), 1-25.
- [2] A. S. El-Bakry, R. A. Tapia, Y. Zhang, *A study of indicator for identifying zero variables in interior-point methods*, (1991).
- [3] C. C. Gonzaga, *Asymptotic complexity of the Newton centering step in path following interior point algorithms*, Manuscritos em preparação, Universidade Federal de Santa Catarina, (1997).
- [4] C. C. Gonzaga, *Complexity of predictor-corrector algorithms for LCP based on a large neighborhood of the central path*, pré-print.
- [5] C. C. Gonzaga, *On the complexity of linear programming*, Resenhas IME-USP Vol. 2, No. 2 (1995) 197-207.
- [6] C. C. Gonzaga, *Path following methods for linear programming*, SIAM Review 34 (1992) 167-227.
- [7] C. C. Gonzaga, *The largest step path following algorithm for monotone linear complementary problems*, Mathematic Programming 76 (1997) 309-332.
- [8] C. C. Gonzaga e J. F. Bonnans, *Fast convergence of the simplified largest step path following algorithm*, Mathematic Programming 76 (1996) 95-115.
- [9] C. C. Gonzaga e R. A. Tapia, *On the convergence of the Mizuno-Todd-Ye algorithm to the analytic center of the solution set*, SIAM Journal on Optimization, pré-print.

- [10] M. Kojima, N. Megiddo, T. Noma e A. Yoshise, *A unified approach to interior algorithms for linear complementary problems*, Lectures Notes in Computer Science, 538 (Springer Verlag, Berlin, 1991).
- [11] M.Kojima, S.Mizuno e A.Yoshise, *A primal-dual interior point algorithm for linear programming*, in N. Meggido, ed., *Progress in Mathematical Programming: Interior Point and Related Methods* (Springer Verlag, New York, 1989) 29-47
- [12] N. Megiddo, *Pathways to the optimal set in linear programming*, in: N. Megiddo, ed., *Progress in Mathematical Programming: Interior Point and Related Methods* (Springer Verlag, New York, 1989) 131-158.
- [13] S. Mizuno, *A new polynomial time method for a linear complementary problem*, *Mathematical Programming* 56 (1992) 31-43.
- [14] R.D.C. Monteiro e I. Adler, *Interior path following primal-dual algorithms: Part I: Linear programming*, *Mathematical Programming* 44 (1989) 27-41.
- [15] R. D. C. Monteiro e T. Tsuchiya, *Limiting of the derivates of certain trajetories associated with a monotone horizontal linear complementary problem*, Working Paper 92-28, Departament of Systems an Industrial Engineering, University of Arizona, AZ (1992).
- [16] C. Ross, T. Terlak e J.-Ph. Vial, *Theory and algorithms for linear optimization - An interior point approach*, Jonh Wiley and Sons Ltd, 1997.
- [17] R. A. Tapia, *On the role of slack variables in quasi-Newton methods for constrained optimization*, in L. C. W. Dixon and G. P. Szego, editors, *Numerical optimization of dynamic systems* (1980) 235-246.