Universidade Federal de Minas Gerais
Programa de Pós-Graduação em Engenharia Elétrica
Centro de Pesquisa e Desenvolvimento em Engenharia Elétrica

Métodos Poliedro-Elipsoidais para Problemas de Otimização Contínuos e Discretos Quasi-Convexos

Por
Augusto dos Santos Moura Júnior

Orientador : Ricardo Takahashi

Tese de Doutorado

Texto de Doutorado submetido à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais, como requisito parcial a obtenção de título de Doutor em Engenharia Elétrica.

10 de Janeiro de 2008

Agradecimentos

Palavras não costumam ser suficientes para demonstrar a gratidão que sinto por aqueles que me ajudaram a me tornar um ser humano melhor, por conseguinte, a todos os que me dedicaram parte do seu tempo de vida durante este processo de elaboração da Tese eu agradeço de coração. Em especial, deixo pública aqui minha homenagem e admiração aos mestres Takahashi, Rodney e MacGregor Smith e meu amor a Augusto, Hilda, Ângela, Rafael, Regina, Paulo e às mulheres da minha vida Paula e Sophia...

Devemos julgar um homem mais pelas suas perguntas que pelas suas respostas.

Voltaire.

Resumo

Esta Tese introduz uma nova família de métodos pertencentes a classe dos algoritmos de *Exclusão de Semi-Espaço* baseados no método *Elipsoidal* e aplicáveis a problemas contínuos, discretos ou mistos discretos e contínuos, escalares ou vetoriais associados à funções quasi-convexas. Esta nova família de métodos é aqui denominada de família de métodos *Poliedro-Elipsoidais*. Quando da aplicação dos métodos aqui propostos para a solução de problemas contínuos, a aceleração do processo de convergência se fundamenta na utilização de cones baseados nas condições de Kuhn-Tucker para eficiência, construídos com informações correntes ou com informações já previamente calculadas, para a contração da região de interesse de busca. Para estes problemas, as funções também poderão ser não necessariamente diferenciáveis. Quando da aplicação dos métodos aqui propostos para a solução de problemas discretos ou mistos contínuos e discretos, a garantia de convergência global se fundamenta na utilização de uma função de enumeração implícita ou explícita de pontos inteiros concomitantemente ao algoritmo *Elipsoidal*. Já a aceleração do processo de convergência para estes problemas discretos e mistos se concretiza na definição de um algoritmo de *Branch-and-Cut*. Testes computacionais para os casos escalares exibirão uma significativa melhoria dos parâmetros de convergência, tais como tempo de cálculo, taxa de redução do volume e número de acessos da função-objetivo, desempenho este com tendência de se tornar ainda melhor, comparativamente, à medida que o esforço de avaliação das informações do problema se tornar mais significativo. Esta nova família de métodos também mostrará ser aplicável, eficientemente, para encontrar pontos não-dominados em problemas vetoriais, bem como para determinar a factibilidade de um problema.

Abstract

This Thesis will introduce a new class of methods that belongs to the *Semi-Space Excluding* class and are based on *Ellipsoidal* algorithm. This new methods' class is suitable for solving continuous, integer or mixed integer variables problems with single or vector cost functions in which all functions can be quasi-convex non necessarily differentiable. This new methods' class is here called *Polyhedron-Ellipsoid* class. When this new methods' class is applied on continuous variables problems, the convergence speed-up is supported by using KTE cones for compressing the search region, assembled from current or already calculated information. When this new methods' class is applied on integer or mixed integer variables problems the global convergence is guaranteed by an implicit or explicit enumeration function together with the *Ellipsoid* algorithm. For these integer or mixed integer problems the convergence speed-up is supported by a *Branch-and-Cut* algorithm. Computational tests, for single-objective problems, will exhibit an significant improve in convergence performance parameters as calculation time, volume reduction tax and number of objective-function calls. The results point to enhanced ones whether the problem evaluation effort increases. The proposed methods' class will also prove to be useful for finding, efficiently, non-dominated points in vector-objective problems as well as for demonstrating that a problem is feasible or not.

Lista de Acrônimos

As principais abreviações e ou acronismos usados nesta Tese são listados a seguir. Algumas siglas consagradas na literatura internacional foram mantidas em Inglês.

BIP	: *Binary Integer Programming* ou Programação Binária ou Zero-Um.
CDF	: Cone de Direções Factibilizantes.
CNF	: Condições Necessárias para Infactibilidade.
CDO	: Cone de Direções Otimizantes.
EAIP	: Esforço de Avaliação das Informações do Problema .
ECAO	: Esforço de Cálculo do Algoritmo de Otimização.
ECT	: Esforço Computacional Total para a determinação da solução Ótima.
GDC	: *Geometric Derived Cuts* ou Cortes Derivados Geometricamente.
HISPE	: *Historical Intersection Search Polyhedron Ellipsoid Method* ou Método *Poliedro-Elipsoidal* com Poliedro de Busca com Histórico de Intersecções.
INF	: Condições necessárias e suficientes de infactibilidade .
IP	: *Integer Programming* ou Programação Discreta.
KKTO	: Condições de Karush-Kuhn-Tucker para otimalidade.
KTE	: Condições de Kuhn-Tucker para eficiência.
L.I.	: Indicação de Independência Linear para vetores.
LIC	: Limite Inferior de Convergência .
LGQ	: *Linear Quadratic Gaussian* ou Gaussiano Quadrático Linear.
LMI	: *Linear Matrix Inequalities* ou Desigualdades Matriciais Lineares.
LSC	: Limite Superior de Convergência.
MCE	: Método Cone-Elipsoidal.
MEDR	: Método Elipsoidal com Dimensão Reduzida.
MEEE	: Método Elipsoidal com Enumeração Explícita.
MINLP	: *Mixed Integer Nonlinear Programming* ou Programação Mista Discreta e Contínua Não-Linear;
MIP	: *Mixed Integer Programming* ou Programação Mista Discreta e Contínua.
MPEH	: Método *Poliedro-Elipsoidal* por Hiperesfera.
MPESC	: Método *Poliedro-Elipsoidal* por Seqüência de Cortes profundos e rasos.
NIP	: *Nonlinear Integer Programming* ou Programação Discreta Não-Linear.

POAC : Projeto de Otimização Assistido por Computador

QCND : Quasi-Convexo Não necessariamente Diferenciável.

VOP : *Vectorial Optimization Problem* ou Problema de Otimização Vetorial, ou Problema de Otimização Multi-objetivo.

Lista de Símbolos

Nesta Tese, vetores são indicados por letras latinas minúsculas em negrito. Escalares são representados por letras minúsculas gregas ou latinas em itálico com ou sem sub-índice. Matrizes são indicadas por letras latinas maiúsculas. A seguir são listados os principais símbolos usados neste trabalho. Símbolos específicos serão definidos in loco.

^{T}	:= transposição de vetores ou matrizes.
$f(\cdot)$	:= conjunto das funções-objetivo de um problema. Neste trabalho estas funções devem ser consideradas funções quasi-convexas não necessariamente diferenciável quando nenhuma outra definição estiver explícita.
$g(\cdot)$	:= conjunto das funções-restrição de um problema. Neste trabalho estas funções devem ser consideradas funções quasi-convexas não necessariamente diferenciável quando nenhuma outra definição estiver explícita.
$\mathbf{x}$	:= vetor correspondente ao conjunto das variáveis de decisão, incluindo as variáveis contínuas e discretas quando existirem.
Ω	:= espaço dos parâmetros factíveis de um problema. Tal espaço será sempre identificado, neste trabalho, com um subconjunto de $\mathbb{R}^n$.
Ω^*	:= conjunto de soluções eficientes, ou Pareto-ótimas, de um problema. O conjunto Ω^* é um subconjunto do espaço de parâmetros Ω.
$\mathbf{x}^*$	:= solução ótima contínua de um problema, onde $\mathbf{x}^* \in \Omega^*$.
$\mathbf{x}^{z*}$	:= solução ótima mista contínua e discreta de um problema, onde $f(\mathbf{x}^*) \leq f(\mathbf{x}^{z*})$ para problemas de minimização.
$\mathbf{x}_k$	:= k-ésimo elemento de uma seqüência de vetores $\mathbf{x}$.
d	:= número de variáveis contínuas de um problema.
z	:= número de variáveis discretas de um problema.
n	:= dimensão de um problema ($z + d = n$).
p	:= número de restrições de um problema, onde $g_i(\cdot) \mid i \leq p$.
m	:= número de objetivos de um problema, onde $f_i(\cdot) \mid i \leq m$.
$\text{dist}(\mathbf{x}, \overline{\Omega})$	: distância euclidiana entre o ponto $\mathbf{x}$ e o conjunto convexo compacto $\overline{\Omega}$.
$max()$	:= função máximo valor dentre um conjunto de argumentos.
$min()$	:= função mínimo valor dentre um conjunto de argumentos.
$rank(M)$	:= função posto da matriz M.
$col(M)$	:= coluna q de uma matriz M, tal que a matriz $M \in \mathbb{R}^{r \times q}$.

$dim(\Upsilon)$:= função dimensão de um conjunto Υ.

$\lfloor \mathbf{x} \rfloor$:= arredondamento para o menor inteiro próximo a $\mathbf{x}$.

$\lceil \mathbf{x} \rceil$:= arredondamento para o maior inteiro próximo a $\mathbf{x}$.

$\langle \mathbf{x} \rangle$:= arredondamento para o ponto inteiro mais próximo a $\mathbf{x}$;

$(\alpha : \beta)$:= conjunto dos números inteiros seqüenciais de α até β incrementados por 1;

$\mathbf{w}(\delta)$:= subconjunto composto por algumas coordenadas de $\mathbf{w}$, onde δ representa o conjunto dos índices dos elementos do vetor.

$Q(\pi, \upsilon)$:= sub-matriz de Q, onde π representa as linhas que serão mantidas e υ representa as colunas.

$\bar{\nabla} f(\mathbf{x})$:= um vetor pertencente ao conjunto dos subdiferenciais da função $f(\cdot)$ avaliado no ponto $\mathbf{x}$ e podendo se tratar de um subderivativo, gradiente ou subgradiente desta função.

$H^{\xi,\mathbf{x}_k}$:= um hiperplano ortogonal a ξ que contém $\mathbf{x}_k$, tal que $\xi^T(\mathbf{x} - \mathbf{x}_k) = 0$.

$H_+^{\xi,\mathbf{x}_k}$:= um semi-espaço no qual $H_+^{\xi,\mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \xi^T(\mathbf{x} - \mathbf{x}_k) \geq 0\}$.

$H_-^{\xi,\mathbf{x}_k}$:= um semi-espaço no qual $H_-^{\xi,\mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \xi^T(\mathbf{x} - \mathbf{x}_k) \leq 0\}$.

$conv(X)$:= *convex hull* ou casca convexa de um conjunto de pontos X.

γ := conjunto dos índices das variáveis que não estão submetidas a restrições de igualdade, também denominado de conjunto das variáveis livres.

E_k := elipsóide de centro no ponto $\mathbf{x}_k$ e matrix Hessiana inversa Q_k.

$vol(E_k)$:= função volume do elipsóide E_k.

$L_{f=\alpha}$:= curva de nível da função $f(\cdot)$, tal que $f(\cdot) = \alpha$.

$L_{f\leq\alpha}$:= conjunto das curvas de nível da função $f(\cdot)$, tal que $f(\cdot) \leq \alpha$.

$\mathbf{w} < \mathbf{y}$:= operador $<$ sobre $\mathbf{w}$ e $\mathbf{y} \in \mathbb{R}^n$ tal que $\mathbf{w}(i) < \mathbf{y}(i)\ \forall\ i = 1, \ldots, n$.

$\mathbf{w} > \mathbf{y}$:= operador $>$ sobre $\mathbf{w}$ e $\mathbf{y} \in \mathbb{R}^n$ tal que $\mathbf{w}(i) > \mathbf{y}(i)\ \forall\ i = 1, \ldots, n$.

$\mathbf{w} \leq \mathbf{y}$:= operador $\leq$ sobre $\mathbf{w}$ e $\mathbf{y} \in \mathbb{R}^n$ tal que $\mathbf{w}(i) \leq \mathbf{y}(i)\ \forall\ i = 1, \ldots, n$.

$\mathbf{w} \geq \mathbf{y}$:= operador $\geq$ sobre $\mathbf{w}$ e $\mathbf{y} \in \mathbb{R}^n$ tal que $\mathbf{w}(i) \geq \mathbf{y}(i)\ \forall\ i = 1, \ldots, n$.

$\mathbf{w} \neq \mathbf{y}$:= operador $\neq$ sobre $\mathbf{w}$ e $\mathbf{y} \in \mathbb{R}^n$ tal que $\mathbf{w}(i) \neq \mathbf{y}(i)$ para algum $i = 1, \ldots, n$.

$\mathbf{w} \prec \mathbf{y}$:= operador $\prec$ sobre $\mathbf{w}$ e $\mathbf{y} \in \mathbb{R}^n$ tal que $\mathbf{w}(i) \leq \mathbf{y}(i)\ \forall\ i = 1, \ldots, n$ e $\mathbf{w}(j) < \mathbf{y}(j)$ para algum $j = 1, \ldots, n$.

$\mathbf{w} \not\prec \mathbf{y}$:= negação de $\mathbf{w} \prec \mathbf{y}$.

$\mathbf{w} \succ \mathbf{y}$:= operador $\succ$ sobre $\mathbf{w}$ e $\mathbf{y} \in \mathbb{R}^n$ tal que $\mathbf{w}(i) \geq \mathbf{y}(i)\ \forall\ i = 1, \ldots, n$ e $\mathbf{w}(j) > \mathbf{y}(j)$ para algum $j = 1, \ldots, n$.

$\mathbf{w} \not\succ \mathbf{y}$:= negação de $\mathbf{w} \succ \mathbf{y}$.

$==$:= operador relacional de igualdade.

$Rt(n)$:= função $(n/(n+1))(n^2/(n^2-1))^{(n-1)/2}$.

$Rp(\alpha,\beta,n)$:= função $((\frac{1}{n+1})^{1/2}(\frac{n^2}{2(n^2-1)})^{n/2}\frac{(2-\alpha^2-\bar{\beta}^2+\rho/2)^{n/2}(\rho-2+\alpha^2+\bar{\beta}^2)^{1/2}}{\alpha+\bar{\beta}}$.

$F(\cdot)$:= matriz $[\ \bar{\nabla}f_1(\cdot)\quad \bar{\nabla}f_2(\cdot)\quad \ldots\quad \bar{\nabla}f_m(\cdot)\ ]$.

$G(\cdot)$:= matriz $[\ \bar{\nabla}g_1(\cdot)\quad \bar{\nabla}g_2(\cdot)\quad \ldots\quad \bar{\nabla}g_p(\cdot)\ ]$.

τ := parâmetro que controla o tamanho da busca por valores já calculados para o método $HISPE$.

Sumário

Agradecimentos **i**

Epígrafe **iii**

Resumo **v**

Abstract **vii**

Lista de Siglas **ix**

Lista de Símbolos **xi**

Lista de Figuras **xxiv**

Lista de Tabelas **xxvii**

Lista de Algoritmos **xxix**

1 Introdução **1**

1.1 Motivações deste Estudo 1

1.2 Conceitos Preliminares 2

1.2.1 Funções Quasi-Convexas 2

1.2.1.1 Definição 2

1.2.1.2 Características das Curvas de Nível 4

1.2.2 Esforço para Avaliação do Problema de Otimização versus Esforço para Cálculo do Algoritmo de Otimização 4

1.2.3 Projetos de Otimização Assistidos por Computador 8

1.3 Teses Propostas 12

1.4 Estrutura desta Tese 13

I Otimização com Variáveis Exclusivamente Contínuas 17

2 Métodos de Exclusão de Semi-Espaço e Elipsoidal 19
2.1 Visão Geral sobre Métodos de Exclusão de Semi-Espaço 19
2.2 Algoritmos de Plano de Corte 20
2.2.1 Algoritmo de Plano de Corte de Kelley 22
2.3 Método *Elipsoidal* 23
2.3.1 Algoritmo *Elipsoidal* 23
2.3.2 Algoritmo *Elipsoidal* com *Deep Cut* 25
2.3.3 Tratamento das Restrições 27
2.3.4 Fatorização de *Cholesky*, LDL^T e UDU^T 28
2.4 Conclusões do Capítulo 30

3 Métodos que Usam Simultaneamente Mais de Um Vetor Subdiferencial para Cálculo de E_{k+1} 33
3.1 Cortes Paralelos 33
3.1.1 Cálculo do Novo Elipsóide para Dois Vetores não Paralelos 36
3.2 Corte em Cunha 41
3.3 Algoritmo dos Dois Subgradientes Sucessivos 45
3.3.1 Proposição de uma Variante do Algoritmo dos Dois Subgradientes Sucessivos Quando Existem Duas Restrições Violadas 50
3.4 Método *Cone-Elipsoidal* 52
3.4.1 Formalização do *Cone de Direções Otimizantes* e do *Cone de Direções Factibilizantes* 53
3.4.2 Algoritmo 58
3.5 Conclusões do Capítulo 59

4 Infactibilidade Estrita, Pareto-Otimalidade e Condições Complementares 61
4.1 Introdução 61
4.2 Definição da Condição de Infactibilidade 63
4.3 Condições de Decisão 67
4.3.1 Cones e Condições Complementares 68
4.4 Conclusões do Capítulo 71

5 Fundamento dos Métodos Poliedro-Elipsoidais para Problemas Contínuos 73
5.1 Formalização do *Poliedro de Busca* 73
5.2 Formalização do *Poliedro de Busca com Histórico de Intersecções* 78

5.3 Conclusões do Capítulo 81

6 Família dos Métodos Poliedro-Elipsoidais para Problemas Contínuas 83

6.1 Métodos *Elipsoidais* com Poliedro de Busca 83

6.1.1 Método *Poliedro-Elipsoidal* por Hiperesfera: $MPEH$ 84

6.1.2 Método *Poliedro-Elipsoidal* por Seqüência de Cortes Profundos e Rasos: $MPESC$ 89

6.2 *Historical Intersection Search Polyhedron Ellipsoid Method* 94

6.3 Conclusões do Capítulo 97

7 Resultados Computacionais para Problemas com Variáveis Contínuas 99

7.1 Problemas Escalares com Variáveis Contínuas 101

7.1.1 Problema Quasi-Convexo Não Necessariamente Diferenciável com Parâmetros Aleatórios 101

7.1.1.1 Influência do Aumento do Número de Restrições no Desempenho dos Algoritmos *Poliedro-Elipsoidais* 110

7.1.1.2 Problema Irrestrito 116

7.1.2 Problema Convexo Não Diferenciável com Parâmetros Aleatórios . . 118

7.1.3 Problema Convexo Diferenciável com Parâmetros Aleatórios 125

7.1.3.1 Problema Quadrático com Restrições Quadráticas 125

7.1.3.2 Problema Quadrático com Restrições Lineares 129

7.1.4 Problemas com Variáveis Relaxadas para Processo Químico de Sintetização 134

7.1.4.1 Processo de Sintetização 1 com Variáveis Relaxadas . . . 134

7.1.4.2 Processo de Sintetização 2 com Variáveis Relaxadas . . . 135

7.2 Problemas Vetoriais com Variáveis Contínuas 137

7.2.1 Problema Quasi-Convexo Não Necessariamente Diferenciável com Parâmetros Aleatórios 137

7.2.1.1 Problema Quadrático Irrestrito 143

7.2.2 Controlador com Norma H_2/H_∞ 145

7.2.3 Problema de Factibilidade 148

7.3 Conclusões do Capítulo 150

II Otimização com Variáveis Discretas e Mistas Discretas e Contínuas 153

8 Métodos Baseados em Elipsóide Aplicados à Programação Discreta e à Programação Mista 155

8.1 Introdução à Programação Discreta e Mista Contínua e Discreta 155
8.2 Restrição da Aplicação Direta dos Métodos Baseados em Elipsóide a Problemas Inteiros e Mistos Inteiros e Contínuos 160
8.3 Princípios Básicos da Aplicação de Métodos Baseados em Elipsóide a Problemas Inteiros e Mistos Inteiros e Contínuos 161
8.4 Conclusões do Capítulo 167

9 Métodos Elipsoidal e Poliedro-Elipsoidal com Hiperplanos de Corte Aplicados a Problemas Discretos 169

9.1 Hiperplanos de Corte Aplicados a Problemas MIP e IP 169
9.1.1 Cortes Convexos e Derivados Geometricamente 170
9.1.2 Cortes Chvátal-Gomory 172
9.2 Variação do Método Elipsoidal com Hiperplanos de Corte: MEHC 175
9.3 Limitações da Aplicação Prática do *MEHC* 178
9.4 Conclusões do Capítulo 179

10 Métodos Elipsoidais e Poliedro-Elipsoidais com Enumeração Aplicados a Problemas Discretos e Mistos 183

10.1 Variação do Método *Elipsoidal* com Enumeração Explícita: *MEEE* 184
10.2 Variações do Método *Elipsoidal* com Enumeração por *Branch-and-Bound*: *MEEBB* 190
10.2.1 Introdução ao Método *Branch-and-Bound* 190
10.2.2 Enumeração por *Branch-and-Bound* 193
10.2.3 Algoritmos *MEDR*, *BB* e *MEEBB* 195
10.3 Variações do Método *Elipsoidal* com *Branch-and-Cut* 205
10.3.1 Introdução ao *Branch-and-Cut* 205
10.3.2 Fundamentos de *Branch-and-Cut* Aplicados ao Método *Elipsoidal* . 205
10.3.3 Método *Elipsoidal* com Enumeração por *Branch-and-Cut*: *MEEBC* 208
10.3.4 Método *Branch-and-Cut Elipsoidal*: *MBCE* 209
10.4 Conclusões do Capítulo 216

11 Resultados Computacionais para Problemas Discretos e Mistos 217

11.1 Testes de Convergência para o Algoritmo *MEHC* 219

11.2 Problema MNIP Escalar Quasi-Convexo 221
11.2.1 Problema Misto Contínuo e Discreto com Parâmetros Aleatórios . . 221
11.2.2 Problema Binário para Processo Químico de Sintetização 233
11.2.2.1 Processo de Sintetização 1 233
11.2.2.2 Processo de Sintetização 2 234
11.3 Problema MNIP Escalar Convexo 236
11.4 Avaliação da Utilização de Informações Históricas no Algoritmo $MBCE$. 243
11.5 Conclusões do Capítulo 246

III Conclusões Finais 249

12 Conclusões e Trabalhos Futuros 251
12.1 Conclusões 251
12.2 Questões a Serem Respondidas por Trabalhos Futuros 253

Apêndice A 263

Lista de Figuras

1.1 Exemplo de uma função quasi-convexa. 3
1.2 Classificação dos problemas dada a dimensão e o esforço computacional associado à avaliação das funções, que os modelam, para um único ponto $\mathbf{x}_k$. 5
1.3 Esforço total para a solução de um problema de otimização. 8
1.4 Diagrama da integração off-line de um PAOC. 12
1.5 Diagrama da integração on-line de um PAOC. 12

2.1 Ilustração de uma seqüência de convergência para um algoritmo de *Planos de Corte*. 22
2.2 Ilustração de uma iteração do método *Elipsoidal* e interpretação geométrica dos seus parâmetros. 25
2.3 Ilustração de duas iterações do método *Elipsoidal* com corte deslocado do centro $\mathbf{x}_k$. 27
2.4 Comparação dos métodos de fatorização para aumento da estabilidade de algoritmo *Elipsoidal*. 31

3.1 Taxa de redução do volume do método *Elipsoidal* em função do aumento da dimensão do problema. 34
3.2 Ilustração de uma iteração do método *Elipsoidal* com *Cortes Paralelos*. . . 36
3.3 Variação do volume gerado por *Cortes Paralelos* em função do $\angle(\bar{\nabla}_k, \bar{\nabla}_\omega)$ e da dimensão n. 38
3.4 Ilustração de duas iterações do método *Elipsoidal* com *Cortes Paralelos*. . . 40
3.5 Variação do volume gerado por *Cortes em Cunha* em função do $\angle(\bar{\nabla}_k, \bar{\nabla}_\omega)$ e da dimensão n. 43
3.6 Ilustração de duas iterações do método *Elipsoidal* com *Cortes em Cunha*. . 44
3.7 Ilustração de duas iterações do método *Dois Subgradientes Sucessivos*. . . . 49
3.8 Ilustração de uma iteração da variante proposta para o método *Dois Subgradientes Sucessivos*. 53

3.9 Ilustração da existência de um *Cone CDO* em um ponto $\mathbf{x}$ no interior da região factível. 55
3.10 Ilustração da existência de um *Cone CDO* em um ponto $\mathbf{x}$ na borda da região factível. 56
3.11 Ilustração da existência de dois *Cones CDFs* associados aos pontos $\mathbf{x}_a$ e $\mathbf{x}_b$ exteriores à região factível. 57
3.12 Ilustração de um problema vetorial com condições iniciais que induzem o algoritmo *Cone-Elipsoidal* a falha. 59

4.1 Ilustração de um ponto $\mathbf{x}^*$ que satisfaz KTE. 65
4.2 Ilustração de um ponto $\mathbf{x}_k$ que satisfaz INF. 66
4.3 Ilustração de um ponto $\mathbf{x}_k$, no qual KTE e INF não são satisfeitas. 66

5.1 Limitação da construção de um método baseado em *Elipsóide* pela utilização direta de $\mathcal{C}^O(\mathbf{x}_k)$ e $\mathcal{C}^O(\mathbf{x}_{k+1})$. 75
5.2 Ilustração de um poliedro de busca para $\mathbf{w}$ e $\mathbf{v}$ factíveis. 77
5.3 Ilustração de um poliedro de busca resultante da intersecção $\mathcal{C}^O(\mathbf{x}_1) \cap \mathcal{C}^O(\mathbf{x}_2) \cap \mathcal{C}^F(\mathbf{x}_3) \cap \Gamma$ para o caso mono-objetivo. 80

6.1 Ilustração de duas iterações de cálculo do método *Poliedro-Elipsoidal* por Hiperesfera. 88
6.2 Ilustração do cálculo de E_{k+1} pelo algoritmo Seqüência de Cortes Profundos e Rasos. 93

7.1 Resultados percentuais para o número de acessos à $f(\mathbf{x})$ dos algoritmos *Kim*, *MPESC* e *HISPE*, em referência a *Shor*, para problemas PCEQC. 106
7.2 Resultados percentuais para o tempo total para convergência dos algoritmos *Kim*, *MPESC* e *HISPE*, em referência a *Shor*, para problemas PCEQC. 107
7.3 Desempenho do algoritmo HISPE mediante a variação do tamanho da busca histórica para o problema PCEQC de dimensões 5, 15, 25 e 35. . . . 109
7.4 Resultados comparativos para problemas PCEQC em função do aumento do número de restrições. 115
7.5 Resultados percentuais para o número de acessos à $f(\mathbf{x})$ dos algoritmos *Kim*, *MPESC* e *HISPE*, em referência a *Shor*, para problemas PCECN. 121
7.6 Resultados percentuais para o tempo total para convergência dos algoritmos *Kim*, *MPESC* e *HISPE*, em referência a *Shor*, para problemas PCECN. 122

7.7 Desempenho do algoritmo HISPE mediante a variação do tamanho da busca histórica para o problema PCECN de dimensões 5, 15, 25 e 35. . . . 124
7.8 Resultados percentuais para o número de acessos à $f(\mathbf{x})$ dos algoritmos *Kim*, *MPESC* e *HISPE*, em referência a *Shor*, para problemas PCEQQ. 127
7.9 Resultados percentuais para o tempo total para convergência dos algoritmos *Kim*, *MPESC* e *HISPE*, em referência a *Shor*, para problemas PCEQQ. 128
7.10 Desempenho do algoritmo HISPE mediante a variação do tamanho da busca histórica para o problema PEQQ de dimensão 35. 128
7.11 Resultados percentuais para o número de acessos à $f(\mathbf{x})$ dos algoritmos *Kim*, *MPESC* e *HISPE*, em referência a *Shor*, para problemas PCEQL. 131
7.12 Resultados percentuais para o tempo total para convergência dos algoritmos *Kim*, *MPESC* e *HISPE*, em referência a *Shor*, para problemas PCEQL. 132
7.13 Desempenho do algoritmo HISPE mediante a variação do tamanho da busca histórica para o problema PEQL de dimensão 35. 133
7.14 Resultados para o número de acessos à $f(\mathbf{x})$ dos algoritmos *MPESC* e *HISPE* para problemas PCVQC. 140
7.15 Resultados para o tempo total para convergência dos algoritmos *MPESC* e *HISPE* para problemas PCVQC. 141
7.16 Conjunto dos pontos avaliados durante a convergência dos métodos *MPESC* e *HISPE* para um problema PCVQC, $m = 2$. 142
7.17 Resultados para o número de acessos à $f(\mathbf{x})$ dos algoritmos *MPESC* e *HISPE* para problemas PVEQI. 144
7.18 Resultados para o tempo total para convergência dos algoritmos *MPESC* e *HISPE* para problemas PVEQI. 145
7.19 conjunto *Pareto-Ótimo* das soluções obtidas pelos algoritmos *LMI* e *HISPE* para o problema H_2/H_∞. 147

8.1 Exemplo comparativo da solução discreta de um problema $\mathbf{x}^{z*}$ e da solução obtida pelo arredondamento da solução contínua $\mathbf{x}^*$. 158
8.2 Falha do método *Elipsoidal* quando aplicado a problemas MINP e NIP. . . 164
8.3 Fluxograma geral de um algoritmo *Elipsoidal* alterado para a inclusão de novas restrições baseadas em planos de cortes. 165
8.4 Fluxograma geral de um algoritmo *Elipsoidal* alterado para enumerar um ponto discreto, ou misto discreto e contínuo. 166

9.1 Corte derivado geometricamente baseado em uma hiperesfera. 171
9.2 Corte derivado geometricamente baseado em um hiperdiamante. 172
9.3 Cortes pela visão de Chvátal-Gomory. 175
9.4 Seqüência de criação de uma nova restrição por GDC para um algoritmo baseado em *Elipsóide*. 180

10.1 Ilustração de uma seqüência de convergência do algoritmo $MEEE$. 189
10.2 Representação em árvore para um problema binário. 192
10.3 Fluxograma representativo do algoritmo *Branch-and-Bound* proposto por Land e Doig. 198
10.4 Fluxograma representativo do algoritmo *Branch-and-Bound* proposto por Dankin. 199
10.5 Fluxograma do algoritmo de enumeração por *Branch-and-Bound* para a proposição de Dakin. 200
10.6 Fluxograma geral para um método *Elipsoidal* com enumeração por *Branch-and-Bound*. 201
10.7 Fluxograma de um método *Branch-and-Cut* sobre a codificação de Dakin com solução do problema obtido por relaxação via *Elipsóide*. 211
10.8 Fluxograma representativo do algoritmo *Branch-and-Cut Elipsoidal*. 214

11.1 Ilustração de uma seqüência de convergência do algoritmo $MEHC$. 220
11.2 Resultados percentuais de Tempo, Acessos $f(\mathbf{x})$ e Cálculos Q_{k+1}: $MBCE^{LD}$ x BB^{LD} x $MBCE^{Da}$ x BB^{DL} para problemas PMEQC. 228
11.3 Resultados dos problemas PMEQC para BB^{Da} e $MBCE^{Da}$: número de variáveis inteiras z x dimensão n. 230
11.4 Resultados dos problemas PMEQC para BB^{LD} e $MBCE^{LD}$: número de variáveis inteiras z x dimensão n. 231
11.5 Contração do volume do elipsóide para os algoritmos $MBCE^{Da}$ e $MBCE^{LD}$, dimensão $n = 4$ e $z = 3$. 232
11.6 Resultados percentuais de Tempo, Acessos $f(\mathbf{x})$ e Cálculos Q_{k+1}: $MBCE^{LD}$ x BB^{LD} x $MBCE^{Da}$ x BB^{DL} para problemas PMEC. 240
11.7 Resultados dos problemas PMEC para BB^{Da} e $MBCE^{Da}$: número de variáveis inteiras z x dimensão n. 241
11.8 Resultados dos problemas PMEC para BB^{LD} e $MBCE^{LD}$: número de variáveis inteiras z x dimensão n. 242

Lista de Tabelas

7.1 Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 5. 102

7.2 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 15. 102

7.3 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 25. 103

7.4 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 35. 103

7.5 Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 5, em função do aumento do número de restrições. 112

7.6 Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 10, em função do aumento do número de restrições. 113

7.7 Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 15, em função do aumento do número de restrições. 114

7.8 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC irrestrito de dimensão 5. 116

7.9 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC irrestrito de dimensão 15. 116

7.10 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC irrestrito de dimensão 25. 117

7.11 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC irrestrito de dimensão 35. 117

7.12 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQN de dimensão 5. 119

7.13 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQN de dimensão 15. 119
7.14 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQN de dimensão 25. 120
7.15 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQN de dimensão 35. 120
7.16 Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQQ de dimensão 15. 126
7.17 Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQQ de dimensão 35. 126
7.18 Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQL de dimensão 15. 129
7.19 Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQL de dimensão 35. 130
7.20 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* do problema de sintetização 1. 135
7.21 Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* do problema de sintetização 2. 136
7.22 Resultados comparativos dos algoritmos *MPESC* e *HISPE* para um problema PCVQC de dimensão 5. 138
7.23 Resultados comparativos dos algoritmos *MPESC* e *HISPE* para um problema PCVQC de dimensão 15. 138
7.24 Resultados comparativos dos algoritmos *MPESC* e *HISPE* para um problema PCVQC de dimensão 25. 138
7.25 Resultados comparativos dos algoritmos *MPESC* e *HISPE* para um problema PCVQI de dimensão 5. 143
7.26 Resultados comparativos dos algoritmos *MPESC* e *HISPE* para um problema PCVQI de dimensão 15. 143
7.27 Resultados comparativos dos algoritmos *MPESC* e *HISPE* para um problema PCVQI de dimensão 25. 144
7.28 Resultados comparativos dos algoritmos *MPESC* e *HISPE* para um problema H_2/H_∞ de dimensão 8. 148
7.29 Resultados dos problemas PCVF paras as dimensões n=[5:10] obtidos pelo algoritmo *HISPE*. 149

11.1 Resultados comparativos dos métodos *MEEE*, *MEEBB*, *MEEBC*, *MBCE* e *BB* para um problema PMEQC, dimensão 3. 222

11.2 Resultados comparativos dos métodos $MEEE$, $MEEBB$, $MEEBC$, $MBCE$ e BB para um problema PMEQC, dimensão 4. 223
11.3 Resultados comparativos dos métodos $MEEE$, $MEEBB$, $MEEBC$, $MBCE$ e BB para um problema PMEQC, dimensão 5. 224
11.4 Resultados comparativos dos métodos $MBCE$ e BB para um problema PMEQC, dimensão 8. 226
11.5 Resultados comparativos dos métodos $MBCE$ e BB para um problema PMEQC, dimensão 9. 227
11.6 Resultados do problema de sintetizacao 1. 234
11.7 Resultados do problema de sintetizacao 2. 236
11.8 Resultados comparativos dos métodos $MBCE$ e BB para um problema PMEC, dimensão 5. 237
11.9 Resultados comparativos dos métodos $MBCE$ e BB para um problema PMEC, dimensão 7. 238
11.10 Resultados comparativos dos métodos $MBCE$ e BB para um problema PMEC, dimensão 9. 239
11.11 Resultados comparativos dos métodos $MBCE_{HISPE}$, $MBCE$ e BB para um problema PMEQC, dimensão 8. 244
11.12 Resultados comparativos dos métodos $MBCE_{HISPE}$, $MBCE$ e BB para um problema PMEQC, dimensão 9. 245

Lista de Algoritmos

1 Geral para os Métodos de *Exclusão de Semi-espaço* 20
2 Geral para os Métodos de *Planos de Corte* 21
3 *Elipsoidal* proposto por Shor 26
4 *Dois Subgradientes Sucessivos* 46
5 *Dois Subgradientes Sucessivos 2.R.A.* 50
6 *Cone-Elipsoidal* 58
7 *Poliedro-Elipsoidal* por Hiperesfera 86
8 Seqüência de Cortes Profundos e Rasos 91
9 *Poliedro-Elipsoidal* por Seqüência de Cortes Profundos e Rasos 92
10 *Historical Intersection Search Polyhedron Ellipsoid* 96
11 *Elipsoidal* com Hiperplanos de Corte 181
12 *Elipsoidal* com Enumeração Explícita 188
13 *Elipsoidal* com Dimensão Reduzida 202
14 *Branch-and-Bound* sobre proposição de Dakin 203
15 *Elipsoidal* com Enumeração por Branch-and-Bound 204
16 *Branch-and-Cut* sobre proposição de Dakin 212
17 *Elipsoidal* com Enumeração por Branch-and-Cut 213
18 *Branch-and-Cut Elipsoidal* 215
19 Fatorização UDU^T 263

Capítulo 1

Introdução

Esta Tese tem por finalidade a construção e a avaliação de novos algoritmos pertencentes à classe de *exclusão de semi-espaços* que corroborem a proposição: *as informações calculadas ao longo da convergência de um algoritmo, podem, e devem, ser reutilizadas para aumentar o desempenho da convergência destes, a despeito do acréscimo de esforço computacional necessário para identificá-las.* Desta feita, um novo algoritmo pertencente à família dos algoritmos *Cone-Elipsoidais* é proposto, bem como uma nova família de algoritmos denominada *Cone-Elipsoidais Históricos* é apresentada, tanto para a aplicação em problemas mono-objetivos como para problemas vetoriais.

Para situar o leitor quanto à relevância do tema, este capítulo apresentará inicialmente a discussão do problema em questão. Em seguida será apresentada uma descrição das teses propostas para serem estudadas no decorrer deste trabalho. Por fim, será apresentada a estruturação dos capítulos, bem como um resumo inicial de cada um deles.

Maiores detalhes sobre os temas tratados nas seções apresentadas neste capítulo podem ser conseguidos nas referências (Rockafellar n.d., Heing 1982, Luenberger 1984, Nemhauser & Wolsey n.d., Floudas n.d., Takahashi 2003).

1.1 Motivações deste Estudo

O desenvolvimento nas últimas décadas das diversas linhas de pesquisa incluídas na área de conhecimento definida pela *Teoria de Otimização* foi fortemente influenciado pelo aparecimento, evolução e popularização dos computadores digitais (Luenberger n.d.). Por questões históricas, muitos dos problemas clássicos da otimização inicialmente surgiram a partir de problemas da física (Luenberger n.d.) definidos por grandes matemáticos, tais como Gauss, Lagrange, Euler e Bernouli. Hoje o desenvolvimento e aplicação da *Teoria de Otimização* se estende por inúmeras áreas de conhecimento, tais como En-

genharia, (Wolsey n.d., Parker. & Rardin 1988), Administração (Nemhauser & Wolsey n.d., Smith n.d.), Bioquímica (Floudas n.d., Duran & Grossmann 1986), Economia (Li & Sun n.d., Moura & Leal 2005), etc.

Mesmo com esta evolução da *Teoria de Otimização*, construir um método de otimização que possa ser usado de forma robusta e eficiente em todos os tipos de problemas existentes, é um desejo que, embora antigo, ainda encontra-se sem qualquer perspectiva de tornar-se realidade (Ho & Pepyne 2002). Embora esta afirmação possa ser vista por alguns como um fator desestimulante para a pesquisa e o desenvolvimento de novas teorias, e suas respectivas implementações através dos algoritmos, para outros tantos ela é o desafio que os moverá na busca e construção do novo. Assim sendo, diferentes idéias e heurísticas têm sido utilizadas com sucesso para a resolução de problemas ou classes de problemas específicos ao longo do desenvolvimento da história da otimização.

Deve-se ressaltar que, não obstante à existência de um número considerável de problemas que podem ser classificados como quasi-convexos (Roberts & Varberg n.d., Floudas n.d.), há apenas uns poucos algoritmos capazes de resolver estes problemas com garantia de convergência global (Westerlund & Pörn 2002), principalmente para os problemas com variáveis discretas. Por conseguinte, o desenvolvimento de algoritmos com a capacidade de solucionar problemas quasi-convexos, não necessariamente diferenciáveis, constituiu o ponto chave da motivação do desenvolvimento desta Tese.

1.2 Conceitos Preliminares

1.2.1 Funções Quasi-Convexas

Por este trabalho tratar de métodos de otimização a serem aplicados a problemas cujas funções-objetivo e restrições são pertencentes à classe de funções quasi-convexas, as seguintes definições e características tornam-se essenciais para o bom entendimento das proposições aqui apresentadas.

1.2.1.1 Definição

A classe das funções denominadas por quasi-convexas pode ser definida pelas seguintes descrições (Soleimani-damaneh 2007, Floudas n.d., Roberts & Varberg n.d.):

i) Seja S um conjunto convexo não vazio definido no $\mathbb{R}^n$. A função $f(\mathbf{x})$ será quasi-convexa se

$$f([(1-\lambda)\mathbf{w} + \lambda\mathbf{v}) \leq max([f(\mathbf{w})\ f(\mathbf{v})]) \tag{1.1}$$

$$\forall\ \lambda \in [0,1] \text{ and } \forall\ \mathbf{w},\ \mathbf{v} \in S$$

ii) Dada uma função $f(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}$, considere J como o menor intervalo (aberto ou fechado, finito ou infinito) que contenha todos os valores de $f(\cdot)$. Defina o conjunto subnível $L_{f\leq\alpha}$ como:

$$L_{f\leq\alpha} = \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) \leq \alpha\} \tag{1.2}$$

Assim,

$$f(\cdot) \textit{ é quasi-convexa se } L_{f\leq\alpha} \textit{ é convexa para cada } \alpha \in J$$

A figura 1.1 mostra um exemplo de uma função de custo quasi-convexa para um problema quasi-convexo não necessariamente diferenciável (QCND). Como pode ser observado, todas as curvas de nível são convexas.

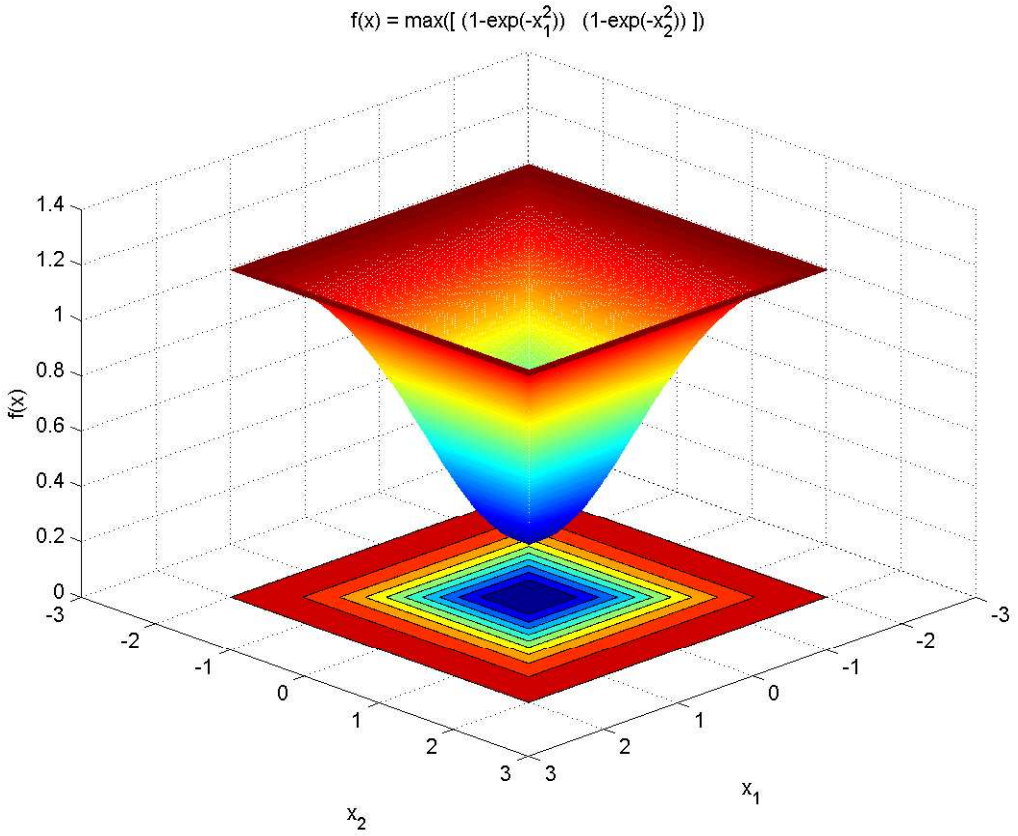


Figura 1.1: Exemplo de uma função quasi-convexa.

1.2.1.2 Características das Curvas de Nível

As curvas de nível são definidas por:

$$L_{f=\alpha} = \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) = \alpha\} \tag{1.3}$$

Dado um vetor ξ e um ponto $\mathbf{x}_k$ o hiperplano ortogonal a ξ que contém $\mathbf{x}_k$ é definido como $H^{\xi,\mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \xi^T(\mathbf{x} - \mathbf{x}_k) = 0\}$. Analogamente, os dois semi-espaços são definidos como $H_-^{\xi,\mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \xi^T(\mathbf{x} - \mathbf{x}_k) \leq 0\}$ e $H_+^{\xi,\mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \xi^T(\mathbf{x} - \mathbf{x}_k) \geq 0\}$.

Uma vez que qualquer conjunto subnível de uma função quasi-convexa é um conjunto convexo, qualquer vetor sub-derivativo η avaliado no ponto $\mathbf{x}_k$ garante que

$$\mathbf{x}^L \in H_-^{\eta,\mathbf{x}_k} \; \forall \, \mathbf{x}^L \in L_{f=f(\mathbf{x}_k)} \tag{1.4}$$

dado que $f(\mathbf{x}^L) - f(\mathbf{x}_k) \geq \eta^T(\mathbf{x}^L - \mathbf{x}_k)$.

1.2.2 Esforço para Avaliação do Problema de Otimização versus Esforço para Cálculo do Algoritmo de Otimização

Para balizar a discussão sobre o esforço computacional necessário para solucionar um problema de otimização, inicialmente deve-se restringir a discussão ao esforço computacional associado exclusivamente aos problemas. Para tanto, a figura 1.2 exibe um mapeamento função da dimensão de um problema de otimização e do esforço necessário para avaliar em um único ponto as funções que modelam este problema. Este mapeamento é apresentado, com um propósito puramente didático, para agrupar nas sete regiões indicadas grandes classes de problemas de otimização. Estas regiões foram escolhidas para que seja possível discutir intuitivamente a aplicabilidade de uma família de métodos com respeito à dimensão dos problemas em relação às especificidades destes problemas. Estas características específicas dos problemas, tais como convexidade, diferenciabilidade e existência de equações analítica, determinam uma variação no esforço computacional necessário para a avaliação das funções que modelam os problemas.

Assim, propõe-se classificar as regiões indicadas como sendo:

*R*1 **:** Região que agrupa os problemas considerados de pequena dimensão e baixo esforço computacional para a avaliação das funções matemáticas que expressam os problemas. Como referência, estes são problemas com dezenas de variáveis e com tempo de avaliação de funções menor que um segundo. Atualmente, existem algumas pesquisas relacionadas com a otimização de problemas situados nesta região,

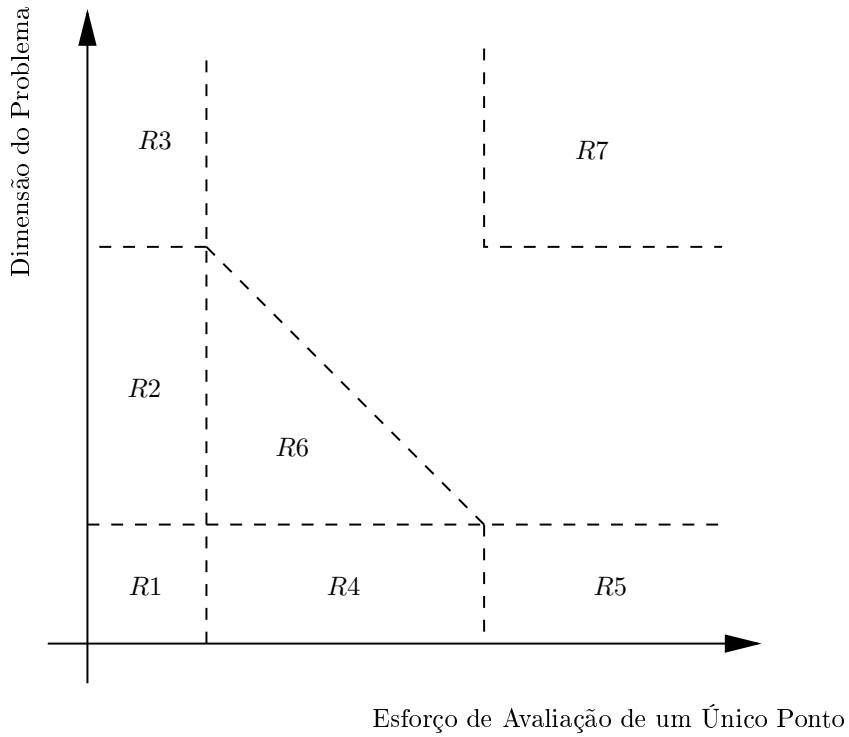


Figura 1.2: Classificação dos problemas dada a dimensão e o esforço computacional associado à avaliação das funções, que os modelam, para um único ponto $\mathbf{x}_k$.

nos quais as funções apresentam características especiais. Destacam-se as funções com características multimodais, com possibilidade de múltiplas escalas, não diferenciáveis e descontínuas. Alguns dos novos algoritmos que continuam a explorar problemas situados nesta região pertencem à classe dos algoritmos evolutivos (Holland 1975, Goldberg 1989). Existem ainda algoritmos determinísticos, da classe de otimização global, que têm sido construídos para lidar com problemas desta região, em particular alguns algoritmos do tipo *Branch-and-Bound* (Land & Doig 1960, Dakin 1965). Para a classe de problemas não-convexos com variáveis discretas, ou mistas contínuas e discretas, alguns métodos se baseiam ainda em aproximações convexas sucessivas (Tawarmalani & Sahinidis 2001).

$R2$ **:** Região que agrupa os problemas de tamanho médio e baixo esforço computacional para a avaliação das funções. Como referência, estes são problemas com centenas ou alguns milhares de variáveis. Os problemas tratados nesta classe incluem os unimodais diferenciáveis, solucionáveis por métodos *Quasi-Newton* (Luenberger 1984), *Programação Quadrática Seqüencial* (Boggs 1995) e *Trust Region* (Conn, Gould & Toint 1987), e os convexos, solucionáveis pelos métodos de *Programação Convexa* (Boyd & Vandenberghe 2004). Para qualquer dos casos, as dificuldades existentes

estão relacionadas com lidar com estabilidade numérica e matrizes esparsas.

$R3$ **:** Região que agrupa os problemas de tamanho elevado e baixo esforço computacional para a avaliação das funções. Atualmente estes problemas atingem a ordem de grandeza de 10^6 variáveis. Esta classe de problemas é constituída basicamente por problemas quadráticos convexos e problemas lineares. Dentre as ferramentas de otimização para o tratamento dos problemas desta região destacam-se os algoritmos de *Pontos Interiores* (Karmarkar 1984) e variações do algoritmo *Simplex* (Dantzig 2002).

$R4$ **:** Região que agrupa os problemas de pequena dimensão e esforço computacional mediano para a avaliação das funções. Como referência estes são problemas com tempo de avaliação de funções menor que uma hora por avaliação. Os mesmos métodos mencionados para a região $R2$ são aplicados nesta região, observando-se a ressalva de que é atribuída menor importância aos quesitos de estabilidade numérica e armazenamento de informações, uma vez que o foco das preocupações passa a ser a aplicação de operações exatas que conduzirão à obtenção da solução com um número menor de iterações. Para alguns problemas desta classe os algoritmos evolutivos podem ser aplicados, dependendo da natureza da função-objetivo.

$R5$ **:** Região que agrupa os problemas de esforço computacional elevado para a avaliação das funções. Como referência estes são problemas com tempo de avaliação de funções da ordem de dias por avaliação. Esta classe é essencialmente constituída por problemas de projetos de engenharia que utilizam modelos matemáticos computacionais, nos quais a representação de um dispositivo a ser otimizado implica na resolução de equações diferenciais parciais ou na utilização intensiva da simulação de *Monte Carlo* (Liu 2001). Os mesmos métodos discutidos para $R4$ com o mesmo foco na velocidade de convergência são aplicados nesta região. Existem também alguns desenvolvimentos particularmente associados a problemas específicos ou métodos híbridos, no intuito de utilizar toda e qualquer informação disponível para produzir a aceleração da convergência. Nesta região encontram-se os problemas tipo *Black Box* (Jones, Schonlau & Welch 1998).

$R6$ **:** Região que agrupa os problemas de dimensão mediana e esforço computacional mediano para a avaliação das funções. O mesmo contexto e os mesmos algoritmos citados para a região $R5$ são aplicados neste caso, com a preocupação adicional dos quesitos de estabilidade numérica e armazenamento de informações, tal como na região $R2$. Como conseqüência disto, os problemas de maior dimensão desta região podem ser

solucionados apenas quando associados a pequenos esforços computacionais para a avaliação das funções, assim como os problemas de esforços computacionais mais elevados para a avaliação das funções podem ser solucionados apenas quando associados a pequenas dimensões.

*R*7 **:** Região que agrupa os problemas de dimensão elevada e esforço computacional elevado para a avaliação das funções. Atualmente, não existem algoritmos capazes de lidar com os problemas desta região.

Para se realizar a análise do esforço computacional referente à solução de um problema de otimização faz-se necessário o conceito de esforço computacional total (ECT) para a solução de um problema. Como ECT, considera-se a soma de dois diferentes esforços:

EAIP: O *Esforço de Avaliação das Informações do Problema* representa todo custo computacional relacionado às informações que caracterizam e definem o problema, demandadas pelo algoritmo no processo de convergência, tal como avaliação das funções de custo, restrições, gradientes, hessianas, etc;

ECAO: O *Esforço de Cálculo do Algoritmo de Otimização* representa o esforço computacional relacionado à execução do algoritmo de otimização. Assim, ele engloba todo o esforço computacional relacionado à busca, classificação, manipulação e operações de cálculo que caracterizam o processo utilizado pelo algoritmo de otimização para encontrar a solução ótima.

Para proposição de um novo método de otimização, qualquer esforço computacional decorrente da teoria que fundamenta o método deve ser sustentado pelo resultado que esta nova abordagem de solução do problema poderá produzir na redução do esforço computacional total para determinação da solução ótima (ECT = EAIP + ECAO).

Conseqüentemente, o fato de um algoritmo de otimização consumir menor esforço computacional para completar uma iteração, comparado a um segundo algoritmo, não significa, necessariamente, que este primeiro algoritmo apresentará menor tempo de convergência total. A figura 1.3 apresenta uma curva, didática, onde dois pontos distintos são identificados para exibir graficamente este raciocínio. Uma vez que a solução mínima de uma função soma, tal como definido para o esforço total, é definido por um vetor de dimensão n com todos os valores correspondentes a -1, fica claro que o ponto de mínimo ECT não coincide com o ponto que representa o menor esforço ECAO. Este raciocínio também permite pressupor que algoritmos que demandem maior EACO podem resultar em menor ECT.

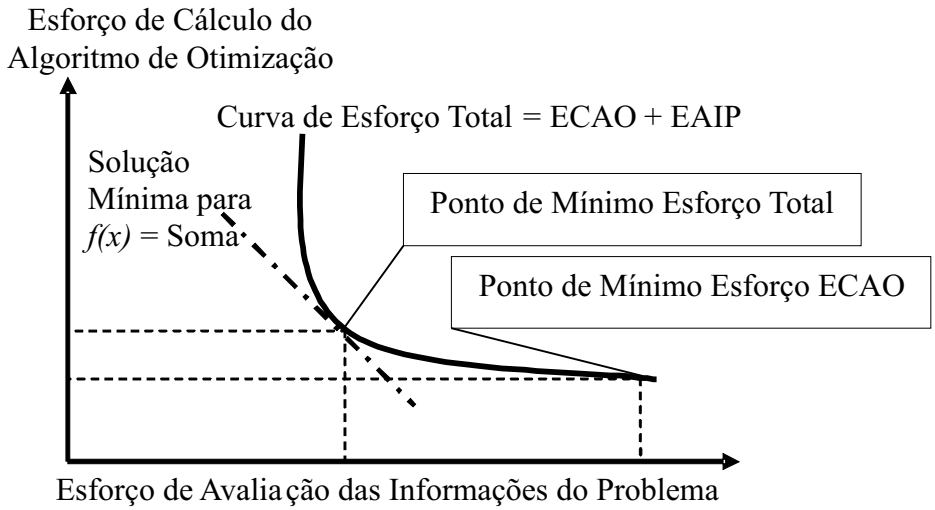


Figura 1.3: Esforço total para a solução de um problema de otimização.

1.2.3 Projetos de Otimização Assistidos por Computador

Na língua portuguesa, otimizar significa "tirar o máximo partido de; obter o melhor resultado de; tornar ótimo; planejar ou desenvolver com o máximo de eficiência" (DLPO 2005). Analogamente, no contexto da Engenharia, os sistemas de otimização se destinam a empregar técnicas e métodos para determinar a melhor solução de problemas abstratos, para os quais seja exeqüível quantificar o grau de adequação de cada possível solução às necessidades que os causaram (Takahashi 2003).

O uso cotidiano do termo otimizar acabou por confundi-lo com o sentido de melhorar. Certamente a otimização conduz a uma inequívoca melhora, já que a solução ótima é melhor no sentido superlativo, ou seja, melhor que todas as demais soluções. Todavia a recíproca, via de regra, não é verdadeira. Esta discussão faz-se necessária para o correto entendimento do termo Otimização no contexto da Engenharia, o qual está sempre associado à busca e obtenção da melhor solução, entre todas as possíveis, para um determinado problema. Qualquer mecanismo que melhore uma solução, mas não obtenha a melhor, não deve ser caracterizado como um projeto de otimização assistido por computador (POAC). Neste caso, o uso do termo "a melhor solução"é entendido como sendo a melhor solução prática, a qual não necessariamente coincidente com a solução teórica ótima.

Um sistema pode ser definido como um conjunto de princípios reunidos de modo a que formem um corpo de doutrina ou como combinação de partes coordenadas entre si e que concorrem para um resultado ou para formarem um conjunto (DLPO 2005). O desenvolvimento de um sistema que pretenda gerar uma solução ótima está intimamente relacionado com a qualidade e verossimilhança da representação a ser realizada para o problema. As características de um problema que ocorre no mundo real deverão

ser traduzidas para um mundo matemático. A representação do problema no ambiente matemático é denominada por modelo do problema ou simplesmente modelo. Nas diversas áreas da engenharia esta denominação pode variar, como exemplo, na área de mineração e metalurgia o termo mais utilizado é *modelo matemático*. A existência, portanto, de um modelo não caracteriza a obtenção de uma solução ótima, sendo apenas um pré-requisito para a implementação do sistema que a obterá. Neste mesmo mundo matemático, um método de otimização compreende a teoria que embasa a abordagem proposta para a solução do problema, enquanto uma implementação particular de um método de otimização ocorrerá na forma de um algoritmo de otimização. Um algoritmo de otimização será capaz de obter apenas a melhor solução para o modelo representado, portanto, quanto pior o modelo, maior será o erro entre a solução ótima obtida e a solução ótima contínua do problema. Por outro prisma, um modelo que pretenda descrever fidedignamente um problema muito complexo poderá inviabilizar, por tempo ou custo, a obtenção de um sistema de otimização capaz de resultar uma solução ótima. Um modelo é normalmente considerado determinístico quando especificar exatamente o comportamento de um experimento e é considerado probabilístico quando buscar descrever experimentos aleatórios ou com variações imprevisíveis (Garcia 1989). Na maioria dos casos práticos a formalização de um modelo, determinístico ou probabilístico, para o problema já está consolidada por teorias bem estabelecidas (Verhoeven 1975, Reed-Hill/Abbaschian & Abbaschian 1991, Hartman & Mutmansky 2002, Kou 2003, Narenda & Parthasarathy 1990, Caminhas, Pereira & Tavares 1998, Braga, Carvalho & Ludemir 2000) e não será objeto deste trabalho.

Supondo a existência de um único objetivo a ser considerado, um conceito a ser estabelecido em otimização diz respeito à quantificação da adequação de uma solução ao problema em questão. Cada possível solução para um problema é simbolizada por um vetor das n variáveis do modelo que representa o problema. A quantificação é usualmente associada ao objetivo do projeto (Takahashi 2003) ou, em outras palavras, a uma função matemática que expressa esta adequação, usualmente denominada de função-objetivo (Luenberger 1984) e neste caso representada por $f(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}$. Conseqüentemente alcançar a solução ótima, $\mathbf{x}^*$, para um problema significa encontrar matematicamente, dentre os diversos valores de $\mathbf{x}$ aquele que maximiza ou minimiza o valor da função-objetivo. Nesta Tese será sempre considerado o termo otimização correspondente a minimização, já que maximizar uma função $f(\cdot)$ equivale a minimizar $-f(\cdot)$, bem como as funções associadas aos problemas serão sempre pertencente à classe de funções quasi-convexas e não necessariamente diferenciáveis. A formulação matemática onde não existem limites para as variações das variáveis representadas por $\mathbf{x}$ caracteriza os problemas de *Otimização Irrestrita* é e representada por (1.5).

$$min\ f(\mathbf{x}) \tag{1.5}$$

Nas aplicações práticas as variáveis estão sempre sujeitas a limitações físicas, tecnológicas ou financeiras. Para representar estas restrições das soluções, definem-se equações de desigualdades ou igualdades matemáticas representadas por $g_i(\cdot) \mid i = 1, \ldots, p$, que delimitam a região onde a solução ótima deve estar contida. Esta região, representado por Ω, usualmente é denominado de região factível (Takahashi 2003). Um problema no qual se deseja encontrar o mínimo da função-objetivo $f(\cdot)$, mas cujo vetor de parâmetros $\mathbf{x}$, está sujeito à região factível Ω definida pelas p restrições $g(\cdot)$, caracteriza os problemas de *Otimização Restrita* e é representado por (1.6).

$$\begin{aligned} &min\ f(\mathbf{x}) \\ &s.a.\ \Omega = \{\mathbf{x} \in \mathbf{R}^n \mid g(\mathbf{x}) \leq 0\ \} \end{aligned} \tag{1.6}$$

onde $g(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^p$ e $\Omega \subset \mathbb{R}^n$.

Um caso particular da apresentação da equação 1.6 é encontrado quando todas as funções envolvidas são lineares, podendo portanto o problema para a *Otimização Linear* (Luenberger 1984, Dantzig 2002, Simmons 1972, Bradley, Hax & Magnati 1977), ser expresso pela equação 1.7.

$$\begin{aligned} &min\ \mathbf{c}^T\mathbf{x} \\ &s.a.\ \Omega = \{\mathbf{x} \in \mathbf{R}^n_+ \mid A\mathbf{x} = \mathbf{b}\} \end{aligned} \tag{1.7}$$

onde $\mathbf{c} \in \mathbb{R}^n$ e $\mathbf{b} \in \mathbb{R}^p$ representam vetores de constantes e $A \in \mathbb{R}^{n \times p}$ representa uma matriz de constantes.

Eliminando-se a condição, inicialmente definida, de que o problema a ser otimizado possui apenas um objetivo associado, ter-se-á não apenas uma função-objetivo para representar a adequação da solução, mas um conjunto de funções-objetivo. Os problemas onde existem apenas uma função-objetivo caracterizam a *Otimização Mono-objetivo ou Escalar*, enquanto os casos que contenham mais de uma função caracterizam a *Otimização Multiobjetivo ou Vetorial* (Chankong & Haimes 1982, Benson 1984, Adán & Novo 2003).

O problema VOP representado por (1.8) pode ser definido como a obtenção do vetor $\mathbf{x}^*$ e do conjunto Ω^*.

$$min\ f(\mathbf{x}) \tag{1.8}$$

$$s.a.\ \Omega^* = \{\mathbf{x}^* \in \Omega \mid \nexists\, \mathbf{x} \in \Omega\ tal\ que\ f(\mathbf{x}) \leq f(\mathbf{x}^*)\ e\ f(\mathbf{x}) \neq f(\mathbf{x}^*)\}$$

$$\Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}^n\}$$

onde, $f(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^m$ e $g(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^p$.

Como conseqüência da existência de um vetor de objetivos desejados, não haverá apenas uma solução ótima para o problema mas um conjunto de soluções ótimas, dado que em diversos casos não se poderá afirmar que uma possível solução $\mathbf{x}^A$ produzirá um resultado melhor do que outra $\mathbf{x}^B$ para todas as funções objetivos. Este conjunto de soluções ótimas, denominado como conjunto *Pareto* ou conjunto *Pareto-Ótimo* (Ehrgott 2000), é representado por Ω^* em (1.8).

Quando da obtenção de Ω^*, em virtude da existência de mais de uma solução ótima passível de ser implementada, surge o problema da decisão (Takahashi 2003), o qual consiste em restringir este conjunto a apenas uma solução por meio de um decisor. A indicação da preferência de um decisor por uma determinada solução ótima, poderá ocorrer *a posteriori*, *a priori* ou de forma *progressiva* ao processo de obtenção de Ω^*. Ressalta-se ainda que a ação do decisor deverá ocorrer de forma consistente, ou seja, partindo da apresentação sistemática de um subconjunto de alternativas, ter-se-á o afunilamento seqüencial das opções disponíveis para que seja possível convergir para uma escolha final. Matematicamente, o modelo deste comportamento consistente é formalizado por uma função denominada *função-utilidade*. Nesta classe de problemas de otimização vetorial destacam-se as formulações P_λ, P_ε, P_ξ, P_{KTE} e P^* (Takahashi 2003, Ehrgott 2000, Dias 2003) que permitem a utilização de um algoritmo escalar para que seja encontrado um ponto pertencente ao conjunto Ω^*.

Como abordagem final para a implementação de um POAC, é possível integrar os projetos aos problemas reais de forma *off-line* ou *on-line*. Em ambos os casos se encontrarão os elementos já definidos: modelo do problema, cálculo dos objetivos a serem otimizados e algoritmo de otimização.

Na integração *off-line* o POAC não possui qualquer conexão com o problema real, sendo normalmente utilizado como ferramenta de auxílio à solução de problemas ou projetos assistidos por computador. Neste caso, cabe a um projetista a tarefa de associar a solução ótima determinada pelo sistema ao processo de solução do problema real em questão. Um diagrama da integração *off-line* é exibido na figura 1.2.3.

Já na integração *on-line* o POAC está diretamente conectado a um processo produtivo, com ou sem a interferência humana, através de um sistema de controle. Neste caso, a

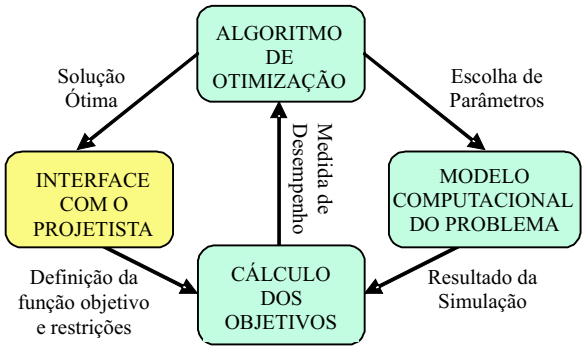


Figura 1.4: Diagrama da integração off-line de um PAOC.
Fonte: Adaptado de TAKAHASHI, R. Otimização escalar e vetorial. Belo Horizonte: UFMG, Departamento de Matemática, Curso de verão, 2003.

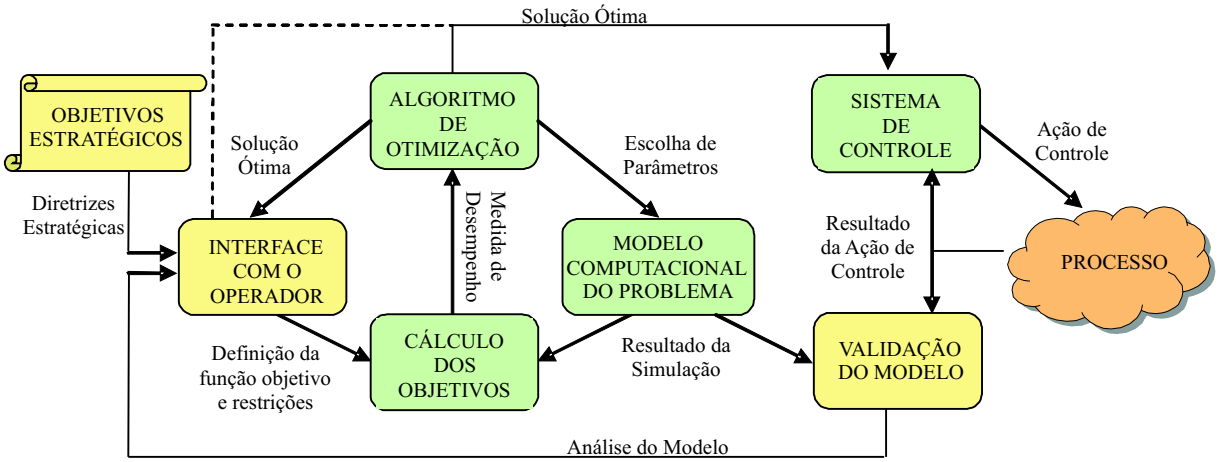


Figura 1.5: Diagrama da integração on-line de um PAOC.
Fonte: Adaptado de TAKAHASHI, R. Otimização escalar e vetorial. Belo Horizonte: UFMG, Departamento de Matemática, Curso de verão, 2003.

solução ótima será utilizada para determinar a região de operação mais eficiente para o processo, tendo por referência os objetivos pré-determinados. É comum, nestes sistemas, que um operador especializado no processo substitua a figura do projetista, bem como a inclusão de um processo de validação do modelo, dada a disponibilidade de dados reais de processo e dados resultantes do modelo. Um diagrama da integração *on-line* é exibido na figura 1.5.

1.3 Teses Propostas

Este trabalho propõe as seguintes teses quanto à atual formulação dos algoritmos pertencentes à família dos métodos baseados em *Elipsóide*:

Tese *i*) *E possível construir uma variante de um método baseado em Elipsóide, que resultará em um elipsóide de menor volume do que o proposto pelos métodos atuais de posto 1 e posto 2 existentes na literatura, quando da existência de mais de dois hiperplanos que restringem a região de interesse de busca da iteração atual* [1].

Tese *ii*) *Existem informações já calculadas, ao longo da convergência do algoritmo que, embora úteis à restrição da região de interesse de busca da iteração atual, são desconsideradas pelos métodos baseados em Elipsóide existentes na literatura.*

Tese *iii*) *É possível construir uma variante de um método baseado em Elipsóide com tempo de convergência total inferior aos tempos obtidos pela proposição original e variações atuais de posto 1 e posto 2 existentes na literatura para algumas classes de problemas.*

Tese *iv*) *É possível construir uma variante de um método baseado em Elipsóide com garantia de convergência global para problemas vetoriais quasi-convexos não necessariamente diferenciáveis.*

Tese *v*) *É possível definir e formalizar condições de infactibilidade que sejam necessárias e suficientes para demonstrar que um problema seja infactível, bem como permitam construir uma variante de um método baseado em Elipsóide o qual decorrido um número finito de iterações caracterizará um ponto que atenda a estas condições, quando da sua aplicação em um problema infactível.*

Tese *vi*) *É possível definir e formalizar uma variante de um método baseado em Elipsóide capaz de solucionar problemas com variáveis mistas discretas e contínuas, com funções quasi-convexas e com garantia de convergência global.*

Tese *vii*) *É possível definir e formalizar um método Branch-and-Cut, que utiliza uma variante de um método Elipsoidal para a solução do problema obtido por relaxação, capaz de solucionar problemas com variáveis mistas discretas e contínuas, com funções quasi-convexas e com garantia de convergência global.*

1.4 Estrutura desta Tese

Para permitir a contextualização desta Tese no âmbito da teoria de otimização com variáveis contínuas, é apresentada no capítulo 2 uma introdução dos métodos *Exclusão de Semi-Espaços*. Em seguida é apresentado o método *Elipsoidal* e, por sua relevância neste

[1] Para a existência de dois hiperplanos, vide os métodos descritos na seção 3.

trabalho, é detalhada a construção do novo elipsóide E_{k+1} pela proposição de Shor e pela utilização de cortes profundos. Também é descrita a forma com a qual este método trata as restrições, bem como algoritmos de fatorização para a melhora da estabilidade numérica deste método.

No capítulo 3 são apresentadas alternativas, existentes na literatura atual, de cálculo do novo elipsóide E_{k+1} através do *Corte Paralelo* e do *Corte em Cunha*, quando da existência de dois semi-planos para restringir a região de interesse de busca. Neste capítulo 3 também é apresentado o método *Dois Subgradientes Sucessivos* para problemas de otimização escalar e o método *Método Cone-Elipsoidal* para problemas vetoriais. No fim do capítulo é proposta uma variante para o método *Dois Subgradientes Sucessivos* e é apresentada uma falha para a garantia de convergência do método *Método Cone-Elipsoidal*.

No capítulo 4 é inicialmente apresentada uma revisão bibliográfica referente às condições de eficiência de Kuhn-Tucker. Em seguida é proposta uma nova condição necessária e suficiente de infactibilidade para problemas não-lineares escalares e vetoriais quasi-convexos, por similaridade estrutural às condições de eficiência. Também são definidos os *cones complementares*, quando da não satisfação das condições de eficiência e infactibilidade.

No capítulo 5 são propostos os fundamentos para os métodos *Poliedro-Elipsoidais* para a solução de problemas escalares e vetoriais com funções quasi-convexas e não necessariamente diferenciáveis com variáveis contínuas. No seu decorrer, os conceitos do poliedro de busca e do poliedro de busca com histórico de intersecções são formalizado por definições e teoremas, através dos quais será garantida a convergência global desta família de métodos propostos.

No capítulo 6 são propostos o método *Poliedro-Elipsoidal* por Hiperesfera, o método *Poliedro-Elipsoidal* por Seqüência de Cortes Profundos e Rasos e o método *Historical Intersection Search Polyhedron Ellipsoid Method*, os quais constituem a família dos métodos *Poliedro-Elipsoidais*. Estes algoritmos constituem a mais significante contribuição desta Tese para a solução de problemas com variáveis puramente contínuas.

O capítulo 7 encerra a primeira parte desta Tese, a qual se concentra na solução de problemas de otimização com variáveis puramente contínuas. Neste capítulo são exibidos os problemas escalares e vetoriais com variáveis puramente contínuas utilizados como teste e discutidos os respectivos resultados obtidos pelos algoritmos *Poliedro-Elipsoidais* e pelos algoritmos de referência para comparação.

O capítulo 8 inicia a segunda parte desta Tese, a qual estuda os problemas escalares com variáveis discretas e mistas discretas e contínuas. Neste capítulo são apresentados os fundamentos para o entendimento dos métodos de otimização que solucionam problemas

com variáveis discretas e mistas discretas e contínuas e são discutidas as limitações da aplicação direta dos métodos elipsoidais para a solução desta classe de problemas.

No capítulo 9 é apresentada a teoria para a geração de novas restrições, na forma de hiperplanos de corte, que permitem a obtenção da solução problema com variáveis discretas a partir da movimentação do limite inferior de convergência correspondente a solução obtida por relaxação. Em seguida é apresentada uma variação de um método baseado em *Elipsóide* capaz de resolver problemas escalares discretos.

No capítulo 10 é apresentada a teoria para a obtenção de métodos para a solução de problemas com variáveis discretas e mistas através do processo de enumeração, implícita ou explícita. Neste capítulo, inicialmente é definido um método baseado em enumeração explícita para a solução de problemas discretos não-lineares. No decorrer do capítulo, é apresentada uma introdução ao método *Branch-and-Bound*, bem como a busca proposta por este método é aplicada para a implementação de uma nova variação de um método baseado em *Elipsóide* com enumeração. Um método *Branch-and-Cut* também é definido e utilizado para criar duas variações de métodos baseados em *Elipsóide* para a solução de problemas com variáveis contínuas e mistas. Os quatro algoritmos propostos neste capítulo concluem a contribuição desta Tese para a solução de problemas escalares discretos e mistos.

No capítulo 11 são exibidos os problemas escalares com variáveis discretas e mistas utilizados como teste e discutidos os respectivos resultados obtidos pelos algoritmos *Poliedro-Elipsoidais* e pelos algoritmos de referência para comparação.

Por fim, o capítulo 1.2 apresenta as conclusão finais referentes a esta Tese e propõe algumas das mais promissoras idéias para a continuidade desta linha de pesquisa.

Parte I

Otimização com Variáveis Exclusivamente Contínuas

Capítulo 2

Métodos de Exclusão de Semi-Espaço e Elipsoidal

Neste capítulo é apresentada uma revisão bibliográfica referente à classe dos algoritmos de *Exclusão de Semi-Espaços*, com destaque ao algoritmo *Elipsoidal* por sua relevância neste trabalho. Também serão apresentados três métodos de fatorização que colaboram com o processo de manutenção da característica de positividade da matriz hessiana inversa do elipsóide.

Maiores detalhes sobre os temas tratados nas seções apresentadas neste capítulo podem ser conseguidos nas referências (Gill, Murray & Saunders 1975, Bierman 1976, Bierman 1977, Shor & Gershovich 1979, Khachiyan 1979, Bland, Goldfarb & Todd 1981, Luenberger 1984, Takahashi 2003).

2.1 Visão Geral sobre Métodos de Exclusão de Semi-Espaço

A classe dos métodos de *Exclusão de Semi-Espaços* utiliza a propriedade na qual o espaço pode ser dividido em dois semi-espaços por um hiperplano. Considerando-se que este hiperplano é definido por um ponto $\mathbf{x}$ e por um vetor $\bar{\nabla} f(\mathbf{x})$ pertencente ao conjunto subdiferencial de um funcional quasi-convexo $f(\cdot)$ avaliado em $\mathbf{x}$, em um destes dois semi-espaços o valor do funcional necessariamente aumenta em relação ao ponto avaliado.

O problema (1.5) será aqui tratado e é apresentado abaixo para a comodidade do leitor:

$$min\ f(\mathbf{x}) \tag{2.1}$$

Portanto, para o problema definido por (2.1), o algoritmo genérico para a classe de métodos de *Exclusão de Semi-Espaços* parte de um ponto inicial $\mathbf{x}_0 \neq \mathbf{x}^*$ e uma região inicial R_0 que contenha o mínimo $\mathbf{x}^*$. Em seguida o algoritmo irá gerar uma seqüência de pontos $\mathbf{x}_k$, tal que $\mathbf{x}_k$ tenda a $\mathbf{x}^*$ através da exclusão de parte da região inicial R_0 que não pertença ao semi-espaço $H_-^{\bar{\nabla} f(\mathbf{x}_k),\mathbf{x}_k}$. A estruturação deste algoritmo é apresentada no algoritmo 1:

Algorithm 1 Geral para os Métodos de *Exclusão de Semi-espaço*

Input: Uma região inicial de busca R_0, tal que $\mathbf{x}^* \in R_0$. Um ponto inicial $\mathbf{x}_0$.

Output: O ponto atual $\mathbf{x}^k$, não necessariamente factível.

function *Exclusão de Semi-espaço*(R_0)
  $k \leftarrow 0$
  loop ▷ Laço de exclusão de semi-espaço.
    $R_{k+1} \leftarrow V(R_k \cap H_-^{\bar{\nabla} f(\mathbf{x}_k),\mathbf{x}_k})$
    $\mathbf{x}_{k+1} \leftarrow T(R_{k+1})$
    if *Critério de parada* **then**
      return $\mathbf{x}_{k+1}$ ▷ Termine o algoritmo.
    end if
    $k \leftarrow k + 1$
  end loop
end function

Deve-se observar que a individualização dos métodos dar-se-á pela definição das funções $T(\cdot)$, a qual determina a nova estimativa para a solução a partir de uma região de busca R_k, e $V(\cdot)$, que estabelece o procedimento de construção da nova região de busca R_{k+1} a partir da região atual e do semi-espaço $H_-^{\bar{\nabla} f(\mathbf{x}_k),\mathbf{x}_k}$. Como decorrência de sua característica construtiva, os algoritmos de *Exclusão de Semi-Espaços* apresentam como garantia de valor decrescente o volume da região de interesse de busca e não o valor da função objetivo, que poderá oscilar ao longo da convergência.

Dentro desta classe de métodos, os algoritmos denominados de *Planos de Corte* foram os primeiros a serem propostos, sendo posteriormente apresentado o algoritmo *Elipsoidal*, o qual foi precursor dos algoritmos de *Pontos Interiores* desenvolvidos para problemas lineares e, mais recentemente, para problemas não-lineares. Para uma referência mais detalhada dos algoritmos de *planos de corte* vide (Luenberger 1984).

2.2 Algoritmos de Plano de Corte

Os métodos de *Planos de Corte* são aplicados a problemas definidos por (2.2):

$$min\ f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} \tag{2.2}$$

$$s.a.\ \mathbf{x} \in \Omega$$

$$\Omega \triangleq \text{é um conjunto convexo fechado}$$

Quando da utilização deste método em problemas cuja função objetivo é convexa e Ω é um conjunto convexo compacto, faz-se necessária a utilização de uma variável auxiliar $\overline{\mathbf{x}}$ para reescrever o problema no formato (2.3).

$$\overline{\mathbf{x}} \triangleq \{\ [r\ \mathbf{x}]^T \mid \overline{\mathbf{x}} \in \mathbb{R}^{n+1} \text{ e } r \in \mathbb{R}\} \tag{2.3}$$

Assim, o problema assume a forma (2.4):

$$min\ f(\mathbf{x}) = r \tag{2.4}$$

$$s.a.\ \mathbf{x} \in \overline{\Omega}$$

$$\overline{\Omega} \triangleq \{\overline{\mathbf{x}} \in \mathbb{R}^{n+1} \mid f(\mathbf{x}) - r \leq 0 \mid \mathbf{x} \in \Omega\}$$

O algoritmo geral dos métodos de *Planos de Corte* pode ser construído, considerando-se um erro ϵ e um dado politopo inicial P_0, tal que $P_0 \supset \overline{\Omega}$, conforme a estruturação a seguir:

Algorithm 2 Geral para os Métodos de *Planos de Corte*

Input: Um politopo inicial P_0, tal que $P_0 \supset \overline{\Omega}$. Um ponto inicial $\mathbf{x}_0$. Um valor da precisão para o teste de parada $\epsilon > 0$.

Output: O ponto atual $\mathbf{x}^k$, não necessariamente factível.

1: **function** *Planos de Corte*(P_0, ϵ)
2: $k \leftarrow 0$
3: $\mathbf{x}_k = \arg\min_{\mathbf{x}} \mathbf{c}^T\mathbf{x} \mid \mathbf{x} \in P_k$
4: **loop**
5: Determinar $H^{\xi,\mathbf{x}_k}$ que separa $\mathbf{x}_k$ de $\overline{\Omega}$, onde $(c^T\xi) > 0$
6: $P_{k+1} = P_k \cap H_-^{\xi,\mathbf{x}_k}$
7: $\mathbf{x}_{k+1} = \arg\min_{\mathbf{x}} \mathbf{c}^T\mathbf{x} \mid \mathbf{x} \in P_{k+1}$
8: **if** $\text{dist}(\mathbf{x}_k, \overline{\Omega}) \leq \epsilon$ **then**
9: **return** $\mathbf{x}_{k+1}$ ▷ Termine o algoritmo.
10: **end if**
11: $k \leftarrow k + 1$
12: **end loop**
13: **end function**

Quanto aos algoritmos de *Planos de Corte*, deve-se ainda registrar que a seqüência de convergência dos pontos $\mathbf{x}$ dar-se-á de forma assintótica e externamente ao conjunto $\overline{\Omega}$, uma vez que as soluções da etapa de otimização linear sempre determina pontos referentes a um dos vértices do novo politopo P_{k+1} que contém $\overline{\Omega}$. Ao se avaliar o processo de construção do novo politopo P_{k+1}, verifica-se a inclusão de uma nova restrição, determinada pelo acréscimo do plano $H^{\xi,\mathbf{x}_k}$ que separa $\mathbf{x}$ de $\overline{\Omega}$, a cada iteração. Esse crescimento do conjunto de restrições impõe ao algoritmo uma velocidade de convergência considerada ineficiente se comparada a outros métodos propostos posteriormente.

A figura 2.1 exibe uma seqüência de convergência para um algoritmo de *Planos de Corte*.

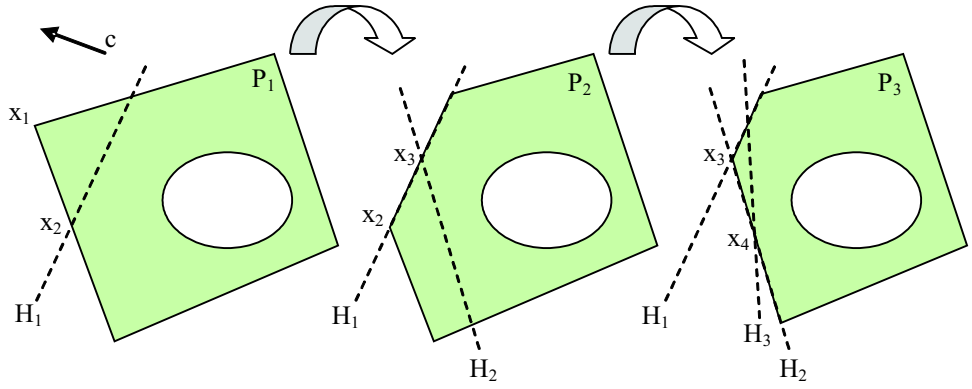


Figura 2.1: Ilustração de uma seqüência de convergência para um algoritmo de *Planos de Corte*.

A principal diferença entre os algoritmos da família de *Planos de Corte*, diz respeito à forma na qual o plano $H^{\xi,\mathbf{x}_k}$ é determinado. Assim, mesmo partindo de condições iniciais idênticas, diferentes algoritmos de *Planos de Corte* podem resultar em distintas seqüências de convergência para os pontos $\mathbf{x}$ e diferentes tempos para a obtenção da solução.

2.2.1 Algoritmo de Plano de Corte de Kelley

A proposição de Kelley (Kelley 1960) para a escolha dos planos de corte $H^{\xi,\mathbf{x}_k}$ é aplicada à solução de problemas da forma apresentada por (2.5), com a premissa de que $g(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^p$:

$$min\ f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} \tag{2.5}$$

$$s.a.\ \mathbf{x} \in \Omega$$

$$\Omega \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid g_i(\mathbf{x}) \leq 0\ \forall\ i = 1, \ldots, p\}$$

Assim, temos como válida a desigualdade exibida em (2.6). Considerando-se que i

representa o índice da restrição mais violada, o plano $H^{\xi,\mathbf{x}_k}$ deverá ser calculado por (2.7) e utilizado para calcular o novo politopo P_{k+1}, tal como mostrado no algoritmo 2.

$$g_i(\mathbf{v}) \geq g_i(\mathbf{w}) + \nabla g_i(\mathbf{v} - \mathbf{w}) \quad \forall\, \mathbf{v}, \mathbf{w} \tag{2.6}$$

$$H^{\xi,\mathbf{x}_k} = \{\, g_i(\mathbf{x}_k) + \nabla g_i(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) = 0 \,\} \tag{2.7}$$

2.3 Método *Elipsoidal*

O desenvolvimento do método *Elipsoidal* é baseado nos precedentes algoritmos de *Relaxação para Desigualdades*, proposto simultaneamente por Agmon (Agmon 1954) e Motzkin e Schoenberg (Motzkin & Schoenberg 1954), método do *Subgradiente* e *Dilatação do Espaço*, proposto por Shor (Shor 1964, Shor 1970b, Shor 1970a), e pelo método das *Seções Centrais*, desenvolvidos independentemente por Levin (Levin 1965) e Newman (Newman 1965).

A primeira descrição do método *Elipsoidal* é encontrada, embora não de forma explícita, em Iudin e Nemirovskii (Iudin & Nemirovskii 1976). Neste mesmo artigo, é indicado que o método *Elipsoidal* constitui um caso especial do algoritmo de *Dilatação do Espaço* de Shor, o qual posteriormente acabou por apresentar a proposição explícita do método *Elipsoidal* (Shor 1977), tal como conhecido atualmente. A partir da importante demonstração por Khachiyan de que o algoritmo *Elipsoidal* apresentava a capacidade de resolver um problema linear em tempo polinomial (Khachiyan 1979), o algoritmo ganhou notoriedade e passou a ser amplamente estudado e aplicado para a resolução de problemas de otimização associados a funções quasi-convexas. Maiores detalhes sobre o método *Elipsoidal* podem ser encontrados na pesquisa elaborada por Bland, Goldfarb e Todd (Bland et al. 1981).

2.3.1 Algoritmo *Elipsoidal*

Considere o problema descrito a seguir, onde $f(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}$, $g(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^p$ e Ω determina uma região factível para do problema:

$$min\ f(\mathbf{x}) \tag{2.8}$$

$$s.a.\ \mathbf{x} \in \Omega$$

$$\Omega \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid g_j(\mathbf{x}) \leq 0,\ \forall\, j = 1, \ldots, p\}$$

O algoritmo *Elipsoidal* começa com uma elipse E_0, vide (2.9), centrada no ponto $\mathbf{x_0}$, que contenha $\mathbf{x}^*$.

$$E_0 \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid (\mathbf{x}-\mathbf{x}_0)^T Q_0^{-1}(\mathbf{x}-\mathbf{x}_0) \leq 1\} \tag{2.9}$$

Tal como já descrito na seção 2.1, considerando-se $f(\mathbf{x})$ e $g_j(\mathbf{x})$, $\forall\, j = 1, \ldots, p$ funções quasi-convexas, pode-se afirmar que o mínimo $\mathbf{x}^*$ pertence ao semi-espaço $H_-^{\bar{\nabla}_k, \mathbf{x}_k}$.

$$H_-^{\bar{\nabla}_k, \mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \bar{\nabla}_k^T(\mathbf{x}-\mathbf{x}_k) \leq 0\} \tag{2.10}$$

onde, $\bar{\nabla}_k$ representa um vetor pertencente ao conjunto subdiferencial de uma das funções $\bar{\nabla} g_j(\mathbf{x}_k)$, quando $\mathbf{x}_k \notin \Omega$, ou um vetor pertencente ao conjunto subdiferencial de $\bar{\nabla} f(\mathbf{x}_k)$, nos demais casos. O tratamento das restrições será detalhado na seção 2.3.3.

Aplicando-se as fórmulas recursivas abaixo descritas, obter-se-á uma seqüência de pontos $\mathbf{x}_{k+1}$ correspondentes ao centro dos elipsóides E_{k+1}, resultantes da intersecção do semi-espaço $H_-^{\bar{\nabla}_k, \mathbf{x}_k}$ com o elipsóide E_k, tal que $E_{k+1} \supset (E_k \cap H_-^{\bar{\nabla}_k, \mathbf{x}_k})$.

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k - \tau(Q_k\bar{\nabla}_k/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2}) \\ Q_{k+1} &= \sigma[Q_k - \delta(Q_k\bar{\nabla}_k)(Q_k\bar{\nabla}_k)^T/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)] \end{aligned} \tag{2.11}$$

onde, $\tau = \frac{1}{n+1}$ é chamado de parâmetro de passo, $\sigma = \frac{2}{n+1}$ de parâmetro de dilatação e $\delta = \frac{n^2}{n^2-1}$ de parâmetro de expansão.

A figura 2.2 exibe a construção de um elipsóide E_{k+1} a partir da preservação do semi-elipsóide determinado por $H_-^{\bar{\nabla}_k, \mathbf{x}_k}$, exibindo uma visão geométrica para os parâmetros supracitados. Como pode ser observado, para o caso de E_k ser uma hiperesfera de raio unitário, o novo elipsóide E_{k+1} resultará da contração de E_k na direção oposta ao vetor $\bar{\nabla}_k$ e na proporção de $(\delta(1-\sigma))^{1/2} = n/(n+1)$, enquanto as demais direções ortogonais à direção de $\bar{\nabla}_k$ sofrerão uma expansão na proporção de $\delta^{1/2} = n/(n^2-1)^{1/2}$.

Uma vez que o elipsóide inicial E_0 contém a solução ótima $\mathbf{x}^*$ e que a seqüência dos elipsóides gerados por (2.11) preserva o semi-espaço que o contém, verifica-se que a convergência conduzirá a um elipsóide de volume zero e, quando este contiver um único ponto, este ponto será $\mathbf{x}^*$.

O algoritmo *Elipsoidal* proposto por Shor é apresentado a seguir:

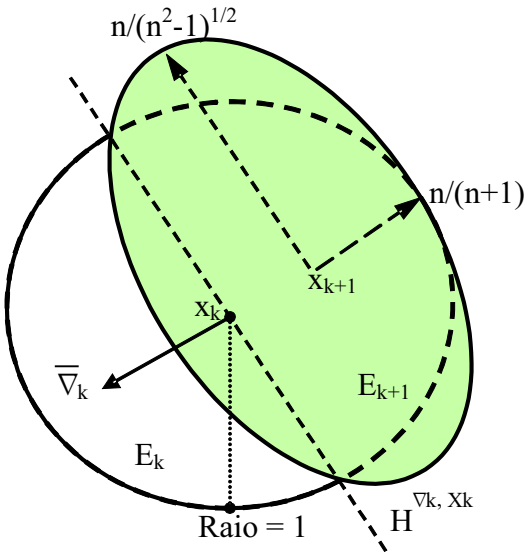


Figura 2.2: Ilustração de uma iteração do método *Elipsoidal* e interpretação geométrica dos seus parâmetros.

2.3.2 Algoritmo *Elipsoidal* com *Deep Cut*

A primeira proposta para a utilização de *Deep Cuts* para aumentar a velocidade de convergência do algoritmo *Elipsoidal* foi apresentada por Shor e Gershovich (Shor & Gershovich 1979).

Considere uma variante para o hiperplano $H^{\bar{\nabla}_k,\mathbf{x}_k}$ e para o semi-espaço $H_-^{\bar{\nabla}_k,\mathbf{x}_k}$, tal como definida a seguir:

$$H(\alpha,Q_k)^{\bar{\nabla}_k,\mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \bar{\nabla}_k^T(\mathbf{x}-\mathbf{x}_k) = \alpha(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2}\} \tag{2.12}$$

$$H_-(\alpha,Q_k)^{\bar{\nabla}_k,\mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \bar{\nabla}_k^T(\mathbf{x}-\mathbf{x}_k) \leq \alpha(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2}\} \tag{2.13}$$

onde o parâmetro α é denominado de profundidade do corte e representa a distância do hiperplano $H(\alpha,Q_k)^{\bar{\nabla}_k,\mathbf{x}_k}$ ao ponto $\mathbf{x}_k$ na métrica correspondente à matriz Q_k.

Como o semi-espaço $H_-(\alpha,Q_k)^{\bar{\nabla}_k,\mathbf{x}_k}$ não mais contém o ponto $\mathbf{x}_k$ que representa o centro do elipsóide E_k, verifica-se que a intersecção entre $H_-(\alpha,Q_k)^{\bar{\nabla}_k,\mathbf{x}_k} \cap E_k$ não mais preserva a metade do elipsóide E_k. Por conseguinte, o elipsóide E_{k+1} gerado possui volume inferior ao que seria obtido no algoritmo *Elipsoidal* de Shor, descrito por (2.11).

Com o conjunto de equações abaixo, obter-se-á uma nova seqüência de pontos $\mathbf{x}_{k+1}$ associados ao centro dos elipsóides E_{k+1}, resultantes da intersecção do semi-espaço $H_-(\alpha,Q_k)^{\bar{\nabla}_k,\mathbf{x}_k}$ com o elipsóide E_k.

Algorithm 3 *Elipsoidal* proposto por Shor

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^*$ será procurada de acordo com o problema definido por (2.8).

Output: A melhor solução factível contínua $\mathbf{x}^f$ encontrada. Para um problema infactível ou para $\mathbf{x}^* \notin E_0$ ter-se-á $\mathbf{x}^f = \emptyset$.

1: **function** *Shor*(E_0)
2: $k \leftarrow 0$
3: $\mathbf{x}^f \leftarrow \emptyset$ ▷ Zere a melhor solução factível.
4: **loop**
5: **if** $g_i(\mathbf{x}_k) \leq 0 \; \forall \; i = 1, \ldots, p$ **then**
6: $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{x}_k)$
7: $\mathbf{x}^f \leftarrow \mathbf{x}_k$ ▷ Melhor solução factível até o momento.
8: **if** $\bar{\nabla}_k = 0$ **then**
9: **return** $\mathbf{x}^f$ ▷ Termine o algoritmo.
10: **end if**
11: **else**
12: $\bar{\nabla}_k \leftarrow \bar{\nabla} g_i(\mathbf{x}_k)$ *para algum* $g_i(\mathbf{x}_k) > 0, \mid i = 1, \ldots, p$
13: **end if**
14: $E_{k+1} \leftarrow (E_k \cap H_-^{\bar{\nabla}_k, \mathbf{x}_k})$ ▷ Vide (2.11)
15: **if** vol $(E_{k+1}) \simeq 0$ ou $vol(E_{k+1}) \simeq vol(E_k)$ **then**
16: **return** $\mathbf{x}^f$ ▷ Termine o algoritmo.
17: **end if**
18: $k \leftarrow k + 1$
19: **end loop**
20: **end function**

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \tau(Q_k \bar{\nabla}_k \; / \; (\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2})$$
$$Q_{k+1} = \sigma[Q_k - \delta(Q_k \bar{\nabla}_k)(Q_k \bar{\nabla}_k)^T \; / \; (\bar{\nabla}_k^T Q_k \bar{\nabla}_k)] \tag{2.14}$$

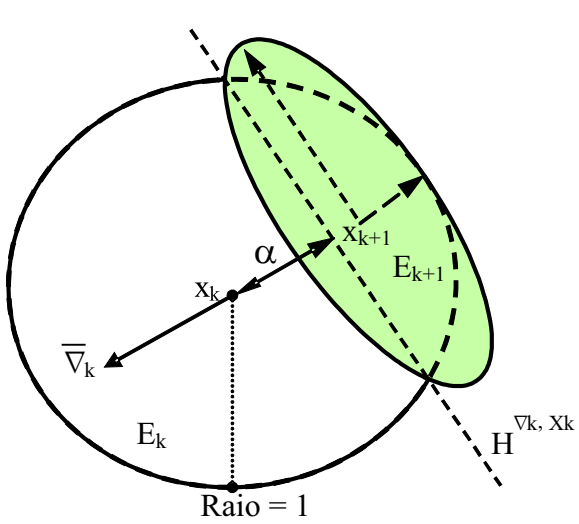
onde, $\tau = \frac{1+n\alpha}{n+1}$ é chamado de parâmetro de passo, $\sigma = \frac{2(1+n\alpha)}{(n+1)(1+\alpha)}$ de parâmetro de dilatação e $\delta = (\frac{n^2}{n^2-1})(1-\alpha^2)$ de parâmetro de expansão.

Deve-se observar que, embora as expressões que atualizam $\mathbf{x}_{k+1}$ e Q_{k+1} sejam correspondentes às utilizadas para o cálculo convencional do algoritmo, o cálculo dos parâmetros τ, σ e δ será função do novo parâmetro α que define $H_-(\alpha, Q_k)^{\bar{\nabla}_k, \mathbf{x}_k}$.

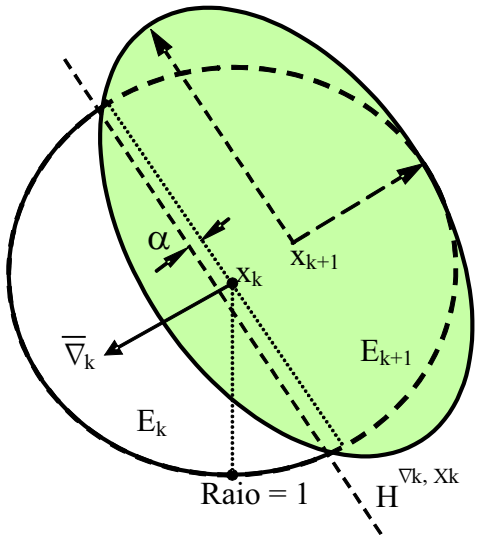
Para a determinação de E_{k+1}, a equação (2.14) é válida no intervalo $\{-1/n \leq \alpha \leq 1\}$, dado que para valores $\alpha \leq -1/n$ ter-se-á $E_{k+1} = E_k$. Para valores de $0 \leq \alpha < 1$, denominados de *Cortes Profundos* ou *Deep Cuts*, verifica-se que o novo cálculo conduzirá a uma aceleração da contração do volume de E_{k+1}, sendo que para $\alpha = 1$ o elipsóide E_{k+1} degenera para um ponto. Deve-se contudo observar que os *Deep Cuts* não garantem que a

solução ótima será preservada em todas as iterações. A faixa de valores $\{-1/n < \alpha < 0\}$ define um conjunto de cortes considerados *rasos*, ou *Shallow*, uma vez que estes preservam mais da metade do elipsóide anterior E_k. Embora pouco explorados pela literatura até o presente momento, estes cortes serão, tal como demonstrado em capítulo futuro, de grande utilidade para a construção dos métodos *Cone-Elipsoidais* e *Histórico-Cone-Elipsoidais*.

A figura 2.3 exibe dois exemplos de cortes deslocados do centro $\mathbf{x}_k$ para a construção do novo elipsóide E_{k+1}.



a) Seqüência com *Deep Cut*.



b) Seqüência com *Shallow Cut*.

Figura 2.3: Ilustração de duas iterações do método *Elipsoidal* com corte deslocado do centro $\mathbf{x}_k$.

2.3.3 Tratamento das Restrições

A classe dos métodos de *exclusão de semi-espaço* tem como característica particular o tratamento direto das restrições associadas ao problema e, portanto, não possui as desvantagens resultantes da utilização de funções *barreira* ou *penalidade*, tais como mal condicionamento numérico ou aproximação inexata das funções restrições.

Assim, embora o tratamento de restrições não lineares genéricas continue sendo um desafio considerável mesmo para os métodos de otimização atualmente conhecidos (Luenberger 1984), a estrutura do algoritmo *Elipsoidal* lhe permite para o cálculo de um ponto fora da região factível, $\mathbf{x}_k \notin \Omega$, a simples atribuição de $\bar{\nabla}_k$ por um vetor pertencente ao conjunto dos sub-diferenciais de uma restrição violada. Isto se deve ao fato de que para um conjunto de restrições quasi-convexas, e conseqüentemente Ω um conjunto quasi-convexo, um vetor pertencente ao conjunto dos sub-diferenciais de uma restrição violada sempre determinará o semi-espaço que conterá, necessariamente, o conjunto Ω.

A região factível Ω pode ser definida como a intersecção das regiões de sub-nível zero das funções-restrição:

$$\Omega = L_{g_1=0} \cap \ldots \cap L_{g_p=0} \tag{2.15}$$

Analogamente, Ω pode ainda ser interpretada como a região de sub-nível zero de uma função $\bar{g}(\cdot)$, definida como:

$$\bar{g}(\mathbf{x}) = max(\{g_j(\mathbf{x}),\, \forall\, j = 1, \ldots, p\}) \tag{2.16}$$

Notoriamente, a minimização de $\bar{g}(\mathbf{x})$ irá excluir, a cada passo em que $\mathbf{x}_k$ seja infactível, um semi-espaço totalmente infactível, convergindo para um ponto factível, caso este exista. Definindo-se uma função $s(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}$, tal que:

$$s(\cdot) = \begin{cases} f(\cdot) & \text{se } \bar{g}(\mathbf{x}) \leq 0, \\ \bar{g}(\cdot) & \text{se } \bar{g}(\mathbf{x}) > 0. \end{cases} \tag{2.17}$$

Desta feita, a otimização de $s(\mathbf{x})$ convergirá para o ótimo $\mathbf{x}^*$, uma vez que, para um ponto $\mathbf{x}_k$ factível ocorrerá a exclusão do semi-espaço que não contém a solução ótima, enquanto que, para um ponto infactível, ocorrerá a exclusão de um semi-espaço totalmente infactível.

2.3.4 Fatorização de *Cholesky*, LDL^T e UDU^T

Em (2.11) o elipsóide E_{k+1} centrado no ponto $\mathbf{x}_{k+1}$ é representado pela matriz simétrica definida positiva Q_{k+1}. Uma vez que a convergência do algoritmo é obtida através de uma implementação em laço, com o recálculo de Q_{k+1} a cada iteração, é inevitável que a precisão finita dos computadores acarrete arredondamento, os quais poderão conduzir à perda da positividade de Q_{k+1}, caso o critério de parada não seja previamente satisfeito.

Uma forma de aumentar a estabilidade numérica do algoritmo é implementar uma fatorização (Bland et al. 1981, Gill et al. 1975, Bierman 1976, Bierman 1977) para o cálculo de Q_{k+1}. Duas das possíveis opções são:

Fatorização de *Cholesky*: Uma vez que Q é matriz simétrica definida positiva, pode ser utilizada a fatorização $Q = JJ^T$, onde J é uma matriz triangular inferior. Esta perspectiva resultará na utilização de um menor volume de memória necessária para armazenar Q e de um menor número de operações matemáticas para a atualização de J_{k+1}. Assim, as equações de atualização de E_{k+1} serão:

$$\begin{aligned}
\mathbf{x}_{k+1} &= \mathbf{x}_k - \tau J_k(J_k^T\bar{\nabla}_k)/(\bar{\nabla}_k^T J_k\bar{\nabla}_k) \\
\widetilde{J}_{k+1} &= \sigma^{1/2}J_k[I-(1-(1-\delta)^2)(J_k^T\bar{\nabla}_k)(J_k^T\bar{\nabla}_k)^T/(\bar{\nabla}_k^T J_k\bar{\nabla}_k)^2] \\
J_{k+1} &= cholesky(\widetilde{J}_{k+1}\widetilde{J}_{k+1}^T)
\end{aligned} \tag{2.18}$$

onde I é a matriz identidade de ordem n e $cholesky(\cdot)$ representa a função de fatorização de Cholesky, sendo que E_{k+1} passa a ser definida pela matriz J_{k+1}, ou seja:

$$E_{k+1} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid (\mathbf{x}-\mathbf{x}_{k+1})^T(J_{k+1}J_{k+1}^T)^{-1}(\mathbf{x}-\mathbf{x}_{k+1}) \leq 1\} \tag{2.19}$$

Fatorização LDL^T e UDU^T: Uma forma ainda mais eficiente de estabilizar a matriz Q é considerar $Q = LDL^T$, onde L é uma matriz unitária triangular inferior e D é uma matriz diagonal definida positiva. Um algoritmo numérico para a atualização de D_{k+1} que garante que a matriz seja positiva definida pode ser encontrado em (Gill et al. 1975). Analogamente, pode-se considerar que $Q = UDU^T$, onde U é uma matriz unitária triangular superior e D é uma matriz diagonal definida positiva. Assim, as equações de atualização de E_{k+1} serão:

$$\begin{aligned}
\mathbf{x}_{k+1} &= \mathbf{x}_k - \tau Q_k\bar{\nabla}_k/(\bar{\nabla}_k^T Q_k\bar{\nabla}_k)^{1/2} \\
\widetilde{D}_{k+1} &= \sigma[D_k - \delta(D_kU_k^T\bar{\nabla}_k)(D_kU_k^T\bar{\nabla}_k)^T/(\bar{\nabla}_k^T U_kD_kU_k^T\bar{\nabla}_k)] \\
[\widetilde{U}_{k+1}, D_{k+1}] &= ud(\widetilde{D}_{k+1}) \\
U_{k+1} &= U_k\widetilde{U}_{k+1} \\
Q_{k+1} &= \widetilde{U}_{k+1}\widetilde{D}_{k+1}\widetilde{U}_{k+1}^T
\end{aligned} \tag{2.20}$$

onde, $[U_0, D_0] = ud(Q_0)$.

Um algoritmo para a implementação da função de fatorização $ud(M)$ é exibido no apêndice 12.2.

A figura 2.4 mostra o resultado[1] comparativo de uma implementação do algoritmo *Elipsoidal* de Shor e com a utilizando da fatorização de *Cholesky*, e da fatorização UDU^T. Para tanto, foi utilizado um problema irrestrito quadrático do tipo $f(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_c)^T B(\mathbf{x} - \mathbf{x}_c)$ onde, claramente, $\mathbf{x}_c$ representa a solução mínima para $f(\mathbf{x})$. Não foram definidos critérios de parada, de modo que os algoritmos executassem até a ocorrência de falha. Tal como esperado, verifica-se que a fatorização UDU^T conseguiu produzir um maior número de iterações, o que corresponde à obtenção de um menor volume final do elipsóide que contém a solução do problema. Percebe-se que a fatorização de *Cholesky* também proporcionou maior estabilidade de cálculo ao algoritmo *Elipsoidal*. Em ambos os casos de utilização da fatorização a solução ótima encontrada com tempos equivalentes e com $\mathbf{x}^*$ foi igual à solução do problema no limite da precisão do computador, ou seja, $(\mathbf{x}_c - \mathbf{x}^*)^T(\mathbf{x}_c - \mathbf{x}^*) = 0$. Assim, ambos os algoritmos de fatorização apresentarão desempenho equivalente para aplicações práticas.

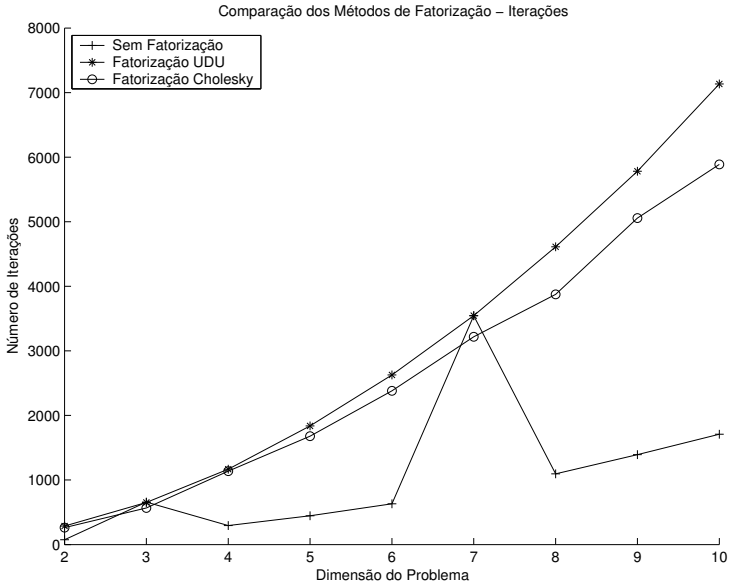
2.4 Conclusões do Capítulo

Neste capítulo foi apresentada uma revisão bibliográfica referente à classe dos algoritmos de *Exclusão de Semi-Espaços*, com destaque ao algoritmo *Elipsoidal* por sua relevância neste trabalho. Também foram apresentados três métodos de fatorização que colaboram com o processo de manutenção da característica de positividade da matriz hessiana inversa do elipsóide.

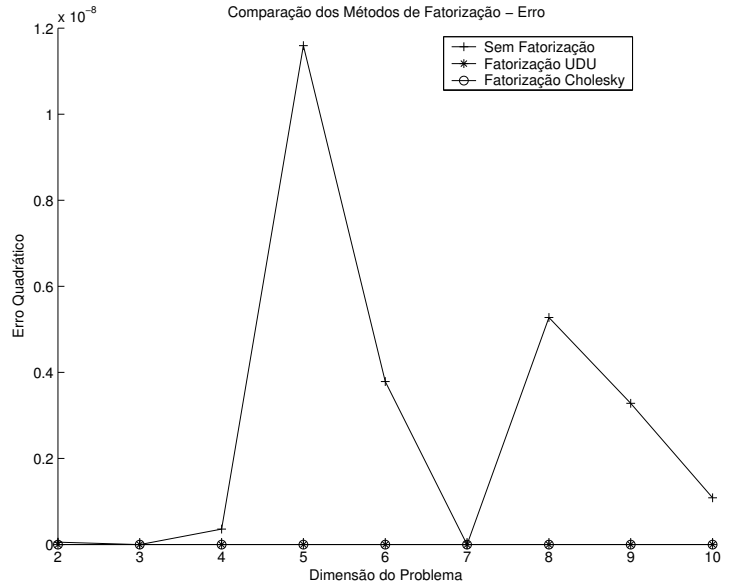
Neste capítulo foram inicialmente apresentados a classe dos algoritmos de *exclusão de semi-espaços* e o algoritmo *Elipsoidal* de Shor e *Deep Cut*. Em seguida foram revistos os métodos de fatorização para manutenção da característica de positividade da matriz hessiana inversa do elipsóide.

No capítulo seguinte serão detalhados os métodos derivados do algoritmo *Elipsoidal* clássico que utilizam a proposição de posto 2 para a definição do novo elipsóide, uma vez que estes são os precursores do conceito de utilização de mais de uma informação simultânea sobre a restrição da região de interesse de busca, para produzir a aceleração da velocidade de convergência do método.

[1] Média dos valores obtidos com a solução de 10 problemas cujos parâmetros B e $\mathbf{x}_c$ foram gerados aleatoriamente para cada dimensão utilizada.



a) Número de Iterações em Função da Dimensão do Problema.



b) Erro Quadrático em Função da Dimensão do Problema.

Figura 2.4: Comparação dos métodos de fatorização para aumento da estabilidade de algoritmo *Elipsoidal*.

Capítulo 3

Métodos que Usam Simultaneamente Mais de Um Vetor Subdiferencial para Cálculo de E_{k+1}

Neste capítulo é apresentada uma revisão da principal literatura existente sobre os métodos derivados do algoritmo *Elipsoidal* que utilizam mais de um vetor subdiferencial, simultaneamente, para o cálculo do novo elipsóide E_{k+1}.

Inicialmente é caracterizado o *Corte Paralelo* e a formulação para cálculo do novo elipsóide para este corte. Em seguida é apresentado o *Corte em Cunha*, bem como as equações para determinação do novo elipsóide para este corte. Também é descrito o algoritmo *Elipsoidal* que utiliza *Dois Subgradientes Sucessivos* para a aceleração da contração do volume do novo elipsóide E_{k+1}. Por fim é apresentado o *Método Cone-Elipsoidal* para problemas vetoriais.

Maiores detalhes sobre os temas tratados nas seções apresentadas neste capítulo podem ser conseguidos nas referências (Todd 1982, Ech-Cherif & Ecker 1984, Kim, Kim & Chang 1994, Dias 2003).

3.1 Cortes Paralelos

Tal como descrito na seção 2.3, o algoritmo *Elipsoidal* utiliza apenas um único subgradiente que determina o semi-espaço $H_{-}^{\bar{\nabla}_k,\mathbf{x}_k}$, vide (2.10), para a determinação do novo elipsóide E_{k+1} por (2.11). Conseqüentemente, ocorre a preservação de metade do elipsóide atual, fazendo com que a redução deste volume se dê em uma taxa constante durante a seqüência de convergência. Em (3.1) exibe-se a taxa de redução do volume do método *Elipsoidal* e na figura 3.1 mostra-se a taxa de redução do volume em função do aumento

da dimensão de um problema.

$$vol(E_{k+1})/vol(E_k) = (n/(n+1))(n^2/(n^2-1))^{(n-1)/2} \tag{3.1}$$

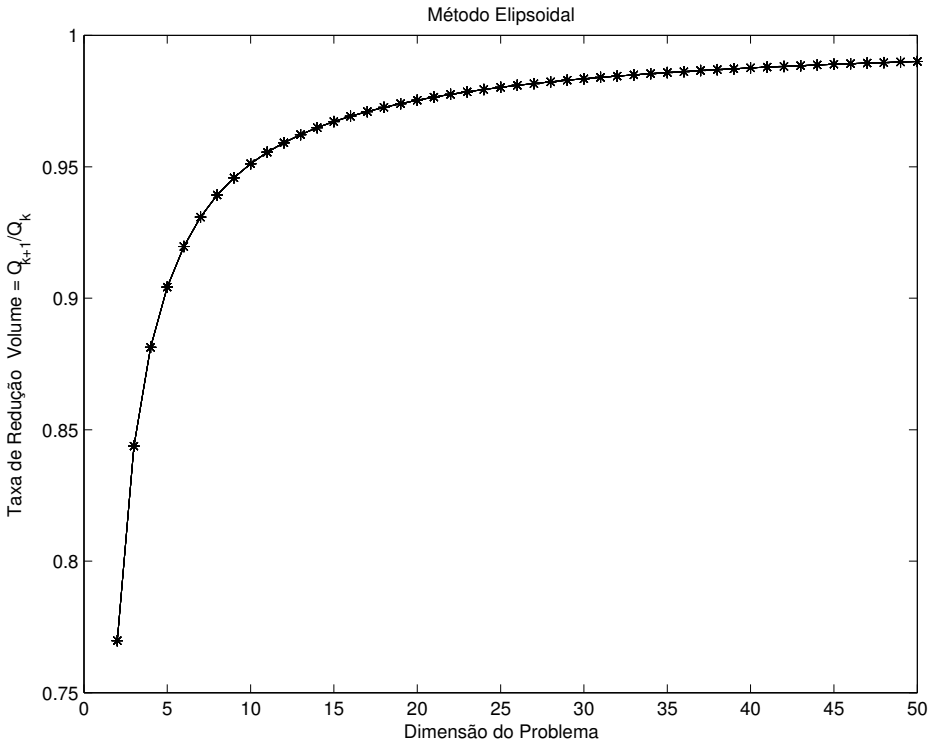


Figura 3.1: Taxa de redução do volume do método *Elipsoidal* em função do aumento da dimensão do problema.

Quando da existência de dois hiperplanos paralelos $H(-\alpha, Q_k)^{\bar{\nabla}_k, \mathbf{x}_k}$ e $H(\beta, Q_k)^{-\bar{\nabla}_k, \mathbf{x}_k}$ determinados por um vetor $\bar{\nabla}_k$, pode-se definir um novo conjunto $H([\alpha, \beta], Q_k)^{\bar{\nabla}_k, \mathbf{x}_k} = E_k \cap H_-(-\alpha, Q_k)^{\bar{\nabla}_k, \mathbf{x}_k} \cap H_-(\beta, Q_k)^{-\bar{\nabla}_k, \mathbf{x}_k}$ que conterá a intersecção destes semi-espaços com o elipsóide atual E_k, tal como indicado nas equações abaixo. O conjunto $H([\alpha, \beta], Q_k)^{\bar{\nabla}_k, \mathbf{x}_k}$ define o chamado *Corte Paralelo* (Todd 1982).

$$H_-(-\alpha, Q_k)^{\bar{\nabla}_k, \mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \bar{\nabla}_k^T(\mathbf{x} - \mathbf{x}_k) \leq -\alpha(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2}\} \tag{3.2}$$

$$H_-(\beta, Q_k)^{-\bar{\nabla}_k, \mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid -\bar{\nabla}_k^T(\mathbf{x} - \mathbf{x}_k) \leq \beta(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2}\} \tag{3.3}$$

$$H([\alpha, \beta], Q_k)^{\bar{\nabla}_k, \mathbf{x}_k} \triangleq \{\mathbf{x} \in E_k | \alpha(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2} \leq -\bar{\nabla}_k^T(\mathbf{x} - \mathbf{x}_k) \leq \beta(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2}\} \tag{3.4}$$

onde, $\{-1 \leq \alpha \leq \beta \leq 1\}$.

A utilização dos parâmetros α, β, $\bar{\nabla}_k$ e E_k permite o cálculo do elipsóide mínimo

$E_{k+1} \supset H([\alpha,\beta], Q_k)^{\bar{\nabla}_k, \mathbf{x}_k}$ [1] como exibido na equação abaixo, onde se considera que $\alpha < \beta$ de modo que o elipsóide E_{k+1} tenha volume positivo. O caso especial no qual $\alpha = \beta$ resultará no elipsóide de dimensão $\mathbb{R}^{n-1}$.

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + \tau(Q_k\bar{\nabla}_k)/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2} \\ Q_{k+1} &= \sigma[Q_k - \delta(Q_k\bar{\nabla}_k)(Q_k\bar{\nabla}_k)^T)/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)] \\ E_{k+1} &\triangleq \{\mathbf{x} \in \mathbb{R}^n \mid (\mathbf{x}-\mathbf{x}_{k+1})(Q_{k+1})^{-1}(\mathbf{x}-\mathbf{x}_{k+1}) \leq 1\} \end{aligned} \tag{3.5}$$

onde,

i) Para $\alpha\beta \leq -1/n$, o elipsóide E_{k+1} será mínimo se $\tau = \sigma = 0$ e $\delta = 1$, ou seja, $E_{k+1} = E_k$

ii) Para $\alpha + \beta = 0$, $\alpha < \beta$ e $\alpha\beta \geq -1/n$, o elipsóide E_{k+1} será mínimo se:

$$\sigma = \frac{1-\beta^2}{1+\beta^2} \tag{3.6}$$

$$\tau = 0 \tag{3.7}$$

$$\delta = \frac{n(1-\beta^2)}{n-1} \tag{3.8}$$

logo,

$$\frac{vol(E_{k+1})}{vol(E_k)} = n^{1/2}(\frac{n}{n-1})^{(n-1)/2}\beta(1-\beta^2)^{(n-1)/2} \tag{3.9}$$

iii) Para $\alpha + \beta \neq 0$, $\alpha < \beta$ e $\alpha\beta \geq -1/n$, o elipsóide E_{k+1} será mínimo se:

$$\rho = (4(1-\alpha^2)(1-\beta^2) + n^2(\beta^2-\alpha^2)^2)^{1/2} \tag{3.10}$$

$$\sigma = \frac{2(1-\alpha\beta) + n(\alpha+\beta)^2 - \rho}{(n+1)(\alpha+\beta)^2} \tag{3.11}$$

$$\tau = \frac{\sigma(\alpha+\beta)}{2} \tag{3.12}$$

$$\delta = \frac{n^2}{2(n^2-1)}(2 - \alpha^2 - \beta^2 + \rho/n) \tag{3.13}$$

[1] A demonstração completa do cálculo do elipsóide mínimo E_{k+1} pode ser encontrada nos Teoremas 1 e 2 de (Todd 1982).

logo,

$$\frac{vol(E_{k+1})}{vol(E_k)} = ((\frac{1}{n+1})^{1/2}(\frac{n^2}{2(n^2-1)})^{n/2}\frac{(2-\alpha^2-\beta^2+\rho/2)^{n/2}(\rho-2+\alpha^2+\beta^2)^{1/2}}{\alpha+\beta} \tag{3.14}$$

Como pode ser facilmente verificado, o conjunto de equações utilizadas para o cálculo do novo elipsóide no método *Elipsoidal* de Shor, vide (2.11), é o caso particular de corte paralelo considerando-se que $\alpha = 0$ e $\beta = 1$. Analogamente, o conjunto de equações utilizadas para o cálculo do novo elipsóide utilizando-se um *Deep Cut*, vide (2.14), é o caso particular de corte paralelo considerando-se que $\alpha = 0$ e $0 < \beta \leq 1$.

A figura 3.2 mostra um exemplo de cálculo do elipsóide E_{k+1} para $\alpha = 0$ e $\beta = 1/(n+1)$.

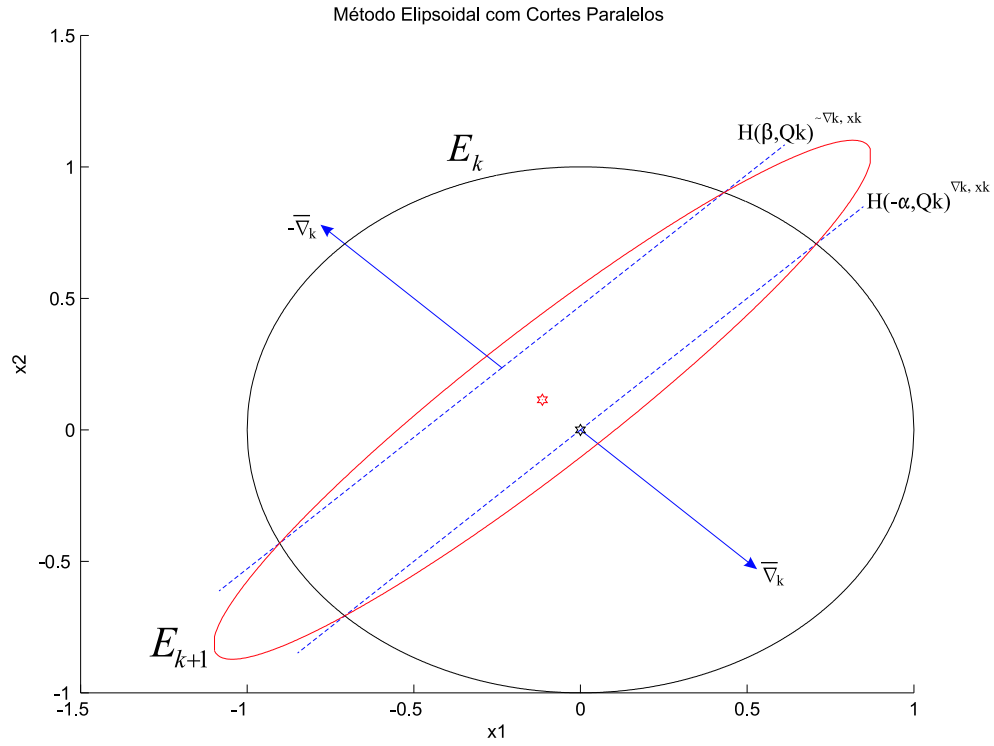


Figura 3.2: Ilustração de uma iteração do método *Elipsoidal* com *Cortes Paralelos*.

3.1.1 Cálculo do Novo Elipsóide para Dois Vetores não Paralelos

O raciocínio desenvolvido para a elaboração de um elipsóide mínimo $\widehat{E}_{k+1}$ da seção 3.1 parte do pressuposto da existência de dois semi-espaços definidos através de um mesmo

vetor $\bar{\nabla}_k$ e dois parâmetros α e β que determinam o afastamento entre eles na métrica do elipsóide atual Q_k.

Extrapolando-se a premissa de que os dois semi-espaços são definidos por um mesmo vetor, teremos como definição dos semi-espaços as equações:

$$H_-(-\alpha, Q_k)^{\bar{\nabla}_k, \mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \bar{\nabla}_k^T(\mathbf{x} - \mathbf{x}_k) \leq -\alpha(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2}\} \tag{3.15}$$

$$H_-(\beta, Q_k)^{-\bar{\nabla}_\omega, \mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid -\bar{\nabla}_\omega^T(\mathbf{x} - \mathbf{x}_k) \leq \beta(\bar{\nabla}_\omega^T Q_k \bar{\nabla}_\omega)^{1/2}\} \tag{3.16}$$

onde -1 $< \frac{\bar{\nabla}_k^T \bar{\nabla}_\omega}{|\bar{\nabla}_k| |\bar{\nabla}_\omega|} < 1$.

Pode-se, então, utilizar (3.17) a (3.23) para determinar um novo elipsóide $E_{k+1} \supset (\, E_k \cap H_-(-\alpha, Q_k)^{\bar{\nabla}_k, \mathbf{x}_k} \cap H_-(\beta, Q_k)^{-\bar{\nabla}_\omega, \mathbf{x}_k}\,)$ que contenha a intersecção dos semi-espaços $H_-(-\alpha, Q_k)^{\bar{\nabla}_k, \mathbf{x}_k}$ e $H_-(\beta, Q_k)^{-\bar{\nabla}_\omega, \mathbf{x}_k}$ com E_k. O volume do elipsóide E_{k+1} poderá ser inferior ao volume do elipsóide $\hat{E}_{k+1}$ que poderia ser obtido com o uso de quaisquer dos dois vetores $\bar{\nabla}_k$ ou $\bar{\nabla}_\omega$, individualmente, empregando-se o conjunto de equações do método *Elipsoidal* de Shor.

As equações, a seguir, definem o cálculo do elipsóide E_{k+1} [2], considerando-se que $\bar{\nabla}_\omega^T Q_k \bar{\nabla}_k \leq 0$.

$$\begin{aligned}
\mathbf{x}_{k+1} &= \mathbf{x}_k + \tau(Q_k \bar{\nabla}_{k,w})/(\bar{\nabla}_{k,w}^T Q_k \bar{\nabla}_{k,w})^{1/2} \\
Q_{k+1} &= \delta[Q_k - \sigma(Q_k \bar{\nabla}_{k,w})(Q_k \bar{\nabla}_{k,w})^T)/(\bar{\nabla}_{k,w}^T Q_k \bar{\nabla}_{k,w})] \\
E_{k+1} &\triangleq \{\mathbf{x} \in \mathbb{R}^n \mid (\mathbf{x} - \mathbf{x}_{k+1})(Q_{k+1})^{-1}(\mathbf{x} - \mathbf{x}_{k+1}) \leq 1\}
\end{aligned} \tag{3.17}$$

onde,

$$\bar{g} = \frac{-\bar{\nabla}_\omega Q_k \bar{\nabla}_k}{(\bar{\nabla}_\omega^T Q_k \bar{\nabla}_\omega)^{1/2}(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2}} \tag{3.18}$$

$$\bar{\beta} = (\frac{1}{n+1})\bar{g}^2 + [1 - (\frac{1}{(n+1)^2})\bar{g}^2]^{1/2}(1 - \bar{g}^2)^{1/2} \tag{3.19}$$

$$\rho = (4(1-\alpha^2)(1-\bar{g}^2) + n^2(\bar{g}^2 - \alpha^2)^2)^{1/2} \tag{3.20}$$

$$\sigma = \frac{2(1 - \alpha\bar{\beta}) + n(\alpha + \bar{\beta})^2 - \rho}{(n+1)(\alpha + \bar{\beta})^2} \tag{3.21}$$

$$\tau = \frac{\sigma(\alpha + \bar{\beta})}{2} \tag{3.22}$$

$$\delta = \frac{n^2}{2(n^2-1)}(2 - \alpha^2 - \bar{\beta}^2 + \rho/n) \tag{3.23}$$

[2] A demonstração completa do cálculo de $\widehat{E}_{k+1}$ pode ser encontrada no Lema 4.1 em (Kim et al. 1994).

logo,

$$\frac{E_{k+1}}{E_k} = ((\frac{1}{n+1})^{1/2}(\frac{n^2}{2(n^2-1)})^{n/2}\frac{(2-\alpha^2-\bar{\beta}^2+\rho/2)^{n/2}(\rho-2+\alpha^2+\bar{\beta}^2)^{1/2}}{\alpha+\bar{\beta}} \tag{3.24}$$

Claramente, deve-se acrescentar as já existentes restrições para os possíveis valores de α e β, que resultam em $vol(E_{k+1}) < vol(\hat{E}_{k+1})$, uma nova restrição quanto ao ângulo máximo que poderá existir entre os dois vetores $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$. Este ângulo máximo também é função da dimensão n. A figura 3.3 mostra a variação da relação $vol(E_{k+1})/vol(\hat{E}_{k+1})$ em função do ângulo $\angle(\bar{\nabla}_k, \bar{\nabla}_\omega)$ e da dimensão n.

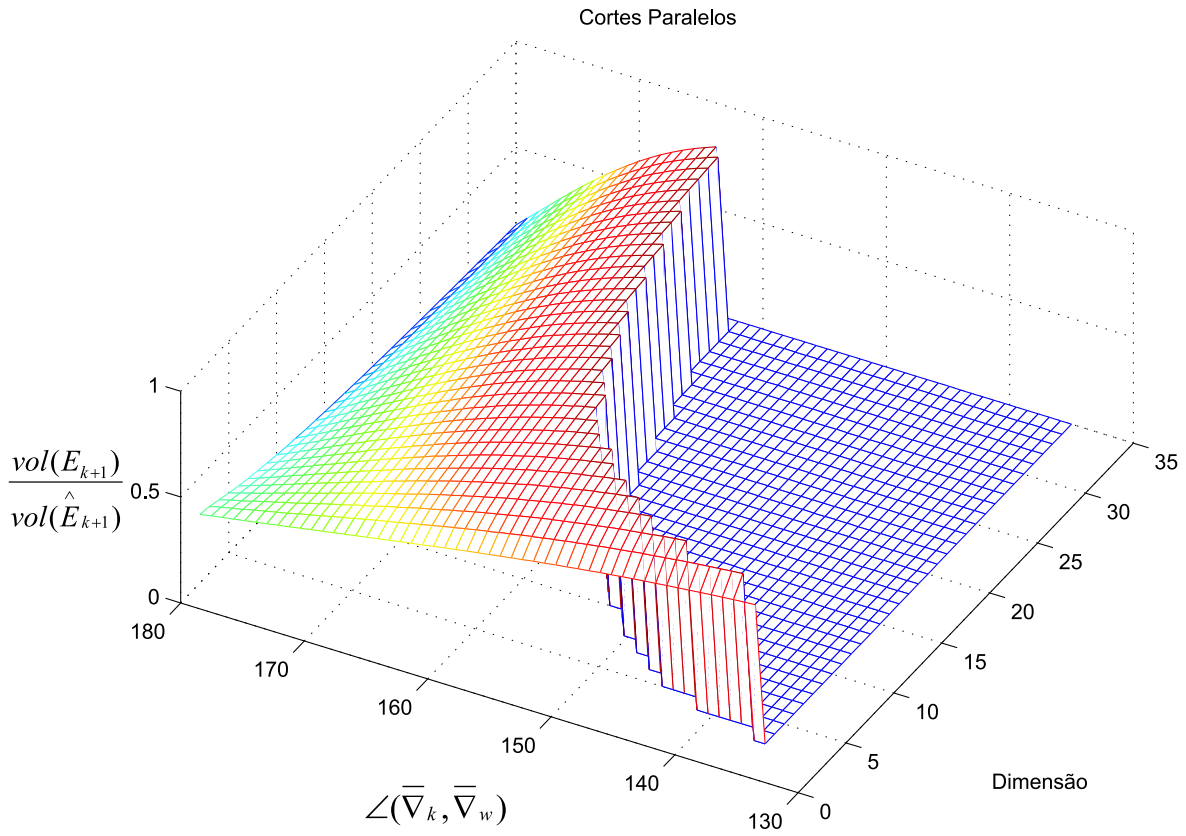
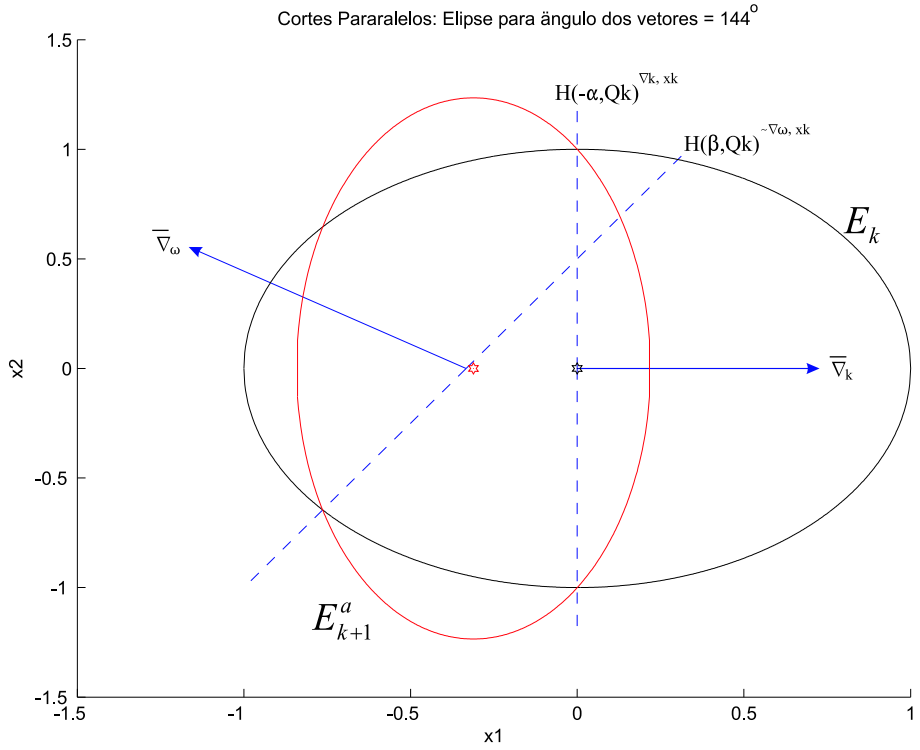


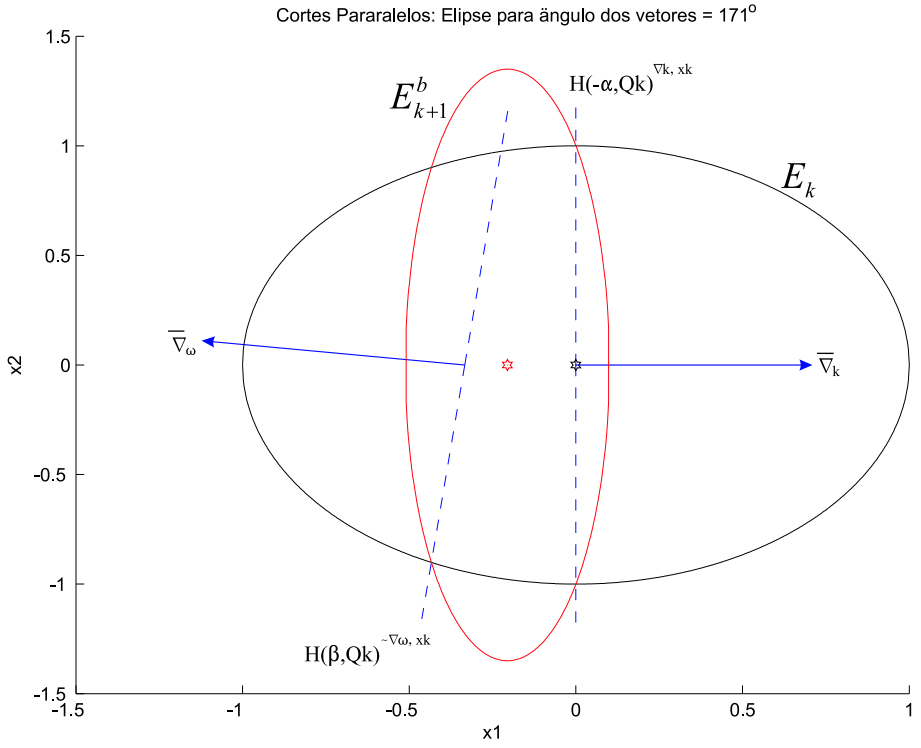
Figura 3.3: Variação do volume gerado por *Cortes Paralelos* em função do $\angle(\bar{\nabla}_k, \bar{\nabla}_\omega)$ e da dimensão n.
Nesta figura, os valores iguais a zero do eixo $\angle(\bar{\nabla}_k, \bar{\nabla}_\omega)$ representam os casos onde $vol(E_{k+1}) \geq vol(\hat{E}_{k+1})$.

A figura 3.4 mostra duas iterações exemplo da aplicação dos *Cortes Paralelos*. No exemplo a é mostrado um par de vetores $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$ com ângulo tal que produz o elipsóide E^a_{k+1}. No exemplo b, para o mesmo elipsóide inicial E_k e o mesmo semi-espaço $H_-(-\alpha, Q_k)^{\bar{\nabla}_k, \mathbf{x}_k}$, é definido um novo semi-espaço $H_-(\beta, Q_k)^{-\bar{\nabla}_\omega, \mathbf{x}_k}$ que resultará no elip-

sóide E^b_{k+1} com volume $vol(E^b_{k+1}) < vol(E^a_{k+1})$. Assim, verifica-se que, para um mesmo valor de β, à medida em que o ângulo entre os vetores $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$ aumenta, o valor de $\bar{\beta}$ também aumenta. Este fato também provoca o aumento do volume do elipsóide calculado.



a) Iteração que resulta no elipsóide E^a_{k+1}.



b) Iteração que resulta no elipsóide E^b_{k+1}.

Figura 3.4: Ilustração de duas iterações do método *Elipsoidal* com *Cortes Paralelos*.

3.2 Corte em Cunha

O algoritmo *Elipsoidal* e suas variantes, que utilizam apenas um vetor $\bar{\nabla}_k$ para a correção do elipsóide E_k de modo a torná-lo E_{k+1}, são denominados *algoritmos de posto 1*, dado que $rank(\bar{\nabla}_k \bar{\nabla}_k^T) = 1$. Tal como já mostrado em (3.1) e pela figura 3.1, estes algoritmos apresentam uma a taxa de redução do volume que é função apenas da dimensão em questão.

Considerando-se a existência de dois semi-espaços $H_-^{\bar{\nabla}_k, \mathbf{x}_k}$ e $H_-^{\bar{\nabla}_\omega, \mathbf{x}_k}$ determinados por dois vetores linearmente independentes $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$, pode-se definir um novo conjunto $H_-^{[\bar{\nabla}_k, \bar{\nabla}_\omega], \mathbf{x}_k} \supset (E_k \cap H_-^{\bar{\nabla}_k, \mathbf{x}_k} \cap H_-^{\bar{\nabla}_\omega, \mathbf{x}_k})$ que conterá a intersecção destes semi-espaços com o elipsóide atual E_k, tal como indicado nas equações abaixo. O conjunto $H_-^{[\bar{\nabla}_k, \bar{\nabla}_\omega], \mathbf{x}_k}$ define o chamado *Corte em Cunha* (Ech-Cherif & Ecker 1984).

$$H_-^{\bar{\nabla}_k, \mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \bar{\nabla}_k^T (\mathbf{x} - \mathbf{x}_k) \leq 0\} \tag{3.25}$$

$$H_-^{\bar{\nabla}_\omega, \mathbf{x}_k} \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \bar{\nabla}_\omega^T (\mathbf{x} - \mathbf{x}_k) \leq 0\} \tag{3.26}$$

$$H_-^{[\bar{\nabla}_k, \bar{\nabla}_\omega], \mathbf{x}_k} \triangleq \{\mathbf{x} \in E_k \mid \bar{\nabla}_k^T (\mathbf{x} - \mathbf{x}_k) \leq 0 \quad e \quad \bar{\nabla}_\omega^T (\mathbf{x} - \mathbf{x}_k) \leq 0\} \tag{3.27}$$

A utilização simultânea dos vetores $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$ permite o cálculo do elipsóide de volume mínimo $E_{k+1} \supset H_-^{[\bar{\nabla}_k, \bar{\nabla}_\omega], \mathbf{x}_k}$ [3] com a correção através de uma matriz $(c_1 \bar{\nabla}_k \bar{\nabla}_k^T + c_2 \bar{\nabla}_\omega \bar{\nabla}_\omega^T)$, onde c_1 e c_2 são constantes, com posto igual a dois, o que justifica o nome destes métodos como *algoritmos de posto 2*.

As equações abaixo demonstram o cálculo do elipsóide de volume mínimo E_{k+1} para dimensão $n > 2$:

$$\begin{aligned}
\mathbf{a}_p &= \bar{\nabla}_k / (\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2} + \bar{\nabla}_\omega / (\bar{\nabla}_\omega^T Q_k \bar{\nabla}_\omega)^{1/2} \\
\mathbf{a}_m &= \bar{\nabla}_k / (\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2} - \bar{\nabla}_\omega / (\bar{\nabla}_\omega^T Q_k \bar{\nabla}_\omega)^{1/2} \\
\mathbf{x}_{k+1} &= \mathbf{x}_k + z_0^* Q_k \mathbf{a}_p \qquad (3.28) \\
Q_{k+1} &= \delta^* Q_k + (\gamma^* - \delta^*) Q_k \mathbf{a}_p \mathbf{a}_p^T Q_k / (\mathbf{a}_p^T Q_k \mathbf{a}_p) \\
&\quad + (\omega^* - \delta^*) Q_k \mathbf{a}_m \mathbf{a}_m^T Q_k / (\mathbf{a}_m^T Q_k \mathbf{a}_m) \qquad (3.29) \\
E_{k+1} &\triangleq \{\mathbf{x} \in \mathbb{R}^n \mid (\mathbf{x} - \mathbf{x}_{k+1})(Q_{k+1})^{-1} (\mathbf{x} - \mathbf{x}_{k+1}) \leq 1\}
\end{aligned}$$

onde, definindo-se $q = \frac{\|\bar{\nabla}_k - \bar{\nabla}_\omega\|}{2}$, e $q^* = \frac{-1 + ((n^2-n)/2)^{1/2}}{n-2}$ temos:

[3] A demonstração completa do cálculo de $\widehat{E}_{k+1}$ pode ser encontrada em (Ech-Cherif & Ecker 1984).

i) Para $q < q^*$

$$z_0^* = \frac{((n+2)q + (n+3) - ((n+2)^2q^2 + 2(n^2-3n-2)q + (n-1)^2)^{1/2})}{4(n+1)} \quad (3.30)$$

$$\gamma^* = (1 - z_0^*)^2 \quad (3.31)$$

$$\delta^* = \gamma^*/(\gamma^* - (z_0^*)^2) \quad (3.32)$$

$$\omega^* = \frac{(1-q^2) * \gamma^*}{(\gamma^* - (q - z_0^*)^2)} \quad (3.33)$$

ii) Para $q \geq q^*$

$$z_0^* = \frac{2q}{n+1} \quad (3.34)$$

$$\gamma^* = \frac{2n(n-1)q^2}{(n+1)^2} \quad (3.35)$$

$$\delta^* = \frac{n(n-1)}{(n+1)(n-2)} \quad (3.36)$$

$$\omega^* = \frac{2n(1-q^2)}{n+1} \quad (3.37)$$

Deve-se observar que embora não esteja mencionado em (Ech-Cherif & Ecker 1984), as equações acima podem ser empregadas para a dimensão $n = 2$ utilizando-se $q^* = \sqrt{3}/2$ e $\delta^* = 0$, uma vez que não existem vetores fora do plano definido por $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$. Outro ponto em questão é que a utilização de (3.17) a (3.23) para valores de $\alpha = \beta = 0$ não produz um elipsóide de volume mínimo que contém o conjunto $H_-^{[\bar{\nabla}_k, \bar{\nabla}_\omega], \mathbf{x}_k}$, diferentemente do que ocorre em (3.28) a (3.37). Por outro lado, (3.28) a (3.37) somente pode ser utilizado para $\alpha = \beta = 0$.

Demonstra-se que,

$$\frac{vol(E_{k+1})}{vol(E_k)} = (\gamma^*(\delta^*)^{(n-2)}\omega^*)^{1/2} \quad (3.38)$$

Considere $\hat{E}_{k+1}$ um elipsóide correspondente ao que poderia ser obtido com o uso de quaisquer dos dois vetores $\bar{\nabla}_k$ ou $\bar{\nabla}_\omega$, individualmente, empregando-se (2.11) referente ao método *Elipsoidal* de Shor. Deve-se observar que, embora a desigualdade $vol(E_{k+1}) < vol(\hat{E}_{k+1})$ seja sempre mantida, a razão entre os dois volumes será função da dimensão n e do ângulo entre os dois vetores $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$. A figura 3.5 mostra a variação da relação $vol(E_{k+1})/vol(\hat{E}_{k+1})$ em função do ângulo $\angle(\bar{\nabla}_k, \bar{\nabla}_\omega)$ e da dimensão n.

A figura 3.6 mostra duas iterações como exemplo da aplicação dos *Cortes em Cunha*. No exemplo da subfigura 3.6.a é mostrado um par de vetores $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$ com ângulo que

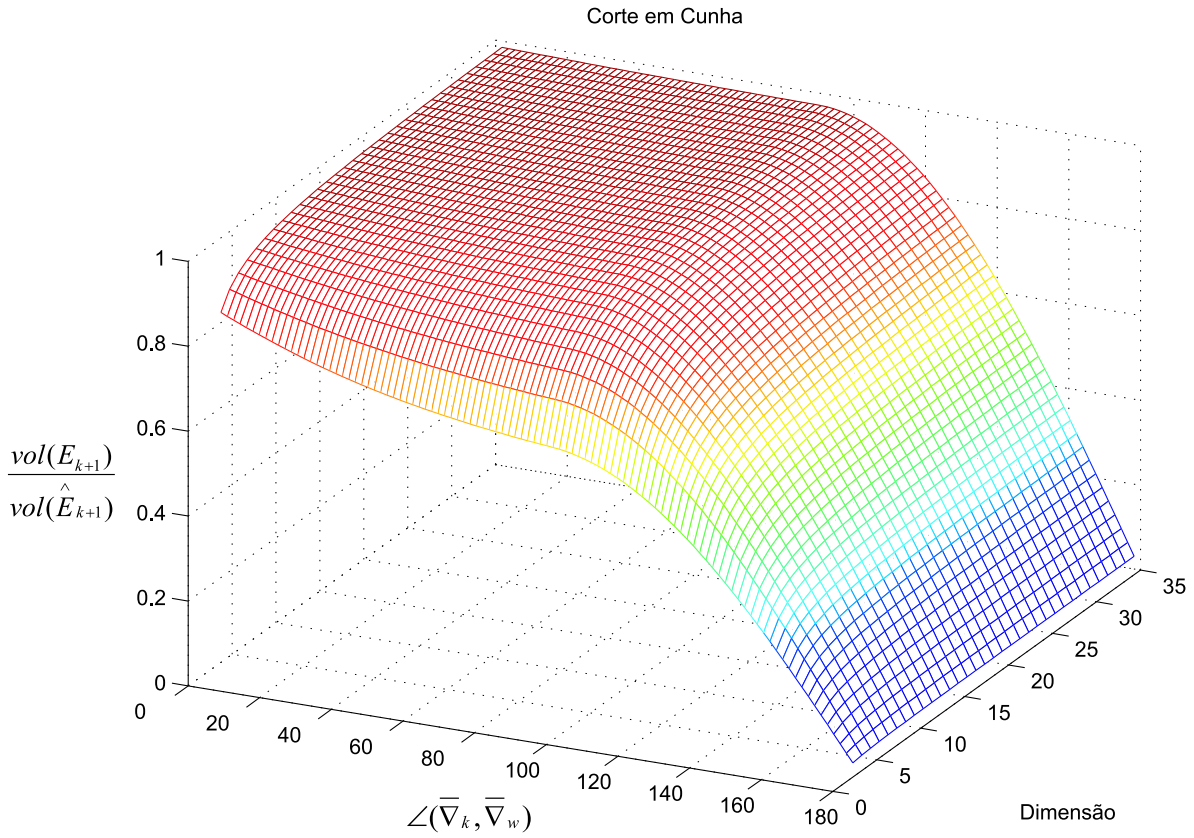
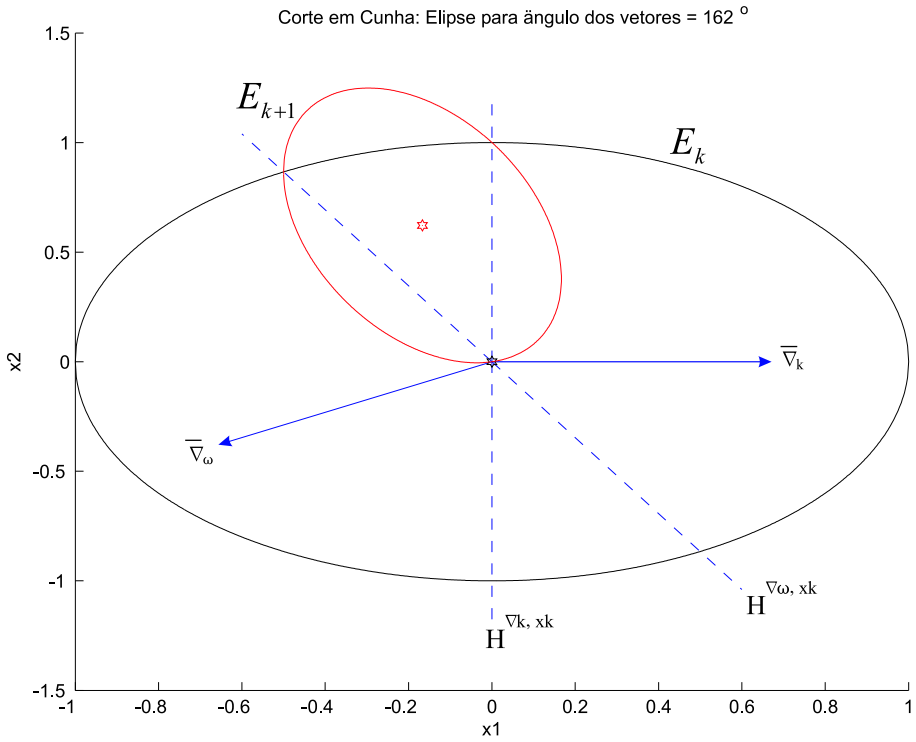
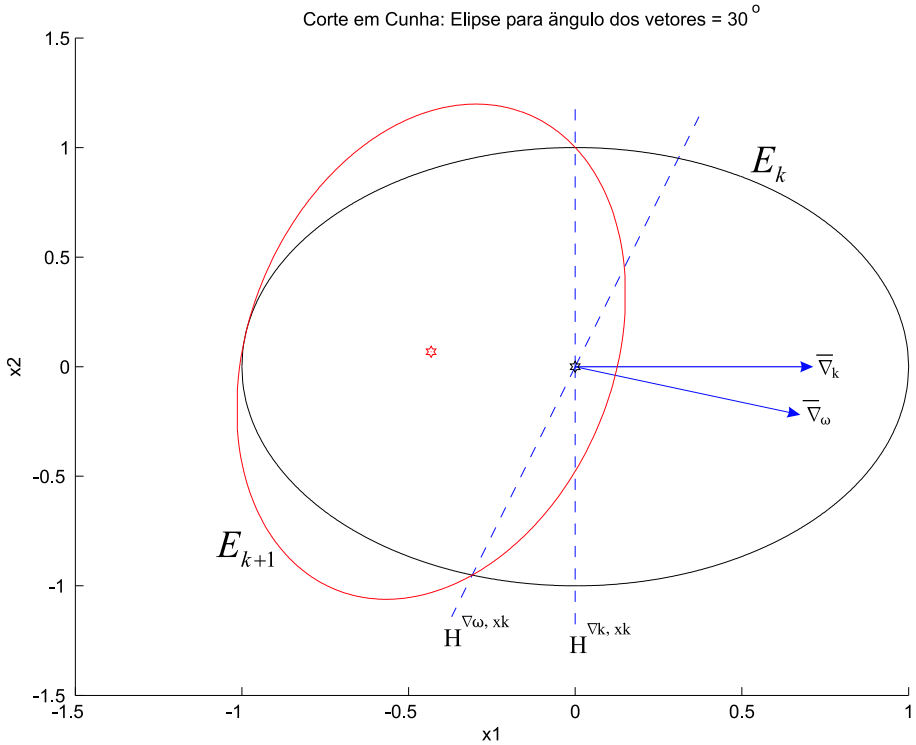


Figura 3.5: Variação do volume gerado por *Cortes em Cunha* em função do $\angle(\bar{\nabla}_k, \bar{\nabla}_\omega)$ e da dimensão n.

produzirá um valor de q menor que o valor crítico q^*. Portanto são utilizadas (3.30) a (3.33) para o cálculo de E_{k+1}. No exemplo da subfigura 3.6.b, para o mesmo elipsóide inicial E_k e o mesmo semi-espaço $H_-^{\bar{\nabla}_k, \mathbf{x}_k}$ é definido um novo semi-espaço $H_-^{\bar{\nabla}_\omega, \mathbf{x}_k}$, tal que $q > q^*$. Assim, são utilizadas (3.34) a (3.37) para o cálculo de E_{k+1}.



a) Iteração com $q < q^*$.



b) Iteração com $q > q^*$.

Figura 3.6: Ilustração de duas iterações do método *Elipsoidal* com *Cortes em Cunha*.

3.3 Algoritmo dos Dois Subgradientes Sucessivos

A utilização dos possíveis cortes, quer sejam *Deep Cuts* de Shor e Gershovich, *Cortes Paralelos* de Todd ou *Cortes em Cunha* de Ech-Cherif e Ecker, para a implementação de variantes do método *Elipsoidal* é feita independentemente. Todavia, uma vez que no transcorrer da convergência para a solução ótima podem surgir ocorrências diversas dos três tipos de corte supracitados, um algoritmo capaz de identificar, dentre os tipos de corte, aquele que propiciará a maior redução possível para a contração do volume do novo elipsóide mínimo E_{k+1}, deve também resultar em menor esforço ECT, quando comparada à implementação individual dos cortes.

O algoritmo *Dois Subgradientes Sucessivos* (Kim et al. 1994) utiliza a avaliação do gradiente em dois pontos consecutivos da seqüência de convergência $\mathbf{x}_k$ e $\bar{\mathbf{x}}_{k+1}$. Os subgradientes $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$, calculados nos respectivos pontos, são utilizados para avaliar a possibilidade de se construir um corte tipo *Deep Cut* ou *Paralelo* ou *Cunha* que gerará o elipsóide E_{k+1}. Tendo por referência o elipsóide $\hat{E}_{k+2}$ [4], pretende-se então obter um elipsóide E_{k+1} de volume inferior ao do elipsóide $\hat{E}_{k+2}$.

O problema (2.8) será aqui tratado e é representado abaixo para a comodidade do leitor:

$$min\ f(\mathbf{x}) \tag{3.39}$$

$$s.a.\ \mathbf{x} \in \Omega$$

$$\Omega \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid g_j(\mathbf{x}) \leq 0,\ \forall\, j = 1, \ldots, p\}$$

A estruturação geral para o algoritmo *Dois Subgradientes Sucessivos* é apresentada a seguir. As correções de erros existentes em (Kim et al. 1994) e incluídas no algoritmo a seguir representam a primeira contribuição desta Tese:

[4] Elipsóide correspondente ao que poderia ser obtido com a utilização em seqüência dos dois subgradientes $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$, empregando-se (2.11) referente ao método *Elipsoidal* de Shor.

Algorithm 4 *Dois Subgradientes Sucessivos*

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^*$ será procurada de acordo com o problema definido por (3.39). Um erro $\epsilon > 0$.

Output: A melhor solução factível contínua $\mathbf{x}^f$ encontrada. Para um problema infactível ou para $\mathbf{x}^* \notin E_0$ ter-se-á $\mathbf{x}^f = \emptyset$.

function *Dois Subgradientes Sucessivos*(E_0, ϵ)

Passo 0: ▷ Cálculo dos parâmetros.

$\tau_0 \leftarrow \frac{1}{n+1}$
$\sigma_0 \leftarrow \frac{2}{n+1}$
$\delta_0 \leftarrow \frac{n^2}{n^2-1}$
$q^* \leftarrow \frac{-1+((n^2-n)/2)^{1/2}}{n-2}$
$\alpha_0 \leftarrow \frac{1}{n+1}$.

▷Definição de funções.

$$s(\cdot) = \begin{cases} g_j(\cdot) & \text{para algum } g_j(\mathbf{x}) > 0,\ j = 1, \ldots, p. \\ f(\cdot) & \text{caso contrário.} \end{cases}$$

Passo 1: ▷ Cálculo de $\bar{\nabla}_k$ e de um ponto temporário $\bar{\mathbf{x}}_{k+1}$.

Passo 1a: $\bar{\nabla}_k \leftarrow \bar{\nabla} s(\mathbf{x}_k)$.
if $\bar{\nabla}_k = 0$ **then return** $\mathbf{x}_k$.
$d_k \leftarrow 1/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2}$
if $(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2} \leq 0$ **then return** $\mathbf{x}_k$ ▷ Perda positividade de Q_k.

Passo 1b: $\bar{\mathbf{x}}_{k+1} \leftarrow \mathbf{x}_k - \tau_0 d_k Q_k \bar{\nabla}_k$

Passo 2: ▷ Construção de elipsóide E_{k+1}.

$\bar{\nabla}_\omega \leftarrow \bar{\nabla} s(\mathbf{x}_{k+1})$.
if $\bar{\nabla}_\omega = 0$ **then return** $\mathbf{x}_k$.
$\bar{d}_{k+1} \leftarrow 1/(\bar{\nabla}_\omega^T Q_k \bar{\nabla}_\omega)^{1/2}$
if $(\bar{\nabla}_\omega^T Q_k \bar{\nabla}_\omega)^{1/2} \leq 0$ **then return** $\mathbf{x}_k$ ▷ Perda positividade de Q_k.
if $\tau_0 d_k (\bar{\nabla}_\omega^T Q_k \bar{\nabla}_k) \geq 0$ **then go to** passo 2c.

Passo 2a: $\bar{g} \leftarrow -d_k \bar{d}_{k+1} \bar{\nabla}_\omega^T Q_k \bar{\nabla}_k$
$\bar{\beta} \leftarrow (\frac{1}{n+1})\bar{g}^2 + [1 - (\frac{1}{(n+1)^2})\bar{g}^2]^{1/2}(1 - \bar{g}^2)^{1/2}$
if $Rp(0, \bar{\beta}, n) > Rt(n)^2$ **then go to** passo 2e.

Passo 2b: ▷ *Cortes Paralelos*, (3.5) a (3.14).

$\alpha \leftarrow 0$

$\rho \leftarrow (4(1-\alpha^2)(1-\bar{g}^2) + n^2(\bar{g}^2-\alpha^2)^2)^{1/2}$

$\sigma \leftarrow \frac{2(1-\alpha\bar{\beta})+n(\alpha+\bar{\beta})^2-\rho}{(n+1)(\alpha+\bar{\beta})^2}$

$\tau \leftarrow \frac{\sigma(\alpha+\bar{\beta})}{2}$

$\delta \leftarrow \frac{n^2}{2(n^2-1)}(2-\alpha^2-\bar{\beta}^2+\rho/n)$

$\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \tau(Q_k\bar{\nabla}_k)/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2}$

$Q_{k+1} \leftarrow \delta[Q_k - \sigma(Q_k\bar{\nabla}_k)(Q_k\bar{\nabla}_k)^T)/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)]$

go to passo 3.

Passo 2c: ▷ *Deep Cut*, (2.14).

$\alpha_k \leftarrow \tau_0 d_k \bar{d}_{k+1}(\bar{\nabla}_\omega^T Q_k \bar{\nabla}_k)$.

if $\alpha_k < \alpha_0$ **then go to** passo 2d.

$\alpha \leftarrow \alpha_k$

$\tau \leftarrow \frac{1+n\alpha}{n+1}$

$\sigma \leftarrow \frac{2(1+n\alpha)}{(n+1)(1+\alpha)}$

$\delta \leftarrow (\frac{n^2}{n^2-1})(1-\alpha^2)$

$\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k - \tau(Q_k\mathbf{g}_k/(\mathbf{g}_k^T Q_k \mathbf{g}_k)^{1/2})$

$Q_{k+1} \leftarrow \sigma[Q_k - \delta(Q_k\mathbf{g}_k)(Q_k\mathbf{g}_k)^T/(\mathbf{g}_k^T Q_k \mathbf{g}_k)]$

go to passo 3.

Passo 2d: $\bar{q} \leftarrow ((1-(n+1)\alpha_k)/2)^{1/2}$.

if $\bar{q} < q^*$ **then go to** passo 2e.

▷ *Corte em Cunha*, (3.28) a (3.37).

$\mathbf{a}_p \leftarrow \bar{\nabla}_k/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2} + \widehat{g}_k/(\widehat{g}_k^T Q_k \widehat{g}_k)^{1/2}$

$\mathbf{a}_m \leftarrow \bar{\nabla}_k/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2} - \widehat{g}_k/(\widehat{g}_k^T Q_k \widehat{g}_k)^{1/2}$

$z_0^* \leftarrow \frac{2q}{n+1}$

$\gamma^* \leftarrow \frac{2n(n-1)q^2}{(n+1)^2}$

$\delta^* \leftarrow \frac{n(n-1)}{(n+1)(n-2)}$

$\omega^* \leftarrow \frac{2n(1-q^2)}{n+1}$

$\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + z_0^* Q_k \mathbf{a}_p$

$Q_{k+1} \leftarrow \delta^* Q_k + (\gamma^*-\delta^*)Q_k\mathbf{a}_p\mathbf{a}_p^T Q_k/(\mathbf{a}_p^T Q_k \mathbf{a}_p)$
$\quad +(\omega^*-\delta^*)Q_k\mathbf{a}_m\mathbf{a}_m^T Q_k/(\mathbf{a}_m^T Q_k \mathbf{a}_m)$

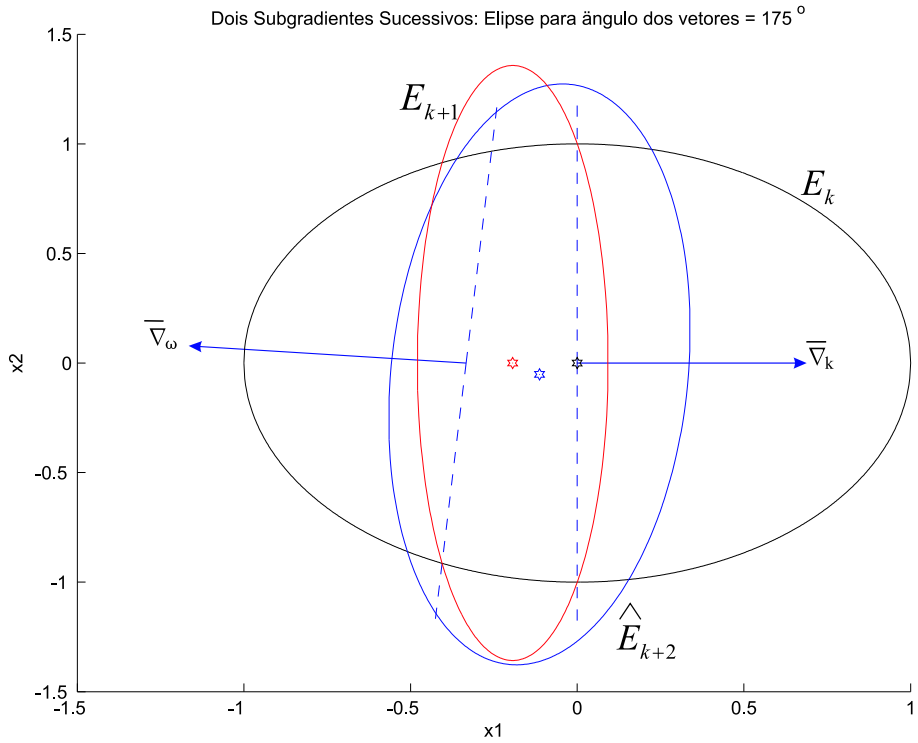
go to passo 3.

Passo 2e: $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_{k+1}$
$g_{k+1} \leftarrow \bar{\nabla}_\omega$.
$\triangleright$ *Shor*, por (2.11).
$Q_{k+1}g_{k+1} \leftarrow \delta_0(Q_k\mathbf{g}_{k+1} - \sigma_0 d_k(Q_k\mathbf{g}_k)(Q_k\mathbf{g}_k)^T\mathbf{g}_{k+1}$
$\mathbf{x}_{k+2} \leftarrow \mathbf{x}_{k+1} - \tau_0(Q_{k+1}\mathbf{g}_{k+1})/(\mathbf{g}_{k+1}^T(Q_{k+1}\mathbf{g}_{k+1}))^{1/2}$
$Q_{k+2} \leftarrow \sigma_0[Q_k - \delta_0(Q_{k+1}\mathbf{g}_{k+1})(Q_{k+1}\mathbf{g}_{k+1})^T/(\mathbf{g}_{k+1}^T(Q_{k+1}\mathbf{g}_{k+1}))]$
$k \leftarrow k+1$

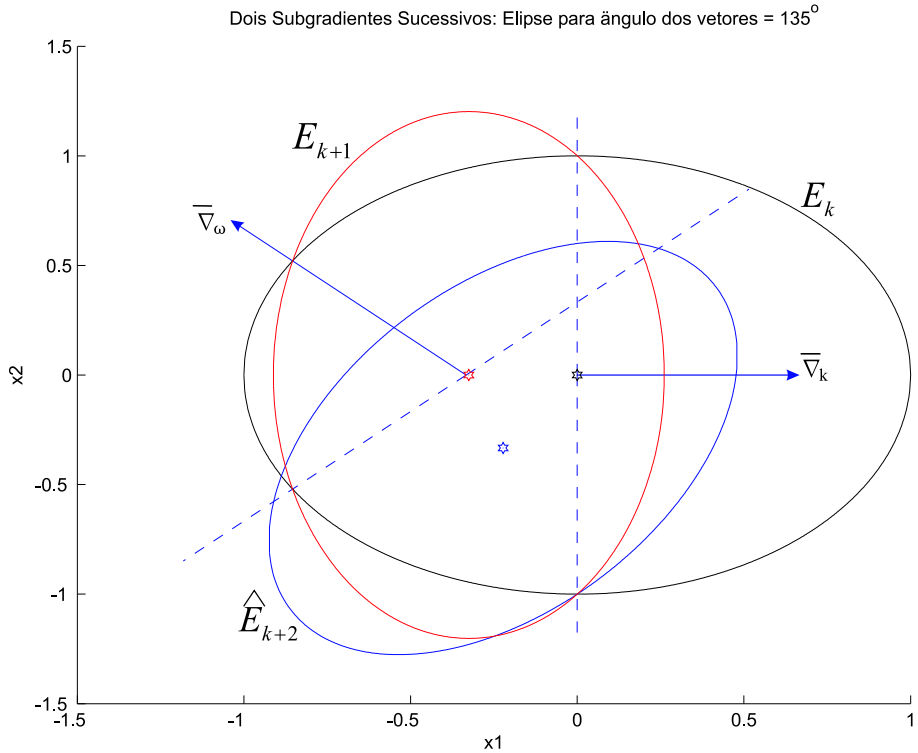
Passo 3: **if** vol $(E_{k+1}) \simeq 0$ ou $vol(E_{k+1}) \simeq vol(E_k)$ **then return** $\mathbf{x}_{k+2}$
$\bar{\nabla}_k \leftarrow \bar{\nabla}_\omega$
$d_k = \bar{d}_{k+1}$
$k \leftarrow k+1$
go to passo 1b.

Como pode ser verificado no algoritmo 4, o objetivo do método *Dois Subgradientes Sucessivos* é verificar se uma seqüência de construção de dois elipsóides consecutivos pelo método de Shor pode ser substituída por uma única iteração do algoritmo *Deep Cut*, utilizando o vetor $\bar{\nabla}_\omega$, ou do algoritmo de *Cortes Paralelos* ou do algoritmo de *Cortes em Cunha*.

A figura 3.7 mostra duas iterações exemplo da aplicação do algoritmo *Dois Subgradientes Sucessivos*. Nesta figura, $\hat{E}_{k+2}$ representa o elipsóide obtido pela seqüência de dois cálculos consecutivos pelo método de Shor aplicadas ao elipsóide E_k. O primeiro cálculo obtém o elipsóide $\hat{E}_{k+1}$, não mostrado nas subfiguras, a partir de E_k e $\bar{\nabla}_k$. O segundo cálculo, obtém o elipsóide $\hat{E}_{k+2}$ a partir de E_{k+1} e $\bar{\nabla}_\omega$. Já o elipsóide E_{k+1} é obtido pela execução do passo 2b, referente à utilização dos dois vetores $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$ para a aplicação dos *Cortes Paralelos*. Na subfigura 3.7.a é mostrado um par de vetores $\bar{\nabla}_k$ ou $\bar{\nabla}_\omega$ com ângulo tal que produz um elipsóide E_{k+1} com volume inferior a $\hat{E}_{k+2}$. No segundo exemplo, o par de vetores $\bar{\nabla}_k$ e $\bar{\nabla}_\omega$ possui um ângulo que produzirá um elipsóide E_{k+1} com volume superior $\hat{E}_{k+2}$.



a) Iteração com $E_{k+1} < \hat{E}_{k+2}$.



b) Iteração com $E_{k+1} > \hat{E}_{k+2}$.

Figura 3.7: Ilustração de duas iterações do método *Dois Subgradientes Sucessivos*.

3.3.1 Proposição de uma Variante do Algoritmo dos Dois Subgradientes Sucessivos Quando Existem Duas Restrições Violadas

Como pode ser observado pela estruturação do algoritmo *Dois Subgradientes Sucessivos*, embora este utilize a formulação dos *Cortes em Cunha*, este corte não é aplicado para o caso em que existam duas restrições violadas $g_\lambda(\cdot)$ e $g_\mu(\cdot) \mid \lambda \; e \; \mu \in [1,p]$, apesar desta sugestão já existir em (Ech-Cherif & Ecker 1984). Esta alteração do algoritmo, certamente acrescentará mais uma possibilidade de redução do volume do novo elipsóide e pode ser facilmente implementada através da inclusão do passo $1a^i$ com o cálculo do novo elipsóide através de (3.28) a (3.37).

As alterações para a estruturação geral para a variante do algoritmo *Dois Subgradientes Sucessivos*, quando da existência de duas restrições violadas, são apresentadas a seguir e constituem o primeiro algoritmo proposto por esta Tese. A função $\bar{\nabla} max\angle(\mathbf{g}(\mathbf{x}))$ define as duas restrições violadas que produzirão o menor elipsóide E_{k+1} a partir de (3.28) a (3.37)

Algorithm 5 *Dois Subgradientes Sucessivos 2.R.A.*

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^*$ será procurada de acordo com o problema definido por (3.39). Um erro $\epsilon > 0$.

Output: A melhor solução factível contínua $\mathbf{x}^f$ encontrada. Para um problema infactível ou para $\mathbf{x}^* \notin E_0$ ter-se-á $\mathbf{x}^f = \emptyset$.

function *Dois Subgradientes Sucessivos 2.R.A.* (E_0, ϵ)

Passo 0: ▷ Cálculo dos parâmetros.

$\tau_0 \leftarrow \frac{1}{n+1}$
$\sigma_0 \leftarrow \frac{2}{n+1}$
$\delta_0 \leftarrow \frac{n^2}{n^2-1}$
$q^* \leftarrow \frac{-1+((n^2-n)/2)^{1/2}}{n-2}$
$\alpha_0 \leftarrow \frac{1}{n+1}$.

▷Definição de funções.

$Rc(\gamma, \delta, \omega) = (\gamma(\delta)^{(n-2)}\omega)^{1/2}$
$Ind(\mathbf{g}(\mathbf{x})) = \{j \mid g_j(\mathbf{x}) > 0 \, \forall \, j = 1, \ldots, p\}$

$$\bar{\nabla} max\angle(\mathbf{g}(\mathbf{x})) = \begin{cases} [\bar{\nabla} g_i(\mathbf{x}), \bar{\nabla} g_i(\mathbf{x})] \mid Ind(\mathbf{x}) = [i], \\ [\bar{\nabla} g_i(\mathbf{x}), \bar{\nabla} g_j(\mathbf{x})] \mid \frac{\bar{\nabla} g_i(\mathbf{x})^T \bar{\nabla} g_j(\mathbf{x})}{\|\bar{\nabla} g_i(\mathbf{x})\| \, \|\bar{\nabla} g_j(\mathbf{x})\|} \leq \frac{\bar{\nabla} g_\ell(\mathbf{x})^T \bar{\nabla} g_\wp(\mathbf{x})}{\|bar \nabla g_\ell(\mathbf{x})\| \, \|\bar{\nabla} g_\wp(\mathbf{x})\|} \\ \qquad\qquad \mid i \neq j, \; \forall \; \ell \in Ind(\mathbf{r}) \;\; \text{e} \;\; \forall \; \wp \in Ind(\mathbf{r}). \end{cases}$$

$$s(\cdot) = \begin{cases} g_j(\cdot) & \text{para algum } g_j(\mathbf{x}) > 0, \; j = 1, \ldots, p. \\ f(\cdot) & \text{caso contrário.} \end{cases}$$

$$\bar{\nabla} s2(\mathbf{x}) = \begin{cases} [\bar{\nabla} f(\mathbf{x}), \bar{\nabla} f(\mathbf{x})] & \text{se } Ind(\mathbf{r}) = \emptyset, \\ \bar{\nabla} max\angle(\mathbf{r}) & \text{se } Ind(\mathbf{r}) \neq \emptyset. \end{cases}$$

Passo 1: ▷ Cálculo de $\bar{\nabla}_k$ e de um ponto temporário $\bar{\mathbf{x}}_{k+1}$.

Passo 1a: $[\bar{\nabla}_k^i, \bar{\nabla}_k^j] \leftarrow \bar{\nabla} s2(\mathbf{x}_k)$.

if $\bar{\nabla}_k^i = \bar{\nabla}_k^j = 0$ **then return** $\mathbf{x}_k$.

if $\bar{\nabla}_k^i \neq \bar{\nabla}_k^j$ **then go to** passo 1a^i.

$\bar{\nabla}_k \leftarrow \bar{\nabla}_k^i$

$d_k \leftarrow 1/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2}$

if $(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2} \leq 0$ **then return** $\mathbf{x}_k$ ▷ Perda positividade de Q_k.

go to passo 1b.

Passo 1a^i: ▷ *Corte em Cunha* para duas restrições violadas, por (3.28) a (3.37).

$\bar{\nabla}_k \leftarrow \bar{\nabla}_k^i$

$\bar{\nabla}_\omega \leftarrow \bar{\nabla}_k^j$

$q \leftarrow \frac{\|\bar{\nabla}_k - \bar{\nabla}_\omega\|}{2}$

$\mathbf{a}_p \leftarrow \bar{\nabla}_k/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2} + \bar{\nabla}_\omega/(\bar{\nabla}_\omega^T Q_k \bar{\nabla}_\omega)^{1/2}$

$\mathbf{a}_m \leftarrow \bar{\nabla}_k/(\bar{\nabla}_k^T Q_k \bar{\nabla}_k)^{1/2} - \bar{\nabla}_\omega/(\bar{\nabla}_\omega^T Q_k \bar{\nabla}_\omega)^{1/2}$

if $q < q^*$ **then**

$z_0^* \leftarrow \frac{((n+2)q+(n+3)-((n+2)^2q^2+2(n^2-3n-2)q+(n-1)^2)^{1/2})}{4(n+1)}$

$\gamma^* \leftarrow (1 - z_0^*)^2$

$\delta^* \leftarrow \gamma^*/(\gamma^* - (z_0^*)^2)$

$\omega^* \leftarrow \frac{(1-q^2)*\gamma^*}{(\gamma^* - (q - z_0^*)^2)}$

else

$z_0^* \leftarrow \frac{2q}{n+1}$

$\gamma^* \leftarrow \frac{2n(n-1)q^2}{(n+1)^2}$

$\delta^* \leftarrow \frac{n(n-1)}{(n+1)(n-2)}$

$\omega^* \leftarrow \frac{2n(1-q^2)}{n+1}$

endif

if $Rc(\gamma^*, \delta^*, \omega^*) \geq Rt(n)^2$ **then**

$\bar{\nabla}_k \leftarrow (\bar{\nabla}_k^i / \parallel \bar{\nabla}_k^i \parallel) + (\bar{\nabla}_k^j / \parallel \bar{\nabla}_k^j \parallel)$

go to passo 1b.

endif

$\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + z_0^* Q_k \mathbf{a}_p$
$Q_{k+1} \leftarrow \delta^* Q_k + (\gamma^* - \delta^*) Q_k \mathbf{a}_p \mathbf{a}_p^T Q_k / (\mathbf{a}_p^T Q_k \mathbf{a}_p)$
$\quad + (\omega^* - \delta^*) Q_k \mathbf{a}_m \mathbf{a}_m^T Q_k / (\mathbf{a}_m^T Q_k \mathbf{a}_m)$
$k \leftarrow k + 1$
go to passo 3.

Passo 1b: Tal como mostrado para o algoritmo 4
Passo 2: Tal como mostrado para o algoritmo 4
Passo 3: **if** vol $(E_{k+1}) \simeq 0$ ou $vol(E_{k+1}) \simeq vol(E_k)$ **then return** $\mathbf{x}_{k+2}$
$k \leftarrow k + 1$
go to passo 1a.

A figura 3.8 exibe uma iteração exemplo da aplicação da proposição da *Variante do Algoritmo dos Dois Subgradientes Sucessivos 2.R.A.*. Nesta figura, o elipsóide E_{k+1} representa o elipsóide obtido pela execução do passo 2b, referente à utilização dos dois vetores $\bar{\nabla}_k$ ou $\bar{\nabla}_\omega$ para a aplicação dos *Cortes Paralelos*. Já o elipsóide $\bar{E}_{k+1}$ representa o elipsóide obtido pela execução do novo passo $1a^i$ que utiliza duas restrições violadas no cálculo do novo elipsóide pela aplicação do *Corte em Cunha*. Como pode ser observado a variação aqui proposta do algoritmo *Dois Subgradientes Sucessivos* é capaz de produzir um elipsóide de volume inferior ao volume do elipsóide que seria obtido pelo algoritmo proposto originalmente em (Kim et al. 1994).

Uma vez considerada a possibilidade de se utilizar, caso existam, as informações de mais de um subgradiente nos pontos $\mathbf{x}_k$ e $\bar{\mathbf{x}}_{k+1}$, abre-se um leque de variantes para o algoritmo *Dois Subgradientes Sucessivos* utilizando-se a pesquisa dos possíveis cortes obtidos com as diversas combinações dos gradientes existentes em $\mathbf{x}_k$ e $\bar{\mathbf{x}}_{k+1}$.

3.4 Método *Cone-Elipsoidal*

Proposto inicialmente por Dias (Dias 2003) para problemas vetoriais convexos diferenciáveis, o *Método Cone-Elipsoidal*, MCE, é baseado apenas nas definições do *Cone de Direções Otimizantes* e *Cone de Direções Factibilizantes* para implementar uma variação do *Algoritmo Elipsoidal* de Shor na qual a direção do corte do elipsóide atual E_k irá produzir um novo elipsóide E_{k+1} cujo centro $\mathbf{x}_{k+1}$ necessariamente estará no interior do cone utilizado.

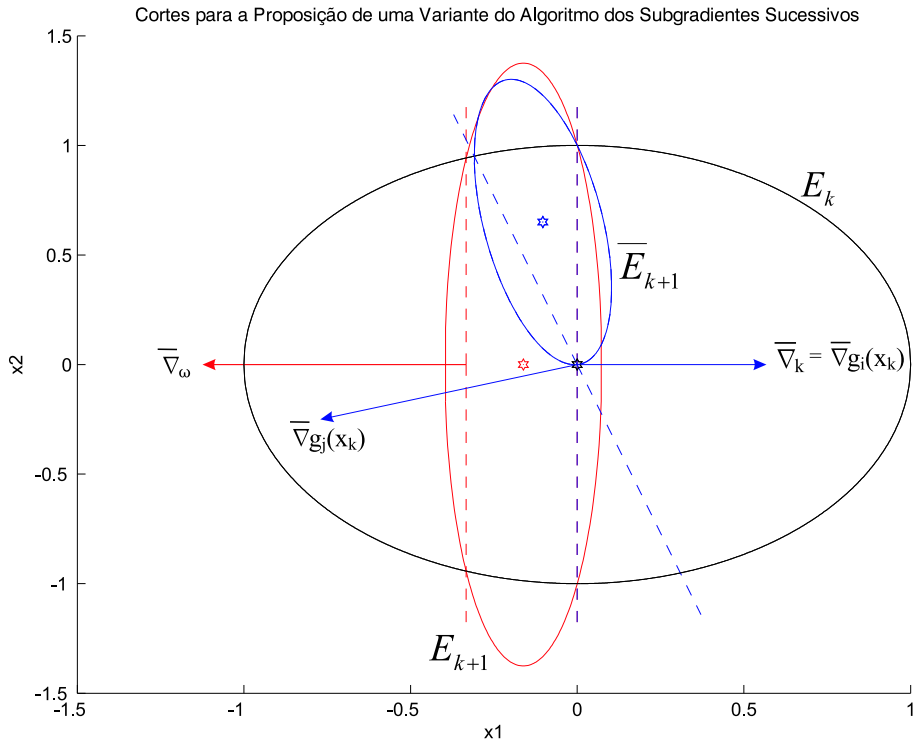


Figura 3.8: Ilustração de uma iteração da variante proposta para o método *Dois Subgradientes Sucessivos*.

3.4.1 Formalização do *Cone de Direções Otimizantes* e do *Cone de Direções Factibilizantes*

O problema (1.8) será aqui tratado e é representado abaixo para a comodidade do leitor:

$$min\ f(\mathbf{x}) \tag{3.40}$$

$$s.a.\ \Omega^* = \{\mathbf{x}^* \in \Omega \mid \nexists \mathbf{x} \in \Omega\ tal\ que\ f(\mathbf{x}) \leq f(\mathbf{x}^*)\ e\ f(\mathbf{x}) \neq f(\mathbf{x}^*)\}$$

$$\Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}^n\}$$

onde, $f(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^m$ e $g(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^p$.

Considere as definições e teoremas, tal como propostos por Dias:

Definição 1 - *Cone de Direções Otimizantes (CDO): Considere* $\mathcal{C}(\mathbf{x}_k)$ *um cone com vértice no ponto* $\mathbf{x}_k$. *Este cone será um CDO,* $\mathcal{C}^O(\mathbf{x}_k)$, *do problema definido por (3.40) se:*

$$\forall \mathbf{x} \in \mathcal{C}(\mathbf{x}_k) \; \exists \varepsilon > 0 \mid f(\mathbf{x}_k + \varepsilon(\mathbf{x} - \mathbf{x}_k)) \leq f(\mathbf{x}_k) \; e \; g(\mathbf{x}_k + \varepsilon(\mathbf{x} - \mathbf{x}_k)) \leq 0 \qquad (3.41)$$

Definição 2 - *Cone de Direções Factibilizantes (CDF): Considere $\mathcal{C}(\mathbf{x}_k)$ um cone com vértice no ponto $\mathbf{x}_k$. Este cone será um CDF, $\mathcal{C}^F(\mathbf{x}_k)$, do problema definido por (3.40) se:*

$$\forall \mathbf{x} \in \mathcal{C}(\mathbf{x}_k) \; \exists \varepsilon > 0 \mid g(\mathbf{x}_k + \varepsilon(\mathbf{x} - \mathbf{x}_k)) \leq g(\mathbf{x}_k) \qquad (3.42)$$

A partir das Definições 1 e 2, temos os seguintes Teoremas:

Teorema 1 *: Considere o problema definido por (3.40) e um ponto $\mathbf{x}_k \mid g_i(\mathbf{x}_k) \leq 0 \; ; \; \forall i = 1, \ldots, p$. Define-se o conjunto $\Psi \triangleq \{i \mid g_i(\mathbf{x}_k) = 0\}$, sendo que h representa o número elementos que o compõe. Suponha $f_i(\cdot)$ e $g_\Psi(\cdot)$ diferenciáveis em $\mathbf{x}_k$ e a matriz G_O definida como indicado a seguir:*

$$G_O = -[\, \nabla f_1(\mathbf{x}_k) \; \ldots \; \nabla f_m(\mathbf{x}_k) \; \nabla g_{\Psi(1)}(\mathbf{x}_k) \; \ldots \; \nabla g_{\Psi(h)}(\mathbf{x}_k) \,] \qquad (3.43)$$

O cone CDO, em $\mathbf{x}_k$, é definido como:

$$\mathbf{x} \in \mathcal{C}^O(\mathbf{x}_k) \iff G_O^T(\mathbf{x} - \mathbf{x}_k) \geq 0 \qquad (3.44)$$

Prova:

i) *Uma vez que $f_j(\cdot)$ é diferenciável em $\mathbf{x}_k$, o vetor $\nabla f_j(\mathbf{x}_k)$ define um hiperplano tangente à superfície de nível da função f_j.*

ii) *O hiperplano corresponderá, por conseguinte, à aproximação de primeira ordem para a hipersuperfície. Conseqüentemente a equação $\nabla f_j(\mathbf{x}_k)^T(\mathbf{x} - \mathbf{x}_k) < 0$ define o semi-espaço tal que:*

$$\forall \mathbf{x} \mid \nabla f_i(\mathbf{x}_k)^T(\mathbf{x} - \mathbf{x}_k) < 0 \; \exists \varepsilon > 0 \mid f(\mathbf{x}_k + \varepsilon(\mathbf{x} - \mathbf{x}_k)) \leq f(\mathbf{x}_k) \qquad (3.45)$$

iii) *O cone $\mathcal{C}(\mathbf{x}_k)$ é a intersecção de todos os semi-espaços associados à $f_i(\cdot)$ e $g_\Psi(\cdot)$*[5]*. Conseqüentemente $\mathcal{C}(\mathbf{x}_k)$ é equivalente à $\mathcal{C}^O(\mathbf{x}_k)$.*

Um primeiro exemplo da ilustração geométrica do Teorema 1, para um ponto $\{\, \mathbf{x} \mid g_i(\mathbf{x}_k) < 0 \; , \; \forall i = 1, 2, 3 \,\}$ é exibido na figura 3.9. Como pode ser observado, o *Cone CDO* é definido pela matriz $G_O = -[\, \nabla f_1(\mathbf{x}) \; \nabla f_2(\mathbf{x}) \,]$, formada exclusivamente dos gradientes

[5] As mesmas definições realizadas em "i"e "ii"podem ser estabelecidas para g_Ψ.

das funções-objetivo, uma vez que o ponto $\mathbf{x}$ se encontra no interior, e não na borda da região factível. Os pontos mínimos irrestritos das funções $f_1(\cdot)$ e $f_2(\cdot)$.

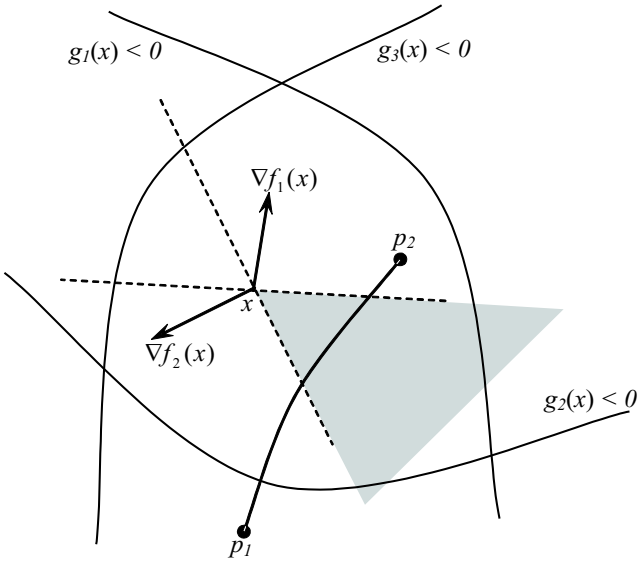


Figura 3.9: Ilustração da existência de um *Cone CDO* em um ponto $\mathbf{x}$ no interior da região factível.

Já o segundo exemplo, figura 3.10, ilustra o Teorema 1 para um ponto $\mathbf{x} \mid \Psi = \{2, 3\}$. Assim, o *Cone CDO* é definido pela matriz $G_O = -[\,\nabla f_1(\mathbf{x})\,\nabla f_2(\mathbf{x})\,\nabla g_2(\mathbf{x})\,\nabla g_3(\mathbf{x})\,]$, uma vez que o ponto $\mathbf{x}$ se encontra na borda da região factível. Em ambos os exemplos, a área colorida mostra, parcialmente, o *Cone CDO* definido pela matriz G_O. Os pontos p_1 e p_2 respectivamente representam os pontos mínimos irrestritos das funções $f_1(\cdot)$ e $f_2(\cdot)$.

Teorema 2 *: Considere o problema definido por (3.40) e um ponto* $\{\mathbf{x}_k \mid g_i(\mathbf{x}_k) > 0$; *para algum* $i = 1, \ldots, p\}$. *Define-se o conjunto* $\Psi \triangleq \{i \mid g_i(\mathbf{x}_k) > 0\}$, *sendo que* v *representa o número elementos que o compõe. Suponha* $f_i(\cdot)$ *e* $g_(\cdot)$ *diferenciáveis em* $\mathbf{x}_k$ *e a matriz* G_F *definida como indicado a seguir:*

$$G_F = -[\,\nabla g_{\Psi(1)}(\mathbf{x}_k)\ \ldots\ \nabla g_{\Psi(v)}(\mathbf{x}_k)\,] \tag{3.46}$$

O cone CDF, em $\mathbf{x}_k$, *é definido como:*

$$\mathbf{x} \in \mathcal{C}^F(\mathbf{x}_k) \iff G_F^T(\mathbf{x} - \mathbf{x}_k) \geq 0 \tag{3.47}$$

Prova:
Analogamente ao demonstrado para o Teorema 1.

Um exemplo da ilustração geométrica do Teorema 2, para dois pontos $\mathbf{x}_a$, $\mathbf{x}_b \mid g_i(\mathbf{x}_a) > 0$ *e* $g_i(\mathbf{x}_b) > 0,\ \forall i = 1, 2, 3\ \}$ é exibido na figura 3.11. Neste caso, o primeiro *Cone*

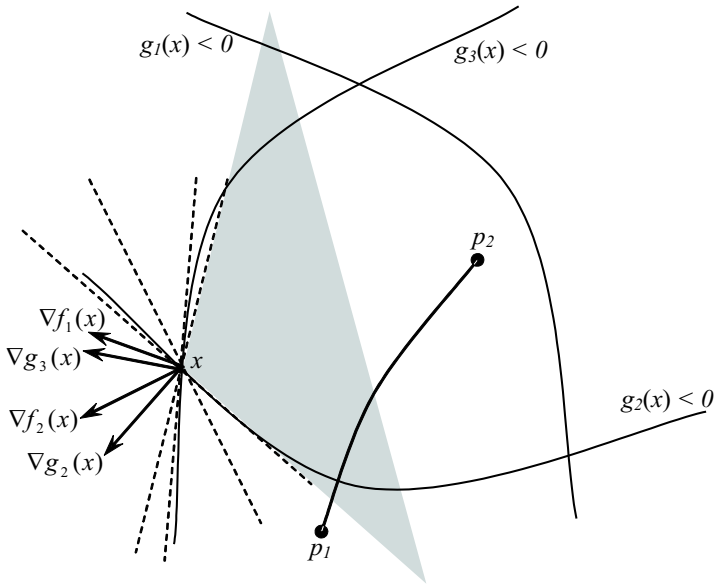


Figura 3.10: Ilustração da existência de um *Cone CDO* em um ponto $\mathbf{x}$ na borda da região factível.

CDF é definido pela matriz $G_F(\mathbf{x}_a) = -[\,\nabla g_2(\mathbf{x}_a)\ \nabla g_3(\mathbf{x}_a)\,]$, enquanto o segundo *Cone CDF* é definido pela matriz $G_F(\mathbf{x}_b) = -[\,\nabla g_1(\mathbf{x}_b)\ \nabla g_3(\mathbf{x}_b)\,]$. A área colorida mostra, parcialmente, os *Cones CDFs* definidos pelas matrizes $G_F(\mathbf{x}_a)$ e $G_F(\mathbf{x}_b)$.

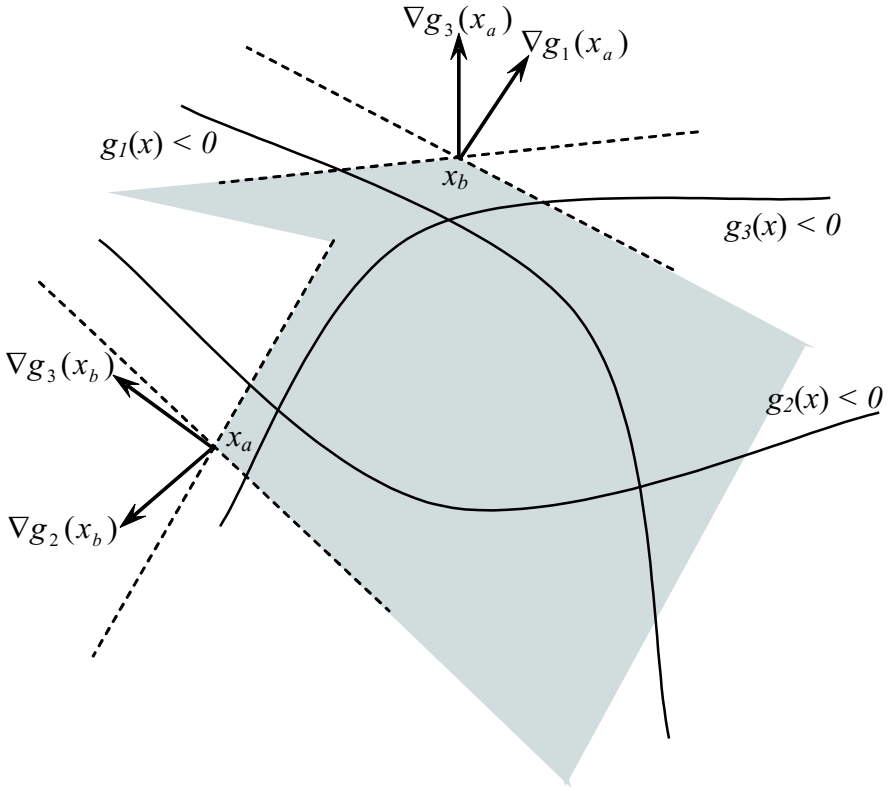


Figura 3.11: Ilustração da existência de dois *Cones CDFs* associados aos pontos $\mathbf{x}_a$ e $\mathbf{x}_b$ exteriores à região factível.

3.4.2 Algoritmo

O objetivo principal é o de possibilitar a utilização do algoritmo MCE na aplicação de casos de problemas vetoriais, partindo do pressuposto que um cone CDO, ou CDF, definido em um ponto $\mathbf{x}_k$ qualquer, sempre ilumina uma parte do conjunto de soluções ótima Ω^*. A transcrição do algoritmo MCE, baseado nas Definições 1 e 2 e nos Teoremas 1 e 2, é exibida a seguir.

Algorithm 6 *Cone-Elipsoidal*

function $MCE(E_0)$

Inicializar o centro do elipsóide, $\mathbf{x}_0$ e a matriz descritora do elipsóide, Q_0, de forma a que o elipsóide inicial contenha pontos pertencentes ao conjunto *Pareto-Ótimo* do problema.

Inicializar a variável lógica EXCONE que indica a existência de um cone no valor *verdade*.

while EXCONE
  if $\mathbf{x}_k$ infactível **then**
    montar matriz G_k com subgradiente das restrições ativas.
  else montar G_k com subgradiente das restrições ativas mais subgradiente das funções-objetivo
  endif
  calcular ε_k no interior do cone definido por G_k, simultaneamente verificando se tal cone *existe*.
  if cone *existe* **then**
    calcular um vetor ξ dentro do cone.
    utilizar a fórmula do método elipsoidal de Shor baseada no vetor $-\xi$ para atualizar o centro do elipsóide, $\mathbf{x}_{k+1}$, e a matriz do elipsóide, Q_{k+1}
  else
    EXCONE $\leftarrow$ *falso*
  endif
endwhile

Deve-se ressaltar, todavia, que uma vez que não é formalizado o *Cone de Busca*, o algoritmo pode não convergir quando o elipsóide inicial não contém todo o conjunto de

soluções ótimas.

Um exemplo desta falha de convergência é exibido na figura 3.12 para problema vetorial irrestrito de duas funções quadráticas com centros sobre o eixo x_1. Para o elipsóide inicial E_0 exibido $(E_0 \cap \Omega^*) \neq \emptyset$. Como pode ser observado, após poucas iterações o elipsóide E_{k+1} não mais contém qualquer ponto do conjunto de soluções ótimas Ω^*, indicado pela reta com os mínimos irrestritos de $f_i(\cdot)$ indicados por asteriscos. Embora o exemplo aqui apresentado não defina $E_0 \supset \Omega^*$, mesmo para esta condição inicial o método pode apresentar falhas.

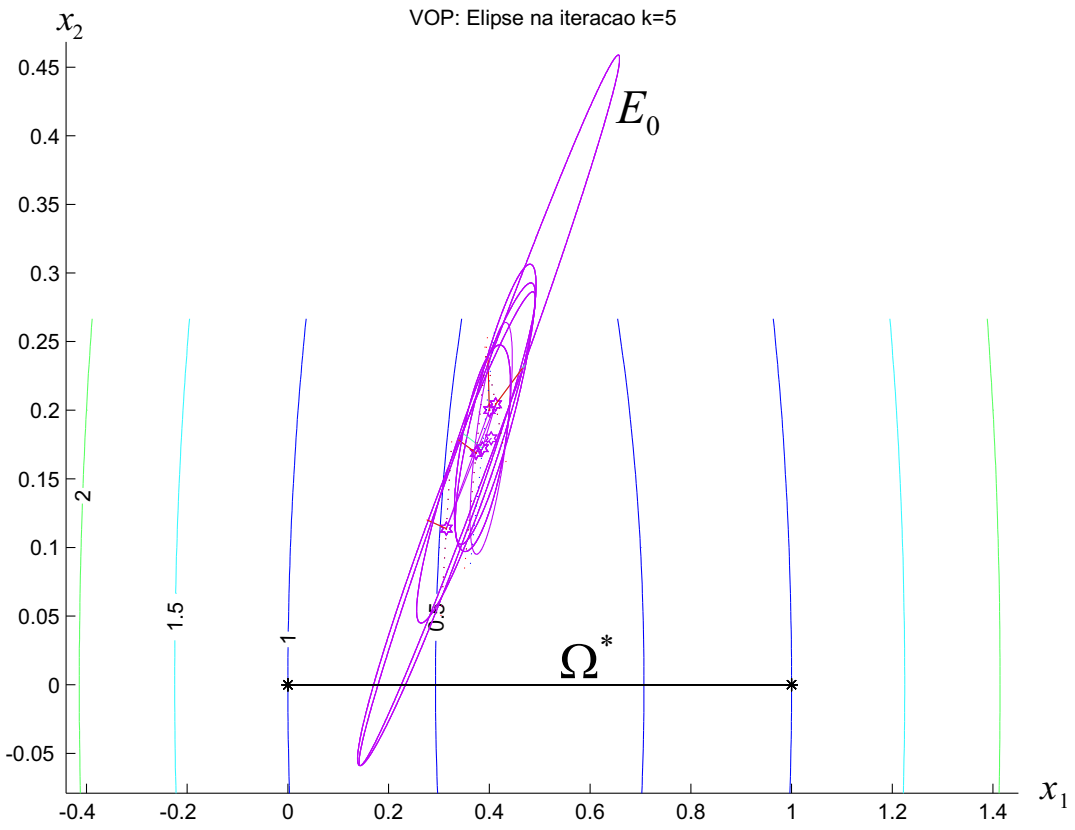


Figura 3.12: Ilustração de um problema vetorial com condições iniciais que induzem o algoritmo *Cone-Elipsoidal* a falha.

3.5 Conclusões do Capítulo

Neste Capítulo foram apresentadas as variantes do método *Elipsoidal* que utilizam mais de um vetor subdiferencial para o cálculo do novo elipsóide.

Para a atualização de *posto 1* da matriz Q_{k+1} foi descrito o método de *Corte Paralelo* enquanto que para a atualização de *posto 2* foi descrito o método de *Corte em Cunha*.

Em seqüência, foi apresentada o algoritmo *Dois Subgradientes Sucessivos* que determina, dentro da seqüência de convergência, qual o melhor corte possível para o incremento da contração do volume do novo elipsóide E_{k+1}. Uma variação para o algoritmo *Dois Subgradientes Sucessivos*, a qual possibilita utilizar a construção de um *Corte em Cunha* quando da existência de duas restrições violadas simultaneamente em uma iteração, foi proposta. Por fim foi apresentado o *Método Cone-Elipsoidal* para problemas vetoriais.

No capítulo seguinte será apresentada uma revisão referente às condições de eficiência de Kuhn-Tucker, será proposta uma nova condição necessária e suficiente de infactibilidade para problemas escalares e vetoriais quasi-convexos, bem como também serão definidos os *cones complementares*.

Capítulo 4

Infactibilidade Estrita, Pareto-Otimalidade e Condições Complementares

Neste capítulo é apresentada uma revisão bibliográfica referente às condições de eficiência de Kuhn-Tucker. Em seguida, iniciam-se as contribuições desta Tese a partir de um proposta, por similaridade estrutural às condições de eficiência, de uma nova condição necessária e suficiente de infactibilidade para problemas não-lineares quasi-convexos.

Em seguida, são definidos os *cones complementares*, quando da não satisfação das condições de eficiência e infactibilidade, utilizados para convergir uma seqüência de pontos para: (i) um ponto factível, em problemas de factibilidade; (ii) um ponto do conjunto *Pareto-Ótimo*, em problemas de otimização, ou (iii) um ponto em que exista a condição necessárias e suficientes de infactibilidade (INF).

Maiores detalhes sobre os temas tratados nas seções apresentadas neste capítulo podem ser conseguidos nas referências (Topkis & Veinott 1967, Bazaraa & Shetty 1979, Rockafellar n.d., Chankong & Haimes 1984, Luenberger 1984, Benson 1984, Benson & Aksoy 1991, Miettinen n.d., Sergienko, Lebedeva & Semenova 2000, Adán & Novo 2003, Takahashi 2003).

4.1 Introdução

Os conceitos de otimalidade e eficiência podem ser definidos matematicamente por diversas formas (Heing 1982, Luc n.d., Miettinen n.d., Takahashi 2003). Esta Tese é principalmente baseada na definição do cone de direções factíveis proposto por (Miettinen & Mäkelä 2001) e derivada do cone contingente definido por (Rockafellar n.d.). Embora não

explicitamente referenciadas por (Dias 2003), a Definição 1 (página 53) e a Definição 2 (página 54) são derivadas dos cones supracitados.

A análise estrutural das condições de eficiência de Kuhn-Tucker (KTE) permite propor por similaridade uma nova visão de uma condição necessária e suficiente de infactibilidade (INF) para problemas não-lineares escalares e vetoriais estritamente quasi-convexos. A principal idéia é caracterizar infactibilidade como a impossibilidade de alcançar simultaneamente valores menores para as restrições que se encontram violadas. Neste caso, a existência de um ponto, no qual esta impossibilidade ocorre, pode ser caracterizada por uma condição aqui denominada como INF. Assim, a condição INF pode ser interpretada como uma simples adaptação das tradicionais condições KTE[1] (Chankong & Haimes 1984).

Como será mostrado na seção 4.2, a não satisfação de ambas as condições KTE e INF em um ponto definirá os *cones complementares*. Estes cones, definidos por vetores subdiferenciais que serão utilizados para testar as condições KTE e INF, representam regiões do espaço nas quais as condições KTE, INF, ou ainda factibilidade, poderão ocorrer. Desta forma, estes cones podem ser utilizados para estabelecer exclusão de regiões do espaço e criar algoritmos que, de forma determinística, terminarão em: (i) um ponto factível, em problemas de factibilidade; (ii) um ponto do conjunto *Pareto-Ótimo*, em problemas de otimização, ou (iii) um ponto em que exista a condição necessária e suficiente de infactibilidade, para ambos os tipos de problemas.

O conceito dos *cones complementares* que é utilizado aqui de forma análoga ao tradicional conjunto de direções factíveis (i.e., do inglês *feasible direction set*) para otimização mono-objetivo (Topkis & Veinott 1967, Bazaraa & Shetty 1979, Luenberger 1984, Miettinen & Mäkelä 2001), o qual deu origem aos métodos de projeção (i.e., do inglês *projective methods*). Estes métodos de projeção se baseiam na projeção de um ponto dentro de determinados conjuntos. Na análise convexa moderna a programação cônica (i.e., do inglês *conic programming*) tem sido estruturada com base em certos cones que contêm conjuntos convexos (Ben-Tal & Nemirovski 2001). Este fato tem permitido a extensão de diversos resultados da programação linear para problemas convexos gerais, incluindo relações de dualidade. Também tem permitido a generalização do método de pontos interiores para a solução de problemas não-lineares convexos (Nesterov, Todd & Ye 1999, Ben-Tal & Nemirovski 2001). Os *cones complementares* aqui empregados são relacionados aos cones tangentes (i.e., do inglês *tangent cones*) da análise convexa.

[1]Deve-se notar que as condições de Karush-Kuhn-Tucker para otimalidade no caso de problemas mono-objetivo são um caso particular das condições KTE.

No contexto da otimização vetorial, conceitos análogos tem sido empregados no estágio de decisão de problemas de programação linear (Benson 1984, Benson & Aksoy 1991), para o propósito de caracterização de condições que descrevam a existência de soluções eficientes em problemas não-lineares (Sergienko et al. 2000), ou para de soluções eficientes (Adán & Novo 2003). Nesta Tese os *cones complementares* são apresentados como objetos complementares associados à não satisfação da condições de eficiência KTE nem da condição de infactibilidade INF.

As condições KTE são verificadas, em problemas factíveis, em pontos eficientes pertencentes ao conjunto *Pareto-Ótimo*. A condição de infactibilidade, dada a sua forma similar em estrutura às condições KTE, são verificadas em um conjunto de pontos não convexo. Sob a condição de que o número de funções-objetivo independentes seja maior que a dimensão do espaço de variáveis, sabe-se que o conjunto *Pareto-Ótimo* possui volume não zero(Chankong & Haimes 1984). Analogamente, os pontos nos quais são verificadas a condição de INF determinam um conjunto de volume não zero quando o número de restrições violadas é maior que a dimensão do espaço de variáveis.

Um resultado tradicional da programação linear é a construção de certificados de infactibilidade (i.e., do inglês *infeasibility certificates*) baseados na noção de dualidade (Luenberger 1984, Ben-Tal & Nemirovski 2001). Na literatura recente sobre análise convexa, alguns certificados de infactibilidade têm sido obtidos para a programação cônica por meio de dualidade cônica (i.e., do inglês *conic duality*) (Nesterov et al. 1999, Ben-Tal & Nemirovski 2001). A condição INF, aqui proposta, é equivalente a um certificado de infactibilidade em espaços de dimensão finita e apresenta a singularidade de ser derivada unicamente de variáveis primais. Ressalta-se ainda que a condição INF é provida de uma estrutura particular, a qual será discutida nas próximas seções, bem como que esta região poderá ser procurada com algoritmos (Kiwiel 1995, Liao & Todd 1996, Goffin & Vial 1999, Shor 1977) utilizados para procurar a região factível.

4.2 Definição da Condição de Infactibilidade

O problema (1.8) será aqui tratado e é representado abaixo para a comodidade do leitor:

$$min\ f(\mathbf{x}) \tag{4.1}$$

$$s.a.\ \Omega^* = \{\mathbf{x}^* \in \Omega \mid \nexists\, \mathbf{x} \in \Omega\ tal\ que\ f(\mathbf{x}) \leq f(\mathbf{x}^*)\ e\ f(\mathbf{x}) \neq f(\mathbf{x}^*)\}$$

$$\Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}^n\}$$

onde, $f(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^m$ e $g(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^p$.

Considere o problema descrito por (4.1). O objetivo principal de um problema de otimização vetorial é encontrar algum ponto $\mathbf{x}^* \in \Omega^*$ que descreva o melhor compromisso entre as diferentes funções-objetivo que definem $f(\cdot)$. Isto também significa que não existe outro ponto $\mathbf{x} \in \Omega^*$ no qual alguma função-objetivo $f_j(\cdot)$ do vetor $f(\cdot)$ pode ter seu valor melhorado, quando comparado ao valor $f_j(\mathbf{x}^*)$, sem a piora do valor de alguma outra função-objetivo presente no mesmo vetor $f(\cdot)$. O conjunto dos pontos que apresentam esta característica é chamado conjunto *Pareto-Ótimo* ou *Pareto*, Ω^*, e seus pontos são chamados de soluções *Pareto-Ótimo*, eficientes, não-dominadas ou não-inferiores (Luc n.d., Miettinen n.d., Takahashi 2003). O conceito de dominância é apresentado a seguir a partir da seguinte notação:

$$
\begin{aligned}
\mathbf{w} \prec \mathbf{y} \quad &:= \text{operador } \prec \text{ sobre } \mathbf{w} \text{ e } \mathbf{y} \in \mathbb{R}^n \text{ tal que } \mathbf{w}(i) \leq \mathbf{y}(i)\ \forall\, i = 1, \ldots, n \\
&\quad \text{e } \mathbf{w}(j) < \mathbf{y}(j) \text{ para algum } j = 1, \ldots, n. \\
\mathbf{w} \not\prec \mathbf{y} \quad &:= \text{negação de } \mathbf{w} \prec \mathbf{y}. \\
\mathbf{w} \succ \mathbf{y} \quad &:= \text{operador } \succ \text{ sobre } \mathbf{w} \text{ e } \mathbf{y} \in \mathbb{R}^n \text{ tal que } \mathbf{w}(i) \geq \mathbf{y}(i)\ \forall\, i = 1, \ldots, n \\
&\quad \text{e } \mathbf{w}(j) > \mathbf{y}(j) \text{ para algum } j = 1, \ldots, n. \\
\mathbf{w} \not\succ \mathbf{y} \quad &:= \text{negação de } \mathbf{w} \succ \mathbf{y}.
\end{aligned}
$$

Definição 3 - *Dominância : Diz-se que o ponto* $\mathbf{x}_1$ *domina o ponto* $\mathbf{x}_2$ *se* $f(\mathbf{x}_1) \prec f(\mathbf{x}_2)$.

Assim, para o problema definido por (4.1) e um ponto qualquer $\bar{\mathbf{x}}$, uma das quatro condições abaixo irá ocorrer:

(a) $\bar{\mathbf{x}} \in \Omega^*$, o que significa que as condições de eficiência KTE são satisfeitas (Chankong & Haimes 1984), ou, para algum $\lambda \in \mathbb{R}^m$ e $\mu \in \mathbb{R}^p$:

$$
\text{(KTE)} \quad \begin{cases} \lambda^T F(\bar{\mathbf{x}}) + \mu^T G(\bar{\mathbf{x}}) \ = \ 0 \\ \lambda \succ 0 \ , \ \mu \geq 0 \\ g_j(\bar{\mathbf{x}}) \ \leq \ 0 \ ; \forall\, j = 1, \ldots, p \\ \mu(i) g_i(\bar{\mathbf{x}}) \ = \ 0 \ ; \ \forall\, i = 1, \ldots, p \end{cases} \tag{4.2}
$$

A figura 4.1 apresenta uma ilustração de um ponto $\mathbf{x}^*$ que satisfaz as condições de eficiência de Kuhn-Tucker. Como pode ser visto geometricamente, existe um conjunto de multiplicadores não negativos λ, μ, tal que $(\lambda(1)\bar{\nabla} f_1(\mathbf{x}^*) + \lambda(2)\bar{\nabla} f_2(\mathbf{x}^*) + \mu(1)\bar{\nabla} g_1(\mathbf{x}^*) + \mu(2)\bar{\nabla} g_2(\mathbf{x}^*)) = 0$.

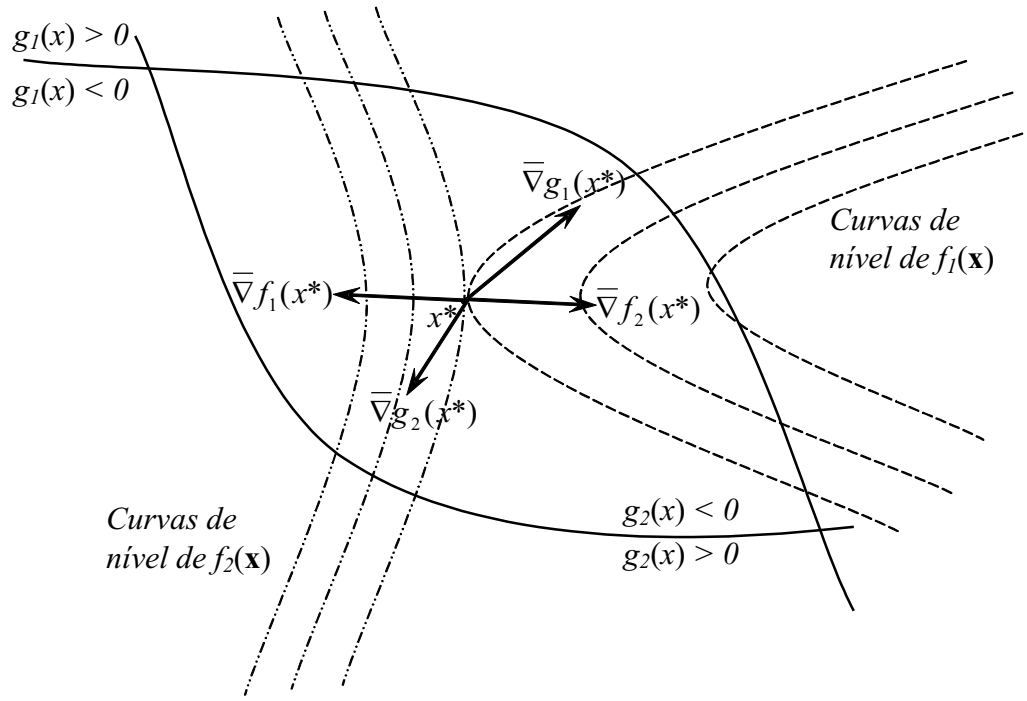


Figura 4.1: Ilustração de um ponto $\mathbf{x}^*$ que satisfaz KTE.

(b) $\bar{\mathbf{x}} \in \Lambda$, com Λ definido como o conjunto de pontos nos quais:

$$(\text{INF}) \begin{cases} \exists\ i \mid g_i(\bar{\mathbf{x}}) > 0 \\ \mu^T G(\bar{\mathbf{x}}) \ = \ 0 \\ \mu \succ 0 \\ g_j(\bar{\mathbf{x}}) < 0 \ \Rightarrow \ \mu_j \ = \ 0 \end{cases} \tag{4.3}$$

para algum multiplicador $\mu \in \mathbb{R}^m$.

A figura 4.2 apresenta uma ilustração de um ponto $\mathbf{x}_k$ com três restrições violadas e que satisfaz as condições para infactibilidade. Como pode ser visto geometricamente, existe um conjunto de multiplicadores positivos $\mu(1) > 0$, $\mu(2) > 0$ e $\mu(3) > 0$, tal que $(\mu(1)\bar{\nabla} g_1(\mathbf{x}_k) + \mu(2)\bar{\nabla} g_2(\mathbf{x}_k) + \mu(3)\bar{\nabla} g_3(\mathbf{x}_k)) = 0$.

(c) $\bar{\mathbf{x}} \in \Omega$ e $\bar{\mathbf{x}} \notin \Omega^*$.

(d) $\bar{\mathbf{x}} \notin \Omega$ e $\bar{\mathbf{x}} \notin \Lambda$.

Um exemplo, no qual nem a condição de infactibilidade INF nem a condição de eficiência de KTE são atendidas no ponto $\bar{\mathbf{x}}_k$, é exibido na figura 4.3. Como pode ser observado, não existe um conjunto de multiplicadores positivos λ e μ, tal que $(\lambda\bar{\nabla} f_1(\mathbf{x}_k) + \mu(1)\bar{\nabla} g_1(\mathbf{x}_k) + \mu(2)\bar{\nabla} g_2(\mathbf{x}_k)) = 0$.

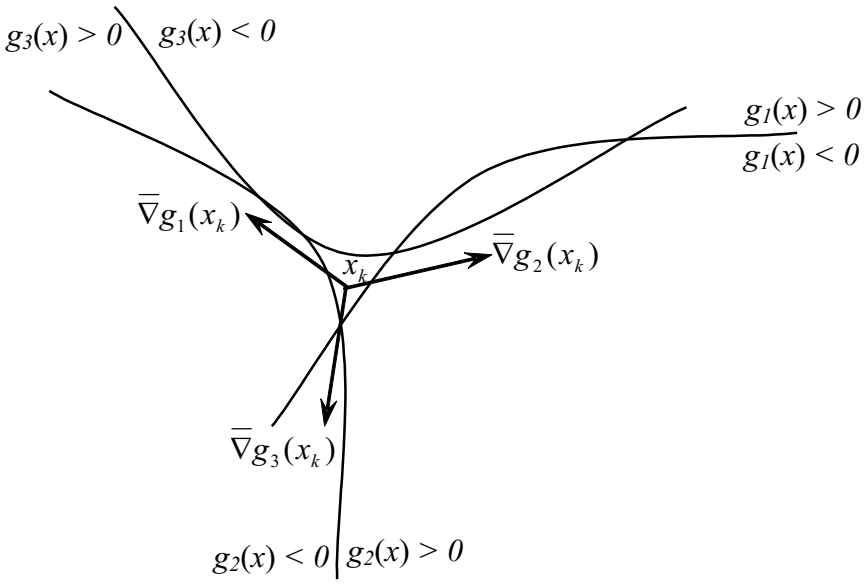


Figura 4.2: Ilustração de um ponto $\mathbf{x}_k$ que satisfaz INF.

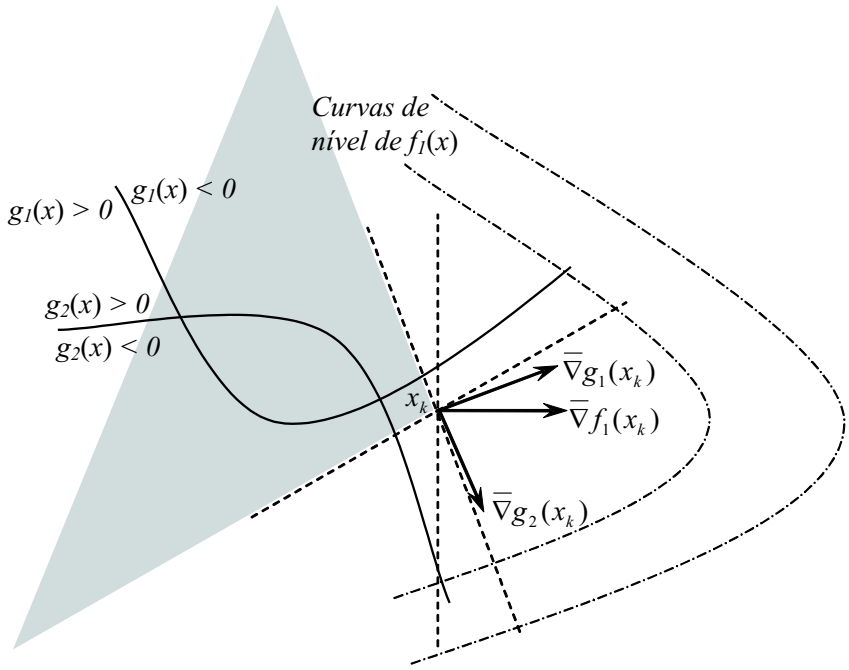


Figura 4.3: Ilustração de um ponto $\mathbf{x}_k$, no qual KTE e INF não são satisfeitas. A área colorida mostra, parcialmente, o *Cone Complementar* definido por $\bar{\nabla} f_1(\mathbf{x}_k)$, $\bar{\nabla} g_1(\mathbf{x}_k)$ e $\bar{\nabla} g_2(\mathbf{x}_k)$.

4.3 Condições de Decisão

Lema 1 *: Considere os conjuntos Λ e Ω tal como definido por (4.3) e (4.1). Então:*

$$\bar{x} \in \Lambda \Rightarrow \Omega = \emptyset \tag{4.4}$$

Prova:
Esta conclusão pode ser obtida pela observação de que, interpretando-se as funções-restrição violadas $g_j(\cdot)$ como funções-objetivo de um problema vetorial irrestrito auxiliar, as condições INF são equivalentes a considerar $\bar{\mathbf{x}}$ como sendo um ponto não-dominado deste problema auxiliar. Isto significa que, no problema original, cada função violada somente deixará de ser violada ao custo de violar ainda mais outra restrição. A convexidade das conjuntos subníveis das funções $g(\cdot)$ garantem que a condição de infactibilidade INF é global.

O Lema 1 estabelece uma condição suficiente para a existência do conjunto Λ para problemas definidos por (4.1). O Lema seguinte estabelece a condição necessária.

Lema 2 *: Seja $g_i(\cdot) : \mathbb{R}^n \mapsto \mathbb{R}$ uma função quasi-convexa. Considere o problema de factibilidade:*

$$\begin{aligned} &\textit{encontre } \mathbf{x} \\ &\textit{s.a. } \mathbf{x} \in \Omega \end{aligned} \tag{4.5}$$

no qual o conjunto Ω é definido tal que:

$$\Omega \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid g_i(\mathbf{x}) \leq 0 \ , \ i = 1, \ldots, p\} \tag{4.6}$$

Considere também o conjunto Γ como sendo um conjunto não vazio e definido por:

$$\Gamma \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid g_i(x) \leq 0 \ , \ i = 1, \ldots, r\} \tag{4.7}$$

para $r \leq p$. Por fim considere o conjunto Λ tal como definido por (4.3). Então:

$$\Omega = \emptyset \Rightarrow \Lambda \neq \emptyset \tag{4.8}$$

Prova:
Para $\Omega = \emptyset$ temos que $r < p$. Existem dois casos a serem considerados:

(i) *$\exists \ k \leq p$ tal que o conjunto $\Phi = \{\mathbf{x} \in \mathbb{R}^n \mid g_k(\mathbf{x}) \leq 0\}$ determina que $\Phi \cap \Gamma = \emptyset$. Neste caso, defina um problema auxiliar de otimização vetorial (minimização) com as funções $\max\{g_1(\mathbf{x}), \ldots, g_r(\mathbf{x})\}$ e $g_k(\mathbf{x})$. O conjunto Pareto-Ótimo do problema*

auxiliar é uma curva que intercepta a fronteira de Γ*, e aproxima, ao menos assintoticamente, a fronteira de* Φ*. Este conjunto Pareto-Ótimo é composto por pontos que pertencem a* Λ*.*

(ii) $\nexists\ k \leq p$ *tal que o conjunto* $\Phi_k = \{\mathbf{x} \in \mathbb{R}^n \mid g_k(\mathbf{x}) \leq 0\}$ *determina* $\Phi_k \cap \Gamma = \emptyset$*. Neste caso, defina o conjunto* $\Psi = \Gamma \cap \Phi_{k1} \cap \Phi_{k2} \cap \ldots \cap \Phi_{kv}$*, incluindo todos os possíveis conjuntos* Φ_{ki} *de modo que* Ψ *permaneça não vazio. Utilize agora qualquer* Φ_α*, tal que* $\Psi \cap \Phi_\alpha = \emptyset$*, e defina o problema vetorial auxiliar (minimização) com as funções* $\max\{g_1(\mathbf{x}), \ldots, g_r(\mathbf{x}), g_{k1}(\mathbf{x}), g_{k2}(\mathbf{x}), \ldots, g_{kv}(\mathbf{x})\}$ *e* $g_\alpha(\mathbf{x})$*. O conjunto Pareto-Ótimo deste problema auxiliar é uma curva que intercepta a fronteira de* Ψ *e* Φ_α*. Este conjunto Pareto-Ótimo é composto por pontos que pertencem a* Λ*.*

O mais importante resultado com respeito às condições INF é apresentado no Teorema 3 a partir dos Lemas 1 e 2.

Teorema 3 - *Conjunto Infactível : Considere o problema de otimização definido por (4.1), o conjunto* Λ *definido por (4.3), o conjunto* Ω^* *definido pelas condições KTE e o conjunto* Ω *tal como definido no Lema 2. Então:*

$$\begin{aligned} &\Lambda = \emptyset \Leftrightarrow \Omega \neq \emptyset \Leftrightarrow \Omega^* \neq \emptyset \\ &\Lambda \neq \emptyset \Leftrightarrow \Omega = \emptyset \Leftrightarrow \Omega^* = \emptyset \end{aligned} \tag{4.9}$$

Prova:
Diretamente dos Lemas 1 e 2.

A partir do Teorema 3, a existência do conjunto Λ torna-se uma condição de decisão para um problema definido por 4.1. As condições KTE e INF podem ser consideradas condições de decisão, uma vez que estas podem ser utilizadas em um algoritmo para decidir se um problema é infactível ou se um ponto pertence ao conjunto *Pareto-Ótimo*. Alcançada quaisquer destas condições, um algoritmo que esteja sendo executado para solucionar um problema definido por (4.1) ou um problema de factibilidade, pode ser interrompido.

4.3.1 Cones e Condições Complementares

Nos casos (c) e (d) apresentados na seção 4.2, no qual um ponto $\bar{x}$ não satisfaz qualquer das condições de decisão, algumas condições complementares a KTE e INF podem ser estabelecidas. A condição associada à não satisfação da condição KTE é dada por:

Corolário 1 *: Considere o cone $\mathcal{C}^O(\bar{\mathbf{x}})$, com vértice em $\bar{\mathbf{x}}$, formalizado pela Definição 1 (página 53), o conjunto Ω definido por (4.6) e o conjunto Ω^* definido por (4.1). Se $\bar{\mathbf{x}} \in \Omega$ e $\bar{\mathbf{x}} \notin \Omega^*$, então $\mathcal{C}^O(\bar{\mathbf{x}}) \cap \Omega^* \neq \emptyset$.*

A existência do cone $\mathcal{C}^O(\bar{\mathbf{x}})$ é complementar às condições KTE no sentido de que este cone existe se e somente se as condições KTE não são satisfeitas em um ponto qualquer $\bar{\mathbf{x}}$. Similarmente, uma condição complementar pode ser estabelecida para o caso da condição INF não ser satisfeita.

Corolário 2 *: Considere um cone $\mathcal{C}^F(\bar{\mathbf{x}})$, com vértice $\bar{\mathbf{x}}$, formalizado pela Definição 2 (página 54) e o conjunto Ω definido por (4.6). Suponha que $\bar{\mathbf{x}} \notin \Omega$ e $\bar{\mathbf{x}} \notin \Lambda$. Então:*

i. Se $\Omega \neq \emptyset$, tem-se que $\mathcal{C}^F(\bar{\mathbf{x}}) \supset \Omega$;

ii. Se $\Lambda \neq \emptyset$, tem-se que $\mathcal{C}^F(\bar{\mathbf{x}}) \cap \Lambda \neq \emptyset$.

De forma análoga ao caso anterior, a existência do cone $\mathcal{C}_I(\bar{x})$ existe se e somente se a condição INF não é satisfeita em um ponto qualquer $\bar{x}$. Ambos os cones $\mathcal{C}^O(\bar{\mathbf{x}})$ e $\mathcal{C}^F(\bar{\mathbf{x}})$ indicam regiões nas quais as condições KTE e INF poderão ser satisfeitas.

Por conseguinte, algumas notas referentes à estrutura dos conjuntos Ω, Ω^* e Λ devem ser ressaltadas. As primeiras três notas são correspondentes entre si, sendo a Nota 1 bem conhecida.

Nota 1 *: Suponha as funções $g_i(\cdot)$ quasi-convexas. Então, o conjunto Ω é um conjunto vazio, um simples ponto no $\mathbb{R}^n$ ou uma região de volume não zero neste espaço.*

Nota 2 *: Suponha as funções $g_i(\cdot)$ quasi-convexas e defina o conjunto $\Omega_g(\phi)$ para um dado vetor $\phi \in \mathbb{R}^m$, $\phi \geq 0$:*

$$\Omega_g(\phi) \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid g(\mathbf{x}) \leq \phi\} \tag{4.10}$$

Então, o conjunto $\Omega_g(\phi)$ é um conjunto vazio, um simples ponto no $\mathbb{R}^n$ ou uma região de volume não zero neste espaço.

Nota 3 *: Suponha as funções $g_i(\cdot)$ quasi-convexas e considere um $\epsilon \in \mathbb{R}^p$. Então, o conjunto $\Omega_f(\epsilon)$, definido por*

$$\Omega_f(\epsilon) \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) \leq \epsilon\} \tag{4.11}$$

é um conjunto vazio, um simples ponto no $\mathbb{R}^n$ ou uma região de volume não zero neste espaço.

A Nota 4 é um resultado conhecido (Chankong & Haimes 1982, Chankong & Haimes 1984). Dada a estrutura similar de Ω^* e Λ, a Nota 5 também é verdadeira.

Nota 4 *: Para qualquer $\epsilon \in \mathbb{R}^p$, tal que o conjunto $\Omega_f(\epsilon)$ seja não vazio, a relação a seguir é verdadeira.*

$$\Omega_f(\epsilon) \cap \Omega^* \neq \emptyset \tag{4.12}$$

Nota 5 *: Para qualquer $\phi \in \mathbb{R}^m$, tal que os conjuntos $\Omega_g(\phi)$ e Λ são não vazios, a relação a seguir é verdadeira.*

$$\Omega_g(\phi) \cap \Lambda \neq \emptyset \tag{4.13}$$

A nota 4 é, de fato, a base para o método P_ϵ encontrar amostras do conjunto *Pareto-Ótimo* (Chankong & Haimes 1982). A idéia primordial por trás deste método pode ser re-escrita pela seguinte nota:

Nota 6 *: Considere um $\epsilon \in \mathbb{R}^p$ tal que o conjunto $\Omega_f(\epsilon)$ é não vazio. Então, existe um vetor $\omega \in \mathbb{R}^p$ e um escalar $\lambda \in \mathbb{R}$ tal que $\omega \succ 0$, $\lambda > 0$ e:*

$$\Omega_f(\epsilon - \lambda\omega) \in \Omega^* \tag{4.14}$$

Neste caso, Ω_f é reduzido a um único ponto de Ω^* para o máximo λ no qual (4.14) permanece verdadeira.

Como outra conseqüência da similaridade entre Ω^* e Λ, tem-se como verdadeira a nota a seguir:

Nota 7 *: Considere um vetor $\phi \in \mathbb{R}^m$ tal que o conjunto $\Omega_g(\phi)$ é não vazio. Considere também que $\Lambda \neq \emptyset$. Então, existe um escalar $\lambda \in \mathbb{R}$, $\lambda > 0$ tal que:*

$$\Omega_g(\lambda\phi) \in \Lambda \tag{4.15}$$

Neste caso, Ω_g será reduzido a um único ponto de Λ para o máximo λ no qual (4.15) permanece verdadeira.

Isto significa que a condição de infactibilidade pode ser caracterizada de forma similar à forma na qual o conjunto *Pareto-Ótimo* pode ser encontrado. Adicionalmente, tem-se que a busca por pontos factíveis conduzirá a um ponto que caracterize a condição de infactibilidade, caso este exista.

Por fim, pode-se ainda registrar algumas notas com respeito à dimensão dos conjuntos Ω, Ω^* e Λ.

Nota 8 *: Nos casos não-degenerados,* $\dim(\Omega) = n$.

Nota 9 *: Nos casos não-degenerados,* $\dim(\Omega^*) = \min(\{m-1, n\})$.

Nota 10 *: Considere que um problema tem* $\bar{p}$ *restrições violadas. Nos casos não-degenerados,* $\dim(\Lambda) = \min(\{\bar{p}-1, n\})$.

As notas 8 e 9 são bem conhecidas. Como conseqüência da similaridade estrutural entre Ω^* e Λ, a nota 10 pode ser diretamente estabelecida.

4.4 Conclusões do Capítulo

Neste Capítulo foi apresentada uma revisão bibliográfica referente às condições KTE e proposta uma nova condição necessária e suficiente INF para problemas escalares e vetoriais quasi-convexos. Em seqüência, foram definidos os *cones complementares* que permitem a definição de condições complementares às condições KTE e INF. Um conjunto de Lemas, Teoremas e Notas associadas as idéias propostas foram também apresentados.

No capítulo seguinte serão detalhados os fundamentos propostos para os métodos que utilizarão a construção de *cones complementares* para definir a família de métodos *Poliedro-Elipsoidais*.

Capítulo 5

Fundamento dos Métodos Poliedro-Elipsoidais para Problemas Contínuos

Neste capítulo são propostos os fundamentos para os métodos *Poliedro-Elipsoidais* para a solução de problemas escalares e vetoriais com funções quasi-convexas e não necessariamente diferenciáveis (QCND) com variáveis contínuas. Inicialmente o conceito do poliedro de busca é formalizado por definições e teoremas, os quais permitirão construir uma variante de um método baseado em *Elipsóide* para a solução de problemas escalares ou vetoriais QCND. Em seqüência, é proposto o poliedro de busca com histórico de intersecções que definirá a menor região de interesse de busca, sob certas considerações, para a determinação de um novo elipsóide E_{k+1} que certamente preservará parte do conjunto *Pareto-Ótimo*.

Outros detalhes sobre estes temas podem ser conseguidos na referência (Moura & Takahashi 2007).

5.1 Formalização do *Poliedro de Busca*

O problema (1.8) será aqui tratado e é representado abaixo para a comodidade do leitor:

$$min\ f(\mathbf{x}) \tag{5.1}$$

$$s.a.\ \Omega^* = \{\mathbf{x}^* \in \Omega \mid \nexists\, \mathbf{x} \in \Omega\ tal\ que\ f(\mathbf{x}) \leq f(\mathbf{x}^*)\ e\ f(\mathbf{x}) \neq f(\mathbf{x}^*)\}$$

$$\Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}^n\}$$

onde, $f(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^m$ e $g(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^p$.

O cone $\mathcal{C}^O(\mathbf{x})$ define um conjunto de direções que podem melhorar a solução obtida em um ponto $\bar{\mathbf{x}}$. Da definição do conjunto Ω^*, não existe um cone $\mathcal{C}^O$ para um ponto $\mathbf{x}^* \in \Omega^*$. Tal como formalizado pela Definição 1 (página 53) para um dado ponto $\mathbf{x}_k \notin \Omega^*$ existe algum ponto $\mathbf{x}_k + \alpha\mathbf{d}$ que melhore $f(\mathbf{x}_k)$. Para qualquer função-objetivo $f_j(\cdot)$ tem-se que $L_{f_j=f_j(\mathbf{x}_k)} \subset H_-^{\bar{\nabla} f_j(\mathbf{x}_k),\mathbf{x}_k}$. Analogamente, considerando o conjunto $\Psi \triangleq \{i \mid g_i(\mathbf{x}_k) = 0\}$, para qualquer função-restrição $g_i(\cdot) \mid i \in \Psi$ o conjunto $L_{g_i=g_i(\mathbf{x}_k)} \subset H_-^{\bar{\nabla} g_i(\mathbf{x}_k),\mathbf{x}_k}$. Por conseguinte, para um conjunto *Pareto-Ótimo* não vazio existe ao menos um ponto

$$\mathbf{x}^* \in (H_-^{\bar{\nabla} f_1(\mathbf{x}_k),\mathbf{x}_k} \cap \ldots \cap H_-^{\bar{\nabla} f_m(\mathbf{x}_k),\mathbf{x}_k} \cap H_-^{\bar{\nabla} g_\Psi(\mathbf{x}_k),\mathbf{x}_k}). \tag{5.2}$$

Embora a intersecção dos semi-espaços, apresentada acima, possa ser utilizada para definir um algoritmo de otimização baseado em direção de busca, esta intersecção não pode ser utilizada diretamente para definir um método baseado em elipsóide. A razão fundamental para esta restrição encontra-se na limitação da região de busca, a qual é definida pelo elipsóide corrente. Durante o processo de convergência, o volume do elipsóide é comprimido em um volume cada vez menor, mas que sempre deve conter a solução do problema. Diretamente da Definição 1 (página 53), para problemas factíveis, existe um ponto $\mathbf{x}^*$ dentro do cone $\mathcal{C}^O$ definido em todo ponto $\{\mathbf{x} \mid \mathbf{x} \in \Omega \text{ e } \mathbf{x} \notin \Omega^*\}$, todavia não necessariamente o ponto $\mathbf{x}^*$ se encontra dentro do elipsóide corrente. Este fato justifica a falha do método *MCE* discutido na seção 3.4.

A figura 5.1 exemplifica a limitação da construção de um método baseado em *Elipsóide* diretamente da utilização da Definição 1 (página 53) para um problema hipotético com $n = 2$, $m = 2$. Na figura, o primeiro elipsóide E_k intercepta parte do conjunto *Pareto-Ótimo* que se encontra dentro do cone $\mathcal{C}^O(\mathbf{x}_k)$. O novo elipsóide E_{k+1} também contém o subconjunto *Pareto-Ótimo* $\mathcal{C}^O(\mathbf{x}_k) \cap \Omega^*$, entretanto o cone $\mathcal{C}^O(\mathbf{x}_{k+1})$ não intercepta o subconjunto *Pareto-Ótimo* que se encontra dentro do elipsóide E_{k+1}. Os pontos p_1 e p_2 respectivamente representam os pontos mínimos irrestritos das funções $f_1(\cdot)$ e $f_2(\cdot)$.

Para superar esta limitação em problemas com variáveis contínuas, durante o processo de convergência de um método baseado em *Elipsóide*, é necessário que ao menos um ponto não-dominado permaneça compartilhado pela intersecção de todos os semi-espaços utilizados. Para garantir a convergência de um método baseado em *Elipsóide* para problemas vetoriais QCND, primeiramente deve-se ter $(E_0 \cap \mathcal{C}^O(\mathbf{x}_0) \cap \Omega^*) \neq \emptyset$, onde E_0 representa o elipsóide inicial. Em seguida, o próximo elipsóide E_{k+1} deve ser construído de tal forma que $(E_k \cap E_{k+1} \cap \Omega^*) \neq \emptyset$ para todo k até que um critério de parada seja satisfeito. Dado que a intersecção de dois cones $\mathcal{C}^O(\mathbf{x}_k)$ e $\mathcal{C}^O(\mathbf{x}_{k+1})$ não garante que

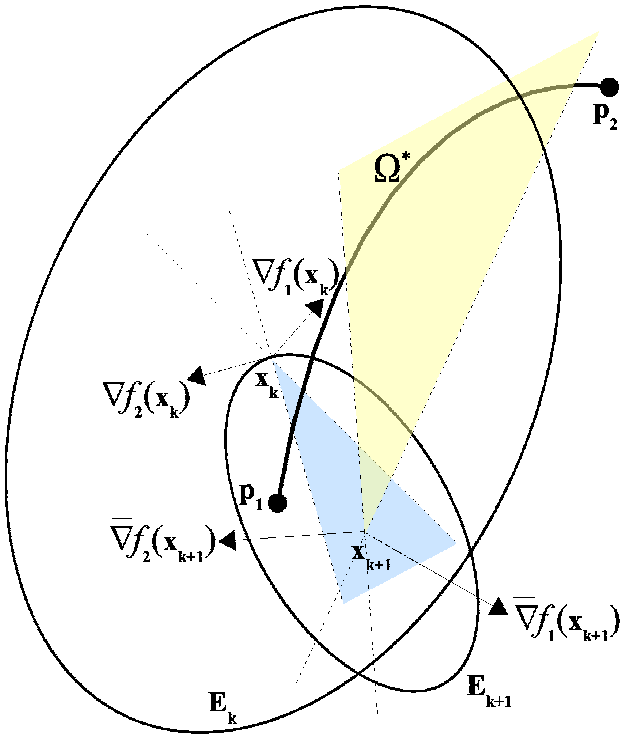


Figura 5.1: Limitação da construção de um método baseado em *Elipsóide* pela utilização direta de $\mathcal{C}^O(\mathbf{x}_k)$ e $\mathcal{C}^O(\mathbf{x}_{k+1})$.

um ponto não-dominado será preservado, a seqüência de elipsóides deve ser baseada em uma definição complementar à Definição 1 (página 53). Todavia, a intersecção dos semi-espaços que definiram os cones $\mathcal{C}^O$ nos diversos pontos da seqüência de convergência pode ser utilizada para definir um conjunto de pontos, no qual existe ao menos um ponto $\mathbf{x}^*$. A região de busca definida por este conjunto de pontos é aqui denominada de poliedro de busca (i.e., do inglês *search polyhedron*) e representa a região que deve ser contida pelo novo elipsóide para que um ponto $\mathbf{x}^*$ seja preservado. Um poliedro de busca é formalizado pela Definição 4 e pelo Teorema 4, tal como proposto em (Moura & Takahashi 2007).

Definição 4 - *Poliedro de Busca : Considere o problema definido por (5.1), a Definição 1 (página 53) e a Definição 2 (página 54). Suponha um ponto* $\mathbf{v} \notin \Omega^*$ *um ponto* $\mathbf{w} \notin \Omega^*$. *Defina um conjunto* $\Theta \triangleq \{j \mid f_j(\mathbf{w}) \geq f_j(\mathbf{v}) \ \forall \ j = 1, \ldots, m\}$. *Considere um conjunto convexo de volume finito não nulo* Γ, *tal que*

$$\Gamma \subset \mathbb{R}^n \begin{cases} (\mathcal{C}^O(\mathbf{v}) \cap \Gamma \cap \Omega^*) \notin \emptyset & \textit{se } \mathbf{v} \in \Omega; \\ (\mathcal{C}^O(\mathbf{w}) \cap \Gamma \cap \Omega^*) \notin \emptyset & \textit{se } \mathbf{w} \in \Omega \textit{ e } \mathbf{v} \notin \Omega; \\ (\Gamma \cap \Omega^*) \notin \emptyset & \textit{nos demais casos;} \end{cases} \tag{5.3}$$

O poliedro de busca é definido por:

$$P_{\mathbf{w}}(\mathbf{v},\Gamma) \triangleq \begin{cases} \Gamma \cap \mathcal{C}^O(\mathbf{w}) \cap \mathcal{C}^O(\mathbf{v}), & f(\mathbf{w}) \prec f(\mathbf{v}) \; e \; \{\mathbf{w},\mathbf{v}\} \in \Omega. \\ \Gamma \cap \mathcal{C}^O(\mathbf{w}) \cap \mathcal{C}^O(\mathbf{v}), & f(\mathbf{v}) \prec f(\mathbf{w}) \; e \; \{\mathbf{w},\mathbf{v}\} \in \Omega. \\ \Gamma \bigcap_i^{\Theta} H_-^{\bar{\nabla} f_i(\mathbf{w}),\mathbf{w}} \cap \mathcal{C}^O(\mathbf{v}), & f(\mathbf{w}) \not\prec f(\mathbf{v}), f(\mathbf{v}) \not\prec f(\mathbf{w}) \; e \; \{\mathbf{w},\mathbf{v}\} \in \Omega. \\ \Gamma \cap \mathcal{C}^O(\mathbf{w}) \cap \mathcal{C}^F(\mathbf{v}), & \mathbf{w} \in \Omega \; e \; \mathbf{v} \notin \Omega. \\ \Gamma \cap \mathcal{C}^F(\mathbf{w}) \cap \mathcal{C}^O(\mathbf{v}), & \mathbf{w} \notin \Omega \; e \; \mathbf{v} \in \Omega. \\ \Gamma \cap \mathcal{C}^F(\mathbf{w}) \cap \mathcal{C}^F(\mathbf{v}), & \{\mathbf{w},\mathbf{v}\} \notin \Omega. \end{cases} \tag{5.4}$$

Teorema 4 - *Poliedro de Busca : Um poliedro de busca formalizado pela Definição 4 garante que* $(P_{\mathbf{w}}(\mathbf{v},\Gamma) \cap \Omega^*) \neq \emptyset$.

Prova:

i) *Para os pontos* $\mathbf{w} \in \Omega \; \mathbf{v} \in \Omega \mid f(\mathbf{w}) \prec f(\mathbf{v})$, $(\mathcal{C}^O(\mathbf{v}) \cap \Gamma \cap \Omega^*) \supset (\mathcal{C}^O(\mathbf{w}) \cap \Gamma \cap \Omega^*)$. *Como por definição* $(\mathcal{C}^O(\mathbf{v}) \cap \Gamma \cap \Omega^*) \neq \emptyset$, *então* $(P_{\mathbf{w}}(\mathbf{v},\Gamma) \cap \Omega^*) \neq \emptyset$.

ii) *Para os pontos* $\mathbf{w} \in \Omega \; \mathbf{v} \in \Omega \mid f(\mathbf{v}) \prec f(\mathbf{w})$, $(\mathcal{C}^O(\mathbf{v}) \cap \Omega^*) \subset (\mathcal{C}^O(\mathbf{w}) \cap \Omega^*)$. *Como por definição* $(\mathcal{C}^O(\mathbf{v}) \cap \Gamma \cap \Omega^*) \neq \emptyset$, *então* $(P_{\mathbf{w}}(\mathbf{v},\Gamma) \cap \Omega^*) \neq \emptyset$.

iii) *Para os pontos* $\mathbf{w} \in \Omega \; \mathbf{v} \in \Omega \mid f(\mathbf{w}) \not\prec f(\mathbf{v})$ *e* $f(\mathbf{v}) \not\prec f(\mathbf{w})$, $(\mathcal{C}^O(\mathbf{v}) \cap \Omega^*) \subset \bigcap_i^{\Theta} H_-^{\bar{\nabla} f_i(\mathbf{w}),\mathbf{w}}$. *Como por definição* $(\mathcal{C}^O(\mathbf{v}) \cap \Gamma \cap \Omega^*) \neq \emptyset$, *então* $(P_{\mathbf{w}}(\mathbf{v},\Gamma) \cap \Omega^*) \neq \emptyset$.

iv) *Para os pontos* $\mathbf{w} \in \Omega \; \mathbf{v} \notin \Omega \mid f(\mathbf{w}) \not\prec f(\mathbf{v})$ *e* $f(\mathbf{v}) \not\prec f(\mathbf{w})$, $\mathcal{C}^F(\mathbf{v}) \supset \Omega^*$. *Como por definição* $(\mathcal{C}^O(\mathbf{w}) \cap \Omega^*) \neq \emptyset$, *então* $(P_{\mathbf{w}}(\mathbf{v},\Gamma) \cap \Omega^*) \neq \emptyset$.

v) *Para os pontos* $\mathbf{w} \notin \Omega \; \mathbf{v} \in \Omega \mid f(\mathbf{w}) \not\prec f(\mathbf{v})$ *e* $f(\mathbf{v}) \not\prec f(\mathbf{w})$, $(\mathcal{C}^O(\mathbf{v}) \cap \Omega^*) \subset \mathcal{C}^F(\mathbf{w})$. *Como por definição* $(\mathcal{C}^O(\mathbf{v}) \cap \Gamma \cap \Omega^*) \neq \emptyset$, *então* $(P_{\mathbf{w}}(\mathbf{v},\Gamma) \cap \Omega^*) \neq \emptyset$.

vi) *Para os pontos* $\mathbf{w} \notin \Omega \; \mathbf{v} \notin \Omega \mid f(\mathbf{w}) \not\prec f(\mathbf{v})$ *e* $f(\mathbf{v}) \not\prec f(\mathbf{w})$, $\Omega^* \subset \mathcal{C}^F(\mathbf{w})$. *Como por definição* $(\mathcal{C}^F(\mathbf{v}) \cap \Gamma \cap \Omega^*) \neq \emptyset$, *então* $(P_{\mathbf{w}}(\mathbf{v},\Gamma) \cap \Omega^*) \neq \emptyset$, *o que completa a demonstração do teorema.*

A figura 5.2 ilustra a idéia de um poliedro de busca em um problema vetorial hipotético com $n = 2$, $m = 2$ e $p = 2$, para a ocorrência de dois pontos factíveis, sendo que o ponto $\mathbf{w}$ não domina e nem é dominado por $\mathbf{v}$. Os pontos p_1 e p_2 respectivamente representam os pontos mínimos irrestritos das funções $f_1(\cdot)$ e $f_2(\cdot)$.

A partir da ótica de um método baseado em elipsóide sobre a Definição 4 e sobre o Teorema 4, o Teorema 5 definirá o novo elipsóide E_{k+1} que contém ao menos um ponto $\mathbf{x}^*$.

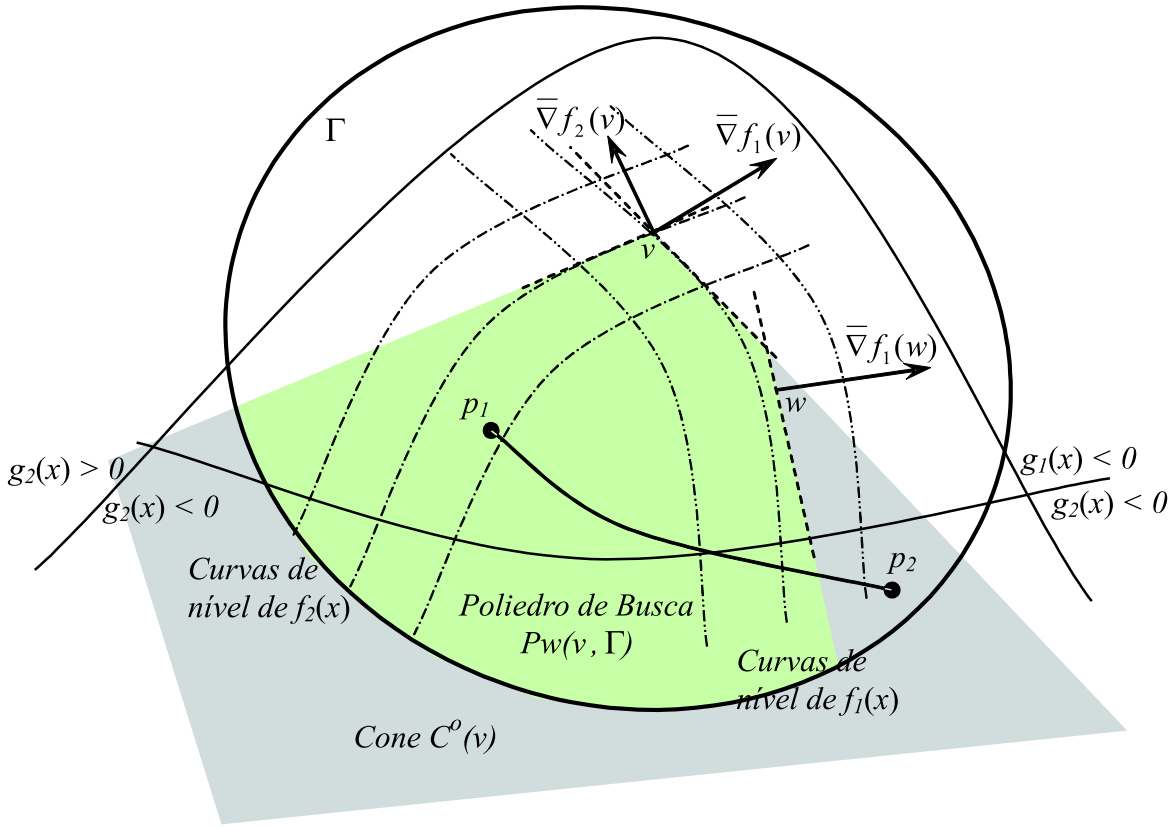


Figura 5.2: Ilustração de um poliedro de busca para **w** e **v** factíveis.

Teorema 5 - *Elipsóide dado um Poliedro de Busca : Considere o problema definido por (5.1), a Definição 4 e o Teorema 4. Defina o conjunto convexo de volume finito não nulo definido pelo elipsóide E_k. Considere o elipsóide inicial $\{E_0 \subset \mathbb{R}^n \mid E_0 \supset \Omega^*\}$. Para uma iteração qualquer k, tal que $\mathbf{x}_i \notin \Omega^* \mid i = 0, \ldots, k$, o novo elipsóide E_{k+1} para o qual necessariamente $(E_{k+1} \cap \Omega^*) \neq \emptyset$ deve ser tal que:*

$$E_{k+1} \supset P_{\mathbf{x}_k}(\mathbf{x}_s, E_k) \tag{5.5}$$

$$\text{onde } s = \begin{cases} k-1, & \nexists\, \mathbf{x}_j \notin \Omega \;\forall\; j = 0, \ldots, k-1. \\ min(\{j\}), & \exists\, \mathbf{x}_j \in \Omega \; e \;\; \nexists\, \mathbf{x}_i \in \Omega \mid f(\mathbf{x}_i) \prec f(\mathbf{x}_j) \forall\; j = 0, \ldots, k-1 \; e \; i < k. \\ i, & \exists\, \mathbf{x}_j \in \Omega \; e \; \exists\, \mathbf{x}_i \in \Omega \mid f(\mathbf{x}_i) \prec f(\mathbf{x}_j) \forall\; j = 0, \ldots, k-1 \; e \; i < k. \end{cases} \tag{5.6}$$

Prova:

i) *Para um ponto $\mathbf{x}_k \notin \Omega$ e $\mathbf{x}_s \notin \Omega$ dado que $(E_0 \supset \Omega^*) \neq \emptyset$ e que os semi-espaços $H_-^{\bar{\nabla} g_i(\mathbf{x}_j), \mathbf{x_j}}$ não excluíram qualquer ponto $\mathbf{x} \in \Omega^*$ para todo $g_i(\mathbf{x}_j) > 0 \mid i =$*

$\{1,\ldots,p\}\ \forall\ j=\{0,\ldots,k-1\}$, *então* $\Omega^* \subset P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$ *e* $(E_{k+1}\cap\Omega^*)\neq\emptyset$.

ii) *Para um ponto* $\mathbf{x}_k\in\Omega$ *e* $\mathbf{x}_s\notin\Omega$, *o Teorema 4 garante que* $(P_{\mathbf{x}_k}(\mathbf{x}_s,E_k)\cap\Omega^*)\neq\emptyset$ *uma vez que pelo item* (i) $(E_t\supset\Omega^*)\ \forall\ t=\{1,\ldots,k-1\}$ *e que* $(E_0\supset\Omega^*)$. *Então* $(E_{k+1}\cap\Omega^*)\neq\emptyset$.

iii) *Para um ponto* $\mathbf{x}_k\notin\Omega$ *e* $\mathbf{x}_s\in\Omega$, *o Teorema 4 garante que* $P_{\mathbf{x}_k}(\mathbf{x}_s,E_k)\supset(E_k\cap\Omega^*)$. *Então, desde que* $(E_k\cap\Omega^*)\neq\emptyset$, *o novo elipsóide* $(E_{k+1}\cap\Omega^*)\neq\emptyset$.

iv) *Para um ponto* $\mathbf{x}_k\in\Omega$ *e* $\mathbf{x}_s\in\Omega$ *existem duas hipóteses:*

- **a)** *Caso* $f(\mathbf{x}_s)\not\prec f(\mathbf{x}_j)\forall\ j=0,\ldots,k-1$ *e* $i<k$ *tem-se que* $\mathbf{x}_s$ *corresponde ao primeiro ponto factível da seqüência* $0,\ldots,k$. *Conseqüentemente* $(E_{s+1}\Omega^*)\neq\emptyset$ *diretamente pelo item* (ii). *Pelo Teorema 4 e pelo item* (iii) *tem-se que* $(P_{\mathbf{x}_t}(\mathbf{x}_s,E_s)\supset(E_{s+1}\Omega^*)\ \forall\ s<t<k$. *Logo, o novo elipsóide* $(E_{k+1}\cap\Omega^*)\neq\emptyset$.
- **b)** *Caso* $f(\mathbf{x}_s)\prec f(\mathbf{x}_j)\forall\ j=0,\ldots,k-1$ *e* $i<k$ *tem-se que* $\mathbf{x}_s$ *domina todos os demais pontos da seqüência* $0,\ldots,k$. *Nomeando-se* $\mathbf{x}_f$ *o primero ponto factível da seqüência* $0,\ldots,k$, *o item* (ii) *garante que* $(E_{f+1}\Omega^*)\neq\emptyset$. *Pelo Teorema 4 e pelo caso* (a) *tem-se que* $(P_{\mathbf{x}_s}(\mathbf{x}_f,E_f)\cap\Omega^*)\supseteq(E_{f+1}\Omega^*)$ *e que* $(P_{\mathbf{x}_t}(\mathbf{x}_s,E_j)\supset(E_{s+1}\Omega^*)\ \forall\ s<t<k$. *Logo, o novo elipsóide* $(E_{k+1}\cap\Omega^*)\neq\emptyset$.

v) *Por fim, embora os itens* (iii) *e* (iv) *sejam interdependentes temos que* $(P_{\mathbf{x}_k}(\mathbf{x}_s,E_k)\cap\Omega^*)\neq\emptyset$ *ou por estes itens ocorrerem após o item* (ii) *ou porque* $\mathbf{x}_0\in\Omega$ *e* $E_0\supset\Omega^*$, *o que completa a demonstração do teorema.*

Deve-se ainda observar que o elipsóide E_{k+1} mantém o mesmo subconjunto *Pareto-Ótimo* $(E_k\cap\Omega^*)$ quando E_{k+1} foi definido por (i) e (iii), e que preserva parte do subconjunto *Pareto-Ótimo* $(E_j\cap\Omega^*))$, tal que $j=max(\{0,\ldots,k-1\})\mid\mathbf{x}_j\in\Omega$, quando E_{k+1} foi definido por (ii) e (iv).

Nota 11 *: Todo poliedro de busca define um conjunto convexo de volume finito não nulo.*

5.2 Formalização do *Poliedro de Busca com Histórico de Intersecções*

Uma vez formalizado o poliedro de busca, também deve ser definido o conjunto resultante da intersecção de diversos poliedros de busca de iterações passadas, aqui denominado poliedro de busca com histórico de intersecções (i.e., do inglês *historical intersection search polyhedron*). A Definição e o Corolário formalizam este conceito.

Definição 5 - *Poliedro de Busca com Histórico de Intersecções : Considere o problema definido por (5.1) e a Definição 4. Defina um conjunto* $\zeta \triangleq \{[\mathbf{x}_0, \ldots, \mathbf{x}_k]\}$ *e suponha* $(\zeta \cap \Omega^*) = \emptyset$*. Calcule o índice s por (5.6) e suponha* $(P_{\mathbf{x}_s}(\mathbf{x}_s, \Gamma) \cap \Omega^*) = \emptyset$*. Define-se o poliedro* $P_\zeta(\mathbf{x}_k, \Gamma)$ *pela intersecção de todos os poliedros de busca dos pontos do conjunto* ζ*, tal como apresentado a seguir.*

$$P_\zeta(\mathbf{x}_s, \Gamma) \triangleq \{ \bigcap_j^\zeta P_{\mathbf{x}_j}(\mathbf{x}_s, \Gamma) \} \tag{5.7}$$

Corolário 3 *: A partir da Definição 5, o novo elipsóide* E_{k+1}*, para o qual garantidamente* $(E_{k+1} \cap \Omega^*) \neq \emptyset$*, pode ser definido por:*

$$E_{k+1} \supset P_\zeta(\mathbf{x}_s, \Gamma) \tag{5.8}$$

Nota 12 *: Todo poliedro de busca com histórico de intersecções também define um conjunto convexo de volume finito não nulo.*

A partir das Definições 4 e 5 e do Teorema 5 e do Corolário 3, temos os seguintes Lemas:

Lema 3 *: Considere o problema definido por 5.1, a Definição 5, o Corolário 3 e uma região limitada pelo elipsóide* E_k *definido por (5.3). Então, a menor região de interesse de busca,* R^**, que pode ser determinada pela avaliação de* $f(\cdot)$ *e* $g(\cdot)$ *nos pontos* $\mathbf{x}_k$ *e* $\mathbf{x}_s$ *é definida por:*

$$R^* \iff P_\zeta(\mathbf{x}_s, E_k) \tag{5.9}$$

Prova:
Diretamente do Teorema 5 e do Corolário 3.

Lema 4 *: Considere o caso mono-objetivo,* $m = 1$*, derivado do problema definido por por 5.1 para o Lema 3. Considere o conjunto* $\Upsilon \triangleq \{j \ \forall \ \mathbf{x}_j \in \Omega \ \forall \ j \in \{1, \ldots, k\}\}$ *e um conjunto* $\Psi \triangleq \{i \ \forall \ \mathbf{x}_i \notin \Omega \ \forall \ i \in \{1, \ldots, k\}\}$ *Então, a menor região de interesse de busca,* R^**, que pode ser determinada pela avaliação de* $f(\cdot)$ *e* $g(\cdot)$ *nos pontos do conjunto* ζ *é definida por:*

$$R^* \triangleq \{ \ E_k \bigcap_j^\Upsilon \mathcal{C}^O(\mathbf{x}_j) \bigcap_i^\Psi \mathcal{C}^F(\mathbf{x}_i) \ \} \tag{5.10}$$

Prova:

Para o caso mono-objetivo, o Teorema 4 se restringe aos casos (i) e (iv). Assim, para $\mathbf{x}_k \in \Omega$*, tem-se que* $P_{\mathbf{x}_k}(\mathbf{x}_0, \Gamma) \Longleftrightarrow \mathcal{C}^O(\mathbf{x}_k)\ \forall\ k \geq 0$*. Para* $\mathbf{x}_k \notin \Omega$*, tem-se que* $P_{\mathbf{x}_k}(\mathbf{x}_0, \Gamma) \Longleftrightarrow \mathcal{C}^F(\mathbf{x}_k)\ \forall\ k \geq 0$*. Logo, a a menor região de interesse de busca descrita por (5.9) torna-se (5.10), o que completa a demonstração.*

Um exemplo da ilustração geométrica do Lema 4 é exibido na figura 5.3 para os pontos $\mathbf{x}_1$ *e* $\mathbf{x}_2 \in \Omega$, um ponto $\{\mathbf{x}_3 \notin \Omega\}$ e um conjunto convexo de volume finito Γ. Neste caso, verifica-se que o poliedro de busca é definido pela intersecção $\mathcal{C}^O(\mathbf{x}_1) \cap \mathcal{C}^O(\mathbf{x}_1) \cap \mathcal{C}^F(\mathbf{x}_3) \cap \Gamma$ e representa a menor região de interesse de busca resultante da avaliação de $f(\cdot)$ e $g(\cdot)$ em $\{\ \mathbf{x}_1\ \mathbf{x}_2$ *e* $\mathbf{x}_3\ \}$.

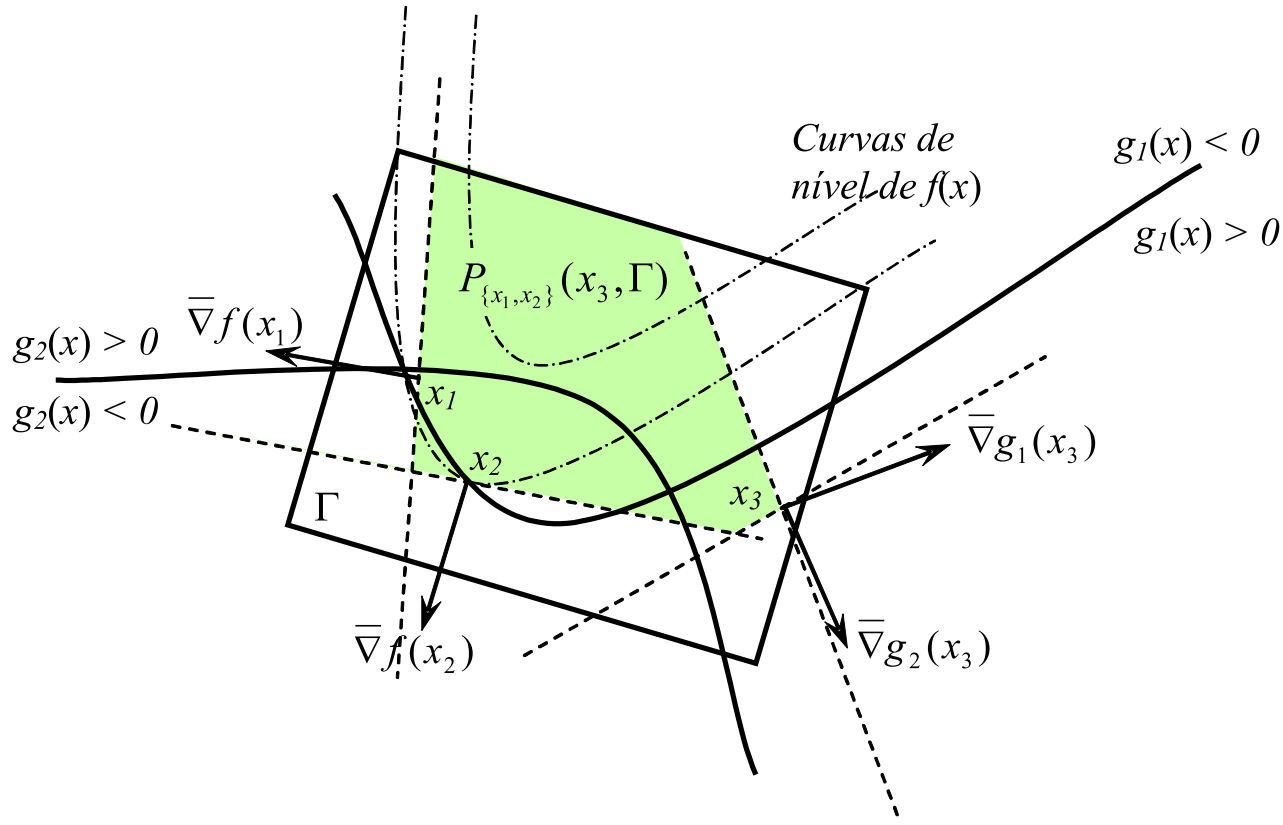


Figura 5.3: Ilustração de um poliedro de busca resultante da intersecção $\mathcal{C}^O(\mathbf{x}_1) \cap \mathcal{C}^O(\mathbf{x}_2) \cap \mathcal{C}^F(\mathbf{x}_3) \cap \Gamma$ para o caso mono-objetivo.

Como pode ser verificado, os métodos *Corte Paralelo*, *Corte em Cunha*, *Dois Subgradientes Sucessivos* e *Cone-Elipsoidal* apresentados no capítulo 3 são, todos eles, baseados em casos particulares do aqui proposto poliedro de busca com histórico de intersecções. Para o caso do *Corte Paralelo* tem-se que os cones definidos em $\mathbf{v}$ e $\mathbf{w}$ são substituídos por simples hiperplanos e os problemas limitados aos casos escalares. Já para o *Corte em Cunha*, utiliza-se um cone composto por apenas dois hiperplanos definidos somente

no ponto $\mathbf{w}$ e os problemas limitados aos casos escalares. Quanto aos *Dois Subgradientes Sucessivos*, este é composto pelo *Corte Paralelo*, pelo *Corte em Cunha* e pelo *Deep Cut* aplicados exclusivamente aos problemas escalares. Neste caso, para o *Deep Cut* tem-se que os cones definidos em $\mathbf{v}$ e $\mathbf{w}$ são substituídos por simples hiperplanos e $f(\mathbf{w}) \prec f(\mathbf{v})$. Por fim, no *Cone-Elipsoidal* a definição se restringe, erroneamente, ao caso em que $f(\mathbf{w}) \prec f(\mathbf{v})\, e\, \{\mathbf{w}, \mathbf{v}\} \in \Omega$.

5.3 Conclusões do Capítulo

Neste Capítulo foram propostos os fundamentos para os métodos *Poliedro-Elipsoidais* para a solução de problemas escalares e vetoriais QCND. No seu desenvolvimento foram propostos Teoremas, Lemas e Corolários e Definições que formalizam os conceitos do poliedro de busca e do poliedro de busca com histórico de intersecções, bem como permitem determinar um novo elipsóide E_{k+1} que certamente preservará parte do conjunto *Pareto-Ótimo*.

No capítulo seguinte serão detalhados os algoritmos que se fundamentam nas definições, aqui apresentadas, para implementar a família de métodos *Poliedro-Elipsoidais*.

Capítulo 6

Família dos Métodos Poliedro-Elipsoidais para Problemas Contínuas

Neste capítulo serão propostos os algoritmos que constituem a família dos métodos *Poliedro-Elipsoidais* , a partir dos Teoremas e Corolários definidos no capítulo 5, para a solução de problemas escalares e vetoriais com variáveis puramente contínuas, quasi-convexos e não necessariamente diferenciáveis.

No seu desenvolvimento, são apresentadas as implementações do método *Poliedro-Elipsoidal* por Hiperesfera, do método *Poliedro-Elipsoidal* por Seqüência de Cortes Profundos e Rasos, e do método *Historical Intersection Search Polyhedron Ellipsoid Method*.

Outros detalhes sobre estes temas podem ser conseguidos na referência (Moura & Takahashi 2007).

6.1 Métodos *Elipsoidais* com Poliedro de Busca

A formalização do poliedro de busca, vide a Definição 4 (página 75) e o Teorema 4 (página 76), permite gerar um método baseado no algoritmo *Elipsoidal*, tal como demonstrado no Teorema 5 (página 77). A justificativa para a proposição de um novo método baseado no algoritmo *Elipsoidal* se deve à perspectiva de que a utilização do poliedro de busca $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$ poderá produzir uma aceleração no processo de convergência para a solução. Em função da provável utilização de mais de um semi-espaço na construção de $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$ ter-se-á para estes métodos um maior esforço computacional ECAO tendo como referência o algoritmo de Shor. Por consequinte, o poliedro de busca deve ser utilizado principalmente para as classes de problemas nas quais o EAIP é maior que o ECAO, tal como

apresentado na seção 1.2.2.

Deve-se ainda enfatizar que, mesmo com a utilização da mesma região de interesse de busca $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$, é possível gerar mais de uma variante de um método baseado em *Elipsóide*. Isto ocorre porque não existe uma fórmula analítica para definir um elipsóide $E_{k+1} \supset P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$. Assim, cada maneira de se calcular $E_{k+1} \supset P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$ definirá uma variante de um método baseado em *Elipsóide*.

A seguir são apresentadas algumas abordagens para o cálculo de E_{k+1} a partir do poliedro de busca $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$.

O problema (1.8) será aqui tratado e é representado abaixo para a comodidade do leitor:

$$min\ f(\mathbf{x}) \tag{6.1}$$

$$s.a.\ \Omega^* = \{\mathbf{x}^* \in \Omega \mid \nexists\, \mathbf{x} \in \Omega\ tal\ que\ f(\mathbf{x}) \leq f(\mathbf{x}^*)\ e\ f(\mathbf{x}) \neq f(\mathbf{x}^*)\}$$

$$\Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}^n\}$$

onde, $f(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^m$ e $g(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^p$.

6.1.1 Método *Poliedro-Elipsoidal* por Hiperesfera: $MPEH$

A primeira proposta de utilizar o conceito do poliedro de busca para acelerar a redução do volume do novo elipsóide E_{k+1}, consiste em utilizar as coordenadas no qual o elipsóide E_k é uma hiperesfera para calcular o novo elipsóide. Para tanto, faz-se necessário uma simplificação do poliedro de busca para um cone polar determinado por hiperplanos avaliados em um único ponto, tal como apresentado a seguir:

Definição 6 - *Cone de Busca : Considere da Definição 4 (página 75) os conjuntos* Γ, Θ *e* Ψ, *bem como os pontos* $\mathbf{v}$ *e* $\mathbf{w}$. *O cone de busca é definido por:*

$$C_{\mathbf{w}}(\mathbf{v}, \Gamma) \triangleq \begin{cases} \Gamma \cap \mathcal{C}^O(\mathbf{w}), & f(\mathbf{w}) \prec f(\mathbf{v})\ e\ \{\mathbf{w}, \mathbf{v}\} \in \Omega. \\ \Gamma \cap \mathcal{C}^O(\mathbf{w}), & f(\mathbf{v}) \prec f(\mathbf{w})\ e\ \{\mathbf{w}, \mathbf{v}\} \in \Omega. \\ \Gamma \bigcap_i^{\Theta} H_-^{\bar{\nabla} f_i(\mathbf{w}), \mathbf{w}}, & f(\mathbf{w}) \not\prec f(\mathbf{v}), f(\mathbf{v}) \not\prec f(\mathbf{w})\ e\ \{\mathbf{w}, \mathbf{v}\} \in \Omega. \\ \Gamma \cap \mathcal{C}^F(\mathbf{w}), & \mathbf{w} \in \Omega\ e\ \mathbf{v} \notin \Omega. \\ \Gamma \cap \mathcal{C}^F(\mathbf{w}), & \mathbf{w} \notin \Omega. \end{cases} \tag{6.2}$$

Corolário 4 - *Cone de Busca : Um cone de busca formalizado pela Definição 6 garante que* $(P_{\mathbf{w}}(\mathbf{v}, \Gamma) \cap \Omega^*) \neq \emptyset$.

Nota 13 *: O cone de busca* $C_{\mathbf{w}}(\mathbf{v}, \Gamma)$ *não necessariamente corresponde à mínima região de interesse de busca* R^* *definida por (5.9).*

O conceito básico deste método é calcular uma esfera que contenha o cone de busca $C_{\mathbf{x}_k}(\mathbf{x_s}, E_k)$. O processo inicia-se com o cálculo da matriz M_{Ck} que caracteriza o cone de busca na iteração k corrente. A matriz M_{Ck} é definida por (6.3) e suas colunas correspondem aos vetores $\bar{\nabla} f_i$ ou $\bar{\nabla} g_j$ que compõem a intersecção de semi-espaços que definem o cone de busca.

$$C_{\mathbf{x}_k}(\mathbf{x_s}, E_k) \iff M_{Ck}^T(\mathbf{x} - \mathbf{x}_k) \geq 0 \tag{6.3}$$

Quando a matriz M_{Ck} possui posto maior ou igual que a dimensão do problema, é possível calcular um elipsóide S_{k+1}, tal que $vol(S_{k+1}) < vol(\hat{E}_{k+1})$. Neste caso, $\hat{E}_{k+1}$ corresponde a um elipsóide que poderia ser obtido com o uso de quaisquer dos dois vetores que definem as colunas de M_{Ck}, individualmente, empregando-se (2.11) referente ao método *Elipsoidal* de Shor.

Para o cálculo de S_{k+1}, é realizada uma transformação de coordenadas onde o elipsóide E_k é uma hiperesfera de raio unitário, representada por $\bar{E}_k$. Nas novas coordenadas o cone $\bar{C}_{\mathbf{x}_k}(\mathbf{x_s}, E_k)$ representa o cone de busca transformado. Uma hiperesfera $\bar{S}_{k+1} \supset \bar{C}_{\mathbf{x}_k}(\mathbf{x_s}, E_k)$ é então calculada, e esta hiperesfera transformada para as coordenadas originais definirá o elipsóide S_{k+1}.

Quando a matriz M_{Ck} possui posto menor que a dimensão do problema ou quando $vol(S_{k+1}) > vol(\hat{E}_{k+1})$ o novo elipsóide E_{k+1} será $\hat{E}_{k+1}$. Caso contrário o novo elipsóide E_{k+1} será S_{k+1}.

Este método, portanto, pode ser entendido como uma generalização do método de *Cortes em Cunha*, apresentado no capítulo 3 e aplicável quando da existência de dois subgradientes.

A estruturação geral do algoritmo que implementa o método *Poliedro-Elipsoidal* por Hiperesfera, MPE-H, é exibida a seguir. Através da análise deste algoritmo pode-se verificar que sua complexidade permanece equivalente à complexidade polinomial do algoritmo *Elipsoidal* de Shor, já que a transformação de coordenadas e o cálculo da hiperesfera $\bar{S}_{k+1}$ são de complexidade polinomial. Neste algoritmo, a função $Rt(n) = (n/(n+1))(n^2/(n^2-1))^{(n-1)/2}$ representa a relação de redução do volume do elipsóide

entre duas iterações sucessivas para o algoritmo *Elipsoidal* de Shor. Para facilitar a representação do algoritmo, define-se que $f(\emptyset) = \infty_+$ e $C_{\mathbf{v}}(\emptyset, \Gamma) = C_{\mathbf{v}}(\mathbf{v}, \Gamma)$. Pelo mesmo motivo foi excluído o critério de parada $\mathbf{x}_k \in \Omega^*$.

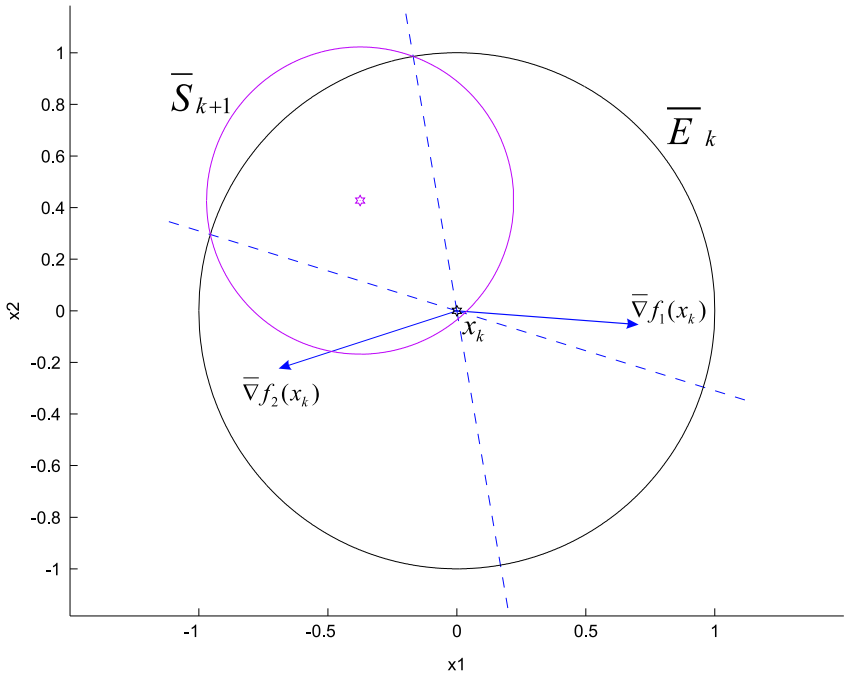
Algorithm 7 *Poliedro-Elipsoidal* por Hiperesfera

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^*$ será procurada de acordo com o problema definido por (6.1).

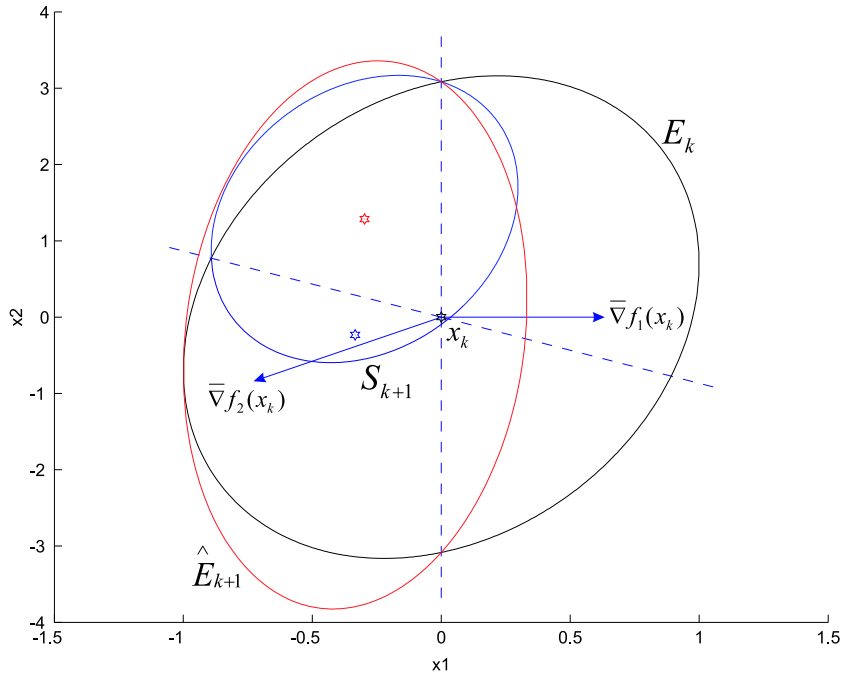
Output: A melhor solução factível contínua $\mathbf{x}^f$ encontrada. Para um problema infactível ou para $\mathbf{x}^* \notin E_0$ ter-se-á $\mathbf{x}^f = \emptyset$.

1: **function** $MPEH(E_0)$
2: $k \leftarrow 0$
3: $\mathbf{x}^f \leftarrow \emptyset$ ▷ Zera o melhor ponto factível
4: **loop**
5: $M_{Ck} \leftarrow C_{\mathbf{x}_k}(\mathbf{x}^{\mathbf{f}}, E_k)$ ▷ Calcula o cone de busca e a matriz M_{Ck}.
6: **if** $\mathbf{x}_k \in \Omega^*$ e $f(\mathbf{x}_k) \prec f(\mathbf{x}^f)$ **then**
7: $\mathbf{x}^f \leftarrow \mathbf{x}_k$ ▷ Novo melhor ponto factível.
8: **end if**
9: **if** $rank(M_{Ck}) \geq n$ **then** ▷ $C_{\mathbf{x}_k}(\mathbf{x_f}, E_k)$.
10: $\bar{S}_{k+1} \leftarrow (\bar{E}_k \cap \bar{C}_{\mathbf{x}_k}(\mathbf{x_f}, E_k))$ ▷ Hiperesfera nas coordenadas transformadas.
11: **if** $vol(\bar{S}_{k+1})/vol(\bar{E}_k) > Rt(n)$ **then**
12: **go to** linha 22
13: **else**
14: $S_{k+1} \leftarrow Q_k \bar{S}_{k+1}$ ▷ Retorna às coordenadas iniciais.
15: $E_{k+1} = S_{k+1}$
16: **go to** linha 24
17: **end if**
18: **else**
19: **go to** linha 22
20: **end if**
21: $\bar{\nabla}_k = \sum_{j=1}^{n} \frac{M_{Ck}(:,j)}{\| M_{Ck}(:,j) \|}$ ▷ Garante $\mathbf{x}_{k+1}$ dentro de $\bar{C}_{\mathbf{x}_k}(\mathbf{x_f}, E_k)$.
22: $E_{k+1} \leftarrow (E_k \cap H_-^{\bar{\nabla}_k, \mathbf{x}_k})$ ▷ Vide (2.11)
23: **if** vol $(E_{k+1}) \simeq 0$ ou $vol(E_{k+1}) \simeq vol(E_k)$ **then**
24: **return** $\mathbf{x}^f$ ▷ Termine o algoritmo.
25: **end if**
26: $k \leftarrow k + 1$
27: **end loop**
28: **end function**

A figura 6.1 exemplifica a utilização do método $MPEH$. A subfigura 6.1.a exibe, tanto para as coordenadas transformadas quanto para as coordenadas originais, uma iteração onde o elipsóide $\bar{S}_{k+1}$ possui volume inferior ao elipsóide $\hat{E}_{k+1}$. Já a subfigura 6.1.b exibe a situação oposta, na qual o elipsóide $\bar{S}_{k+1}$ possui volume superior ao elipsóide $\hat{E}_{k+1}$.

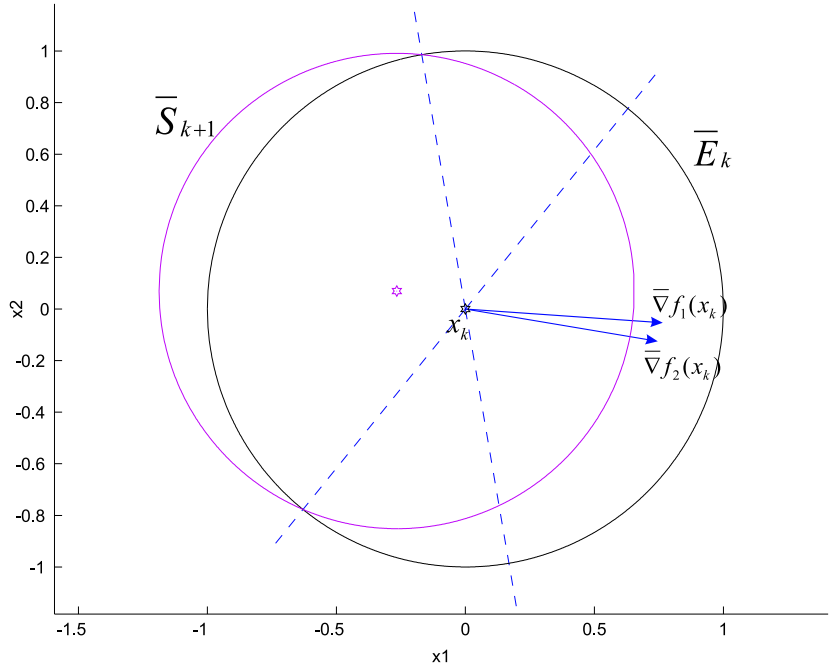


a1) Esfera $\bar{S}_{k+1}$ nas coordenadas transformadas.

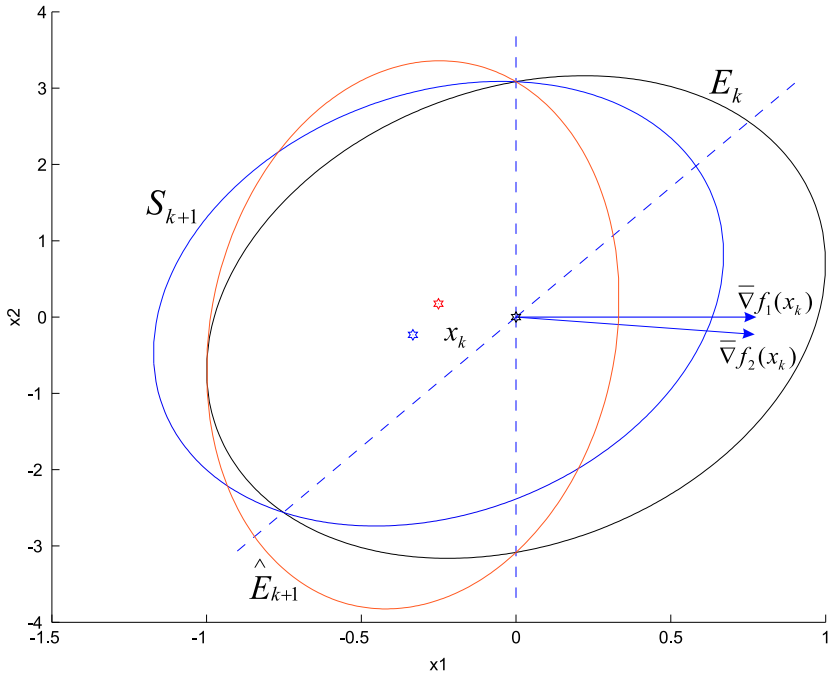


a2) Elipsóide S_{k+1} nas coordenadas originais.

a) Iteração com $vol(S_{k+1}) < vol(\hat{E}_{k+1})$.



b1) Esfera $\bar{S}_{k+1}$ nas coordenadas transformadas.



b2) Elipsóide S_{k+1} nas coordenadas originais.

b) Iteração com $vol(S_{k+1}) > vol(\hat{E}_{k+1})$.

Figura 6.1: Ilustração de duas iterações de cálculo do método *Poliedro-Elipsoidal* por Hiperesfera.

6.1.2 Método *Poliedro-Elipsoidal* por Seqüência de Cortes Profundos e Rasos: $MPESC$

Uma segunda opção de implementação do conceito do poliedro de busca consiste em utilizar uma seqüência de cortes profundos e cortes rasos (i.e., do inglês *Shallow Cuts*) definidos por (2.14) para calcular E_{k+1}. Esta seqüência utilizará sucessivamente os vetores suporte do poliedro de busca, ou seja, os vetores que definem as faces deste poliedro. Embora o uso destes cortes rasos seja pouco expressivo na literatura para o cálculo de um novo elipsóide, dado que seu volume contém sempre mais da metade do elipsóide atual, no contexto do poliedro de busca estes cortes passam a ser de grande utilidade. Isto ocorre porque, durante a seqüência de cortes a maior parte destes cortes será de cortes rasos. Como ganho adicional tem-se que o algoritmo resultante é de fácil compreensão e bom desempenho.

Para a construção do algoritmo $MPESC$, faz-se necessária a seguinte definição.

Definição 7 - *Matrizes M_{Pk} e M_{Xk} : Considere pela Definição 4 (página 75) um poliedro de busca $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$. M_{Pk} corresponde a uma matriz cujas colunas são os vetores suporte de $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$. M_{Xk} corresponde a uma matriz de mesma dimensão de M_{Pk}, tal que*

$$M_{Xk} \triangleq \begin{cases} M_{Xk}(:,i) = \mathbf{x}_k, & M_{Pk}(:,i) \text{ corresponde a um subdiferencial calculado em } \mathbf{x}_k. \\ M_{Xk}(:,j) = \mathbf{x}_s, & M_{Pk}(:,j) \text{ corresponde a um subdiferencial calculado em } \mathbf{x}_s. \end{cases}$$

Por esta definição, tem-se que:

$$P_{\mathbf{x}_k}(\mathbf{x}_s, E_k) \Longleftrightarrow (E_k \bigcap_{i=1}^{col(M_{Pk})} H_-^{M_{Pk}(:,i), M_{Xk}(:,i)}) \tag{6.4}$$

A estruturação geral do algoritmo Seqüência de Cortes Profundos e Rasos, $SCPR$, é apresentada no algoritmo 8. Através da análise deste algoritmo pode-se verificar que sua complexidade corresponde exatamente à complexidade do algoritmo *Elipsoidal* de Shor.

Como pode ser deduzido, a inclusão do teste $\alpha < -1/n$ existe para impedir que o algoritmo tente calcular um novo elipsóide cujo volume não sofrerá redução. Outro ponto a se observar é a utilização de uma função de fatorização sobre a matriz Q_{k+1}. Tal como descrito na seção 2.3.4, o processo de fatorização irá aumentar a estabilidade numérica do algoritmo e reduzir a perda de positividade de Q_{k+1} por erros de arredondamento. Dado que nos testes realizados, vide figura 2.4, observou-se um desempenho equivalente para a fatorização de *Cholesky* e a fatorização UDU, qualquer um destes métodos pode ser utilizado. Como comentário final, embora o teste $vol(E_{k+1}) \simeq vol(E_k)$ não seja necessário

do ponto de vista teórico, na implementação prática do algoritmo ele será muito útil. Este fato decorre da desaceleração da taxa de redução do volume $vol(E_{k+1})/vol(E_k)$ com o passar das iterações. Conseqüentemente, o teste $vol(E_{k+1}) \simeq vol(E_k)$ permite definir se o cálculo de um novo elipsóide será justificado pela redução de volume que ele proporcionará.

O Lema 5 demonstrará que o elipsóide E_{k+1} obtido pela função $SCPR(E_k, M_{Pk}, M_{Xk})$, é tal que $vol(E_{k+1}) \leq \hat{E}_{k+1}$. Neste caso, $\hat{E}_{k+1}$ corresponde a um elipsóide que poderia ser obtido com o uso de quaisquer dos dois vetores suporte de $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$, individualmente, empregando-se (2.11) referente ao método *Elipsoidal* de Shor.

Lema 5 - *Sucessivos Cortes Profundos e Rasos : Considere um elipsóide E_k como um conjunto convexo de volume finito não nulo e o poliedro de busca $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$ definido por (4). Calcule as matrizes M_{Pk} e M_{Xk} definidas por $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$. O elipsóide $E_{k+1} = SCPR(E_k, M_{Pk}, M_{Xk})$ corresponde à dilatação n do máximo elipsóide contido em $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$.*
Prova:
O algoritmo 8 comprimirá o elipsóide E_k através de (2.14) até que a distância de todos os hiperplanos $H^{M_{Pk}(:,j), M_{Xk}(:,j)}$ estejam a uma distância $1/n$, nas coordenadas onde o elipsóide E_{k+1} é uma hiperesfera de raio unitário. Conseqüentemente, a contração n da hiperesfera de raio unitário está contida no poliedro de busca $\bar{P}_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$, que corresponde a $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$ nas coordenadas onde E_{k+1} é uma hiperesfera de raio unitário.

Corolário 5 *Quando $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k) \subset E_k$ e é um simplex, o elipsóide E_{k+1} resultante da função $SCPR(E_k, M_{Pk}, M_{Xk})$ será o elipsóide de volume mínimo que contém $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$.*

A figura 6.2 exemplifica em seis passos a utilização do algoritmo 8 do processo de cálculo do elipsóide E_{k+1}. Como pode ser visto na subfigura 6.2.a, existem dois subgradientes que definem o poliedro de busca. Dado que o elipsóide inicial E_k é uma hiperesfera e os vetores suporte do poliedro de busca definem $n-1$ lados de um tetraedro regular, o elipsóide E_{k+1}^{min} representa a hiperesfera que corresponde ao menor elipsóide que contém o tetraedro. Ao se utilizar o primeiro hiperplano $H^{\bar{\nabla} f_1(\mathbf{x}_k), \mathbf{x}_k}$ para realizar o corte de E_k, é gerada a primeira estimativa do novo elipsóide E_{k+1}. Em seguida na subfigura 6.2.b exibe o elipsóide $\widehat{E}_k$ como a representação da estimativa E_{k+1} do passo anterior. Este elipsóide $\widehat{E}_k$ será cortado pelo segundo hiperplano $H^{\bar{\nabla} f_2(\mathbf{x}_k), \mathbf{x}_k}$ e é gerada a segunda estimativa do novo elipsóide E_{k+1}. Prossegue-se a seqüência de geração dos elipsóides E_{k+1} e $\widehat{E}_k$, até que estes elipsóides convirjam para um único elipsóide, como mostrado na subfigura 6.2.f. Uma alternativa para que o elipsóide E_{k+1} seja sempre mínimo consiste em adicionar como funções-restrição o hiperparalelepípedo definido pelos limites superiores e inferiores das variáveis do problema. Assim sendo, nunca haverá um face do poliedro definida por um setor do elipsóide, dado que as faces do poliedro estarão sempre contidas no elipsóide

Algorithm 8 Seqüência de Cortes Profundos e Rasos

Input: Um elipsóide inicial E_k, uma matriz M_{Pk} e uma matriz M_{Xk}.

Output: O elipsoide E_{k+1}.

function $SCPR(E_k, M_{Pk}, M_{Xk})$
  $E_{k+1} \leftarrow E_k$
  loop
    $j \leftarrow 1$
    $\alpha \leftarrow -1/n$
    $i \leftarrow 0$
    repeat ▷ Cálculo de α.
      $\alpha_j \leftarrow M_{Pk}(:,j)^T \mathbf{x}_{k+1} - M_{Pk}(:,j)^T M_{Xk}(:,j)/(M_{Pk}(:,j)^T Q_k M_{Pk}(:,j))^{1/2}$
      if $\alpha_j < \alpha$ **then**
        $\alpha \leftarrow \alpha_j$ ▷ Melhor Corte.
        $i \leftarrow j$
      end if
      $j \leftarrow j + 1$
    until $j >$ número colunas M_{Pk}
    if $\alpha < -1/n$ **then**
      return E_{k+1} ▷ Termine o algoritmo.
    else
      $E_{k+1} \leftarrow (E_k \cap H_-^{M_{Ck}(:,i), M_{Xk}(:,i)})$ ▷ Vide (2.14)
      $Q_{k+1} \leftarrow Fatorizao(Q_{k+1})$ ▷ Vide (2.18) e (2.20)
    end if
    if $vol(E_{k+1}) \simeq vol(E_k)$ **then**
      return $\mathbf{x}^f$ ▷ Termine o algoritmo.
    end if
    $E_k \leftarrow E_{k+1}$
  end loop
end function

corrente. Logo, na seqüência de poliedros de busca todo poliedro da seqüência será um simplex.

O algoritmo 9 exibe a estruturação geral do método de *Poliedro-Elipsoidal* por Seqüência de Cortes Profundos e Rasos, $MPESC$. Tal como definido para o algoritmo 8, considera-se que M_{Pk} e M_{Xk} são definidas pelo poliedro de busca $P_{\mathbf{x}_k}(\mathbf{x}_s$. Para facilitar a representação do algoritmo, define-se que $f(\emptyset) = \infty_+$ e $C_{\mathbf{v}}(\emptyset, \Gamma) = C_{\mathbf{v}}(\mathbf{v}, \Gamma)$. Pelo mesmo motivo foi excluído o critério de parada $\mathbf{x}_k \in \Omega^*$. Deve-se ainda observar que, embora mais complexo, este algoritmo mantém pelo menos a mesma classe de complexidade polinomial do algoritmo *Elipsoidal* de Shor. Isto ocorre porque a cada iteração foi acrescida a execução do algoritmo 8 e este possui complexidade igual à complexidade do algoritmo *Elipsoidal* de Shor e o número de iterações é menor ou igual ao deste.

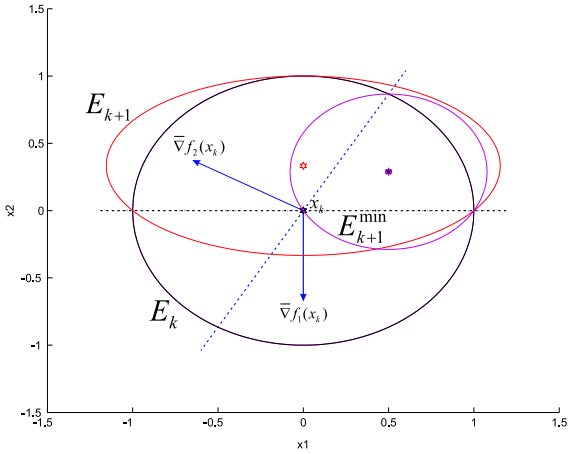
Algorithm 9 *Poliedro-Elipsoidal* por Seqüência de Cortes Profundos e Rasos

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^*$ será procurada de acordo com o problema definido por (6.1).

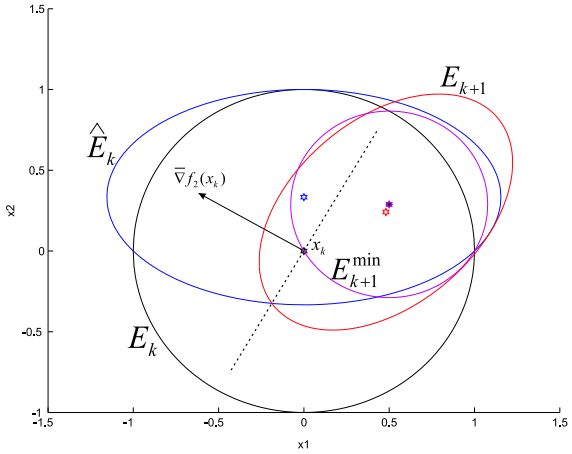
Output: A melhor solução factível contínua $\mathbf{x}^f$ encontrada. Para um problema infactível ou para $\mathbf{x}^* \notin E_0$ ter-se-á $\mathbf{x}^f = \emptyset$.

1: **function** $Shor(E_0)$
2: $k \leftarrow 0$
3: $\mathbf{x}^f \leftarrow \emptyset$ ▷ Zere a melhor solução factível.
4: **loop**
5: $P_{\mathbf{x}_k}(\mathbf{x}^\mathbf{f}, E_k)$ ▷ Calcula o poliedro de busca a partir de $\mathbf{x}^\mathbf{f}, E_k, f(\cdot)$ e $g(\cdot)$ por (5.4).
6: $M_{Pk} \leftarrow P_{\mathbf{x}_k}(\mathbf{x}^\mathbf{f}, E_k)$ ▷ Vide (6.4).
7: $M_{Xk} \leftarrow P_{\mathbf{x}_k}(\mathbf{x}^\mathbf{f}, E_k)$ ▷ Vide (6.4).
8: **if** $\mathbf{x}_k \in \Omega$ e $f(\mathbf{x}_k) \prec f(\mathbf{x}^f)$ **then**
9: $\mathbf{x}^f \leftarrow \mathbf{x}_k$ ▷ Novo melhor ponto factível.
10: **end if**
11: $E_{k+1} \leftarrow SCPR(E_k, M_{Pk}, M_{Xk})$ ▷ Vide Alg.(8)
12: **if** vol $(E_{k+1}) \simeq 0$ ou $vol(E_{k+1}) \simeq vol(E_k)$ **then**
13: **return** $\mathbf{x}^f$ ▷ Termine o algoritmo.
14: **end if**
15: $k \leftarrow k + 1$
16: **end loop**
17: **end function**

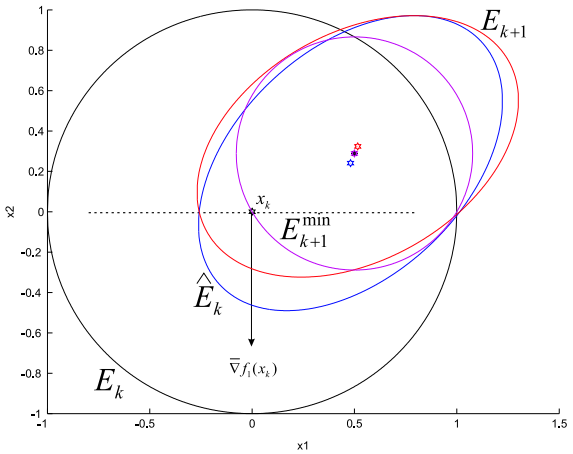
Comparando-se o algoritmo $MPESC$ com o algoritmo *Dois Subgradientes Sucessivos* verifica-se que este último pode ser considerado como um caso particular do primeiro. Isto porque o algoritmo $MPESC$ será sempre capaz de produzir uma região de busca igual ou menor à definida no algoritmo *Dois Subgradientes Sucessivos*. Para o caso específico de um problema com variáveis contínuas, escalar irrestrito, as regiões de busca definidas a cada iteração por ambos os métodos serão quase sempre iguais. A diferença ocorrerá quando os vetores subdiferenciais definirem um corte com vetores não paralelos, sendo que o elipsóide gerado pelo algoritmo $MPESC$ será o de menor volume. Para este tipo de problema, deve-se observar que ambos os métodos utilizarão sempre um ou dois semi-espaços para definir o elipsóide E_{k+1}. Como observação final, ressalta-se que para este tipo de problemas irrestritos, ou para aqueles de baixa dimensão e complexidade, é provável que o algoritmo *Dois Subgradientes Sucessivos* apresente menor tempo de convergência, dada a utilização de equações analíticas para o cálculo do E_{k+1} em contraposição à utilização da função $SCPR$.



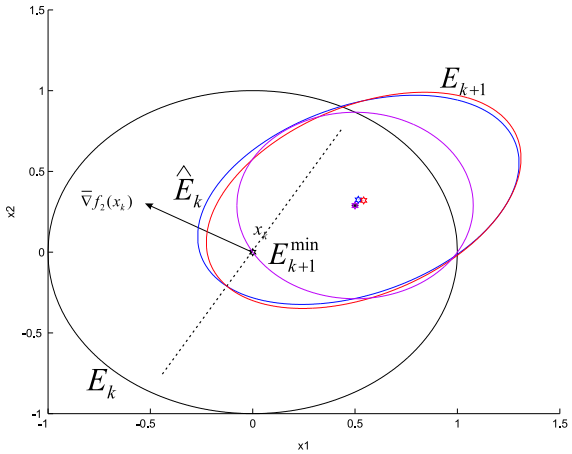
a) Corte do Passo 1.



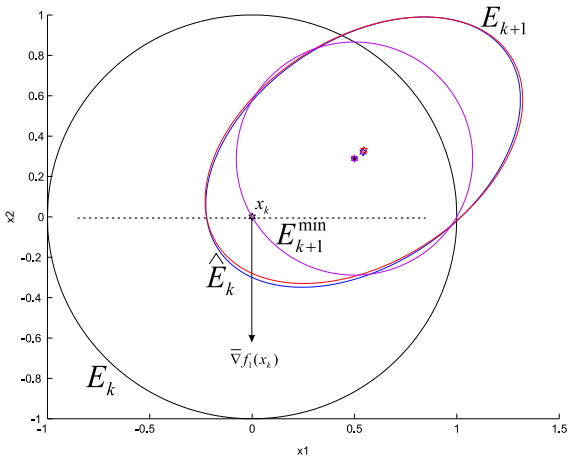
b) Corte do Passo 2.



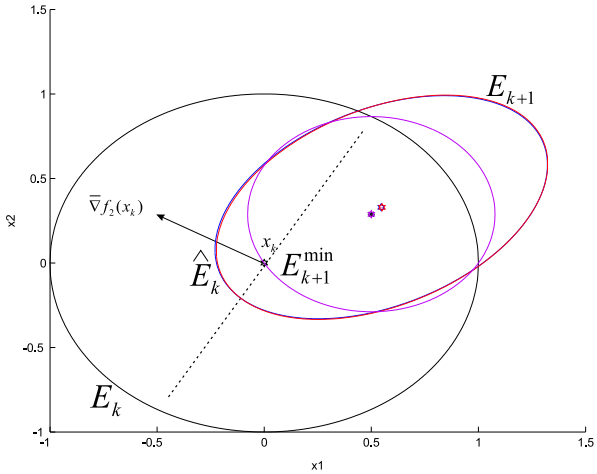
c) Corte do Passo 3.



d) Corte do Passo 4.



e) Corte do Passo 5.



f) Corte do Passo 6.

Figura 6.2: Ilustração do cálculo de E_{k+1} pelo algoritmo Seqüência de Cortes Profundos e Rasos.

6.2 *Historical Intersection Search Polyhedron Ellipsoid Method*

A utilização da definição do poliedro de busca permite a determinação da menor região de busca possível de ser obtida com as informações disponíveis em um determinado ponto $\mathbf{w}$ e um ponto $\mathbf{v}$. Todavia, deve ser observado que durante a seqüência de convergência de um método *Poliedro-Elipsoidal* foram gerados "k"poliedros de busca, os quais representam informações já calculadas. Portanto, desde que estas informações sejam armazenadas, elas poderão ser utilizadas para se tentar obter uma região de interesse de busca ainda menor, ou seja, definir o poliedro de busca com histórico de intersecções, tal como formalizado na Definição 5 (página 79) e no Corolário 3 (página 79). A utilização deste poliedro de busca com histórico de intersecções definirá um novo método baseado em *Elipsóide*, aqui denominado $HISPE$ (i.e., do inglês *Historical Intersection Search Polyhedron Ellipsoid Method*) por referência a (Moura & Takahashi 2007).

Na seção 1.2.2 foi apresentada a existência de certas classes de problemas que justificam um acréscimo no esforço de cálculo do algoritmo de otimização, ECAO, para reduzir o número de avaliações de informações do problema. Este esforço de cálculo extra pode ser entendido no presente contexto como: *armazene e utilize* as informações já calculadas referente aos poliedros de busca das iterações passadas para o cálculo do novo elipsóide E_{k+1}.

A construção do método $HISPE$ inicia-se com a verificação de quais são os vetores suporte que definem o poliedro de busca com histórico de intersecções para a iteração corrente, a partir das informações já calculadas nas iterações passadas. Todavia, a Definição 5 (página 79) pressupõe a utilização de todas as informações previamente calculadas para os pontos do conjunto ζ. Isto implica que o algoritmo $HISPE$ deverá armazenar todos os pontos $\mathbf{x}$ e os respectivos vetores subdiferenciais avaliados nestes pontos. A Definição 8 define duas matrizes com as informações que deverão ser armazenadas para a construção do algoritmo $HISPE$.

Definição 8 - *Matrizes M_{HP} e M_{HX} : Considere pela Definição 5 (página 79) um conjunto $\zeta \triangleq \{[\mathbf{x}_0, \ldots, \mathbf{x}_k]\}$ e um poliedro de busca $P_\zeta(\mathbf{x}_s, E_k)$. Da Definição 6 considere M_{Ck} e defina M_{CXk} correspondente a uma matriz de mesma dimensão de M_{HPk}, cujas colunas representam o ponto onde os vetores $M_C(:, i)$ foram avaliados. Define-se M_{HP} e M_{HX}, tal que*

$$M_{HP} = \bigcup_{j}^{\zeta} M_{Cj} \tag{6.5}$$

$$M_{HX} = \bigcup_{j}^{\zeta} M_{CXj} \tag{6.6}$$

Por esta definição, tem-se que:

$$P_\zeta(\mathbf{x}_s, E_k) \Longleftrightarrow (E_k \bigcap_{i=1}^{col(M_{HP})} H_-^{M_{HP}(:,i),M_{HX}(:,i)}) \tag{6.7}$$

Deve-se esperar que à medida que o elipsóide atual tenha seu tamanho contraído, existirão poliedros de busca calculados em iterações anteriores que conterão integralmente o poliedro de busca da iteração atual. Conseqüentemente, alguns dos poliedros de iterações anteriores em nada contribui para a determinação de $P_\zeta(\mathbf{x}_s, E_k)$. Este fato torna provável a hipótese de que na prática haverá um limite, algum ponto $\mathbf{x} \in \zeta$, a partir do qual não mais será possível encontrar informações úteis nos poliedros de busca calculados anteriormente a este ponto.

Por conseguinte, pode-se definir um parâmetro τ que limitará a utilização das informações armazenadas nas matrizes M_{HP} e M_{HX}. Este parâmetro é aqui denominado por tamanho da busca histórica. Assim,

$$P_\zeta(\mathbf{x}_s, E_k) \subseteq (E_k \bigcap_{i=k-1}^{k-\tau} H_-^{M_{HP}(:,i),M_{HX}(:,i)}) \tag{6.8}$$

Deve ser enfatizado que não existe uma definição prévia de que a partir de um determinado ponto do histórico os valores garantidamente não mais serão reaproveitáveis para os cortes atual e futuros. Isto é decorrente do fato de que um hiperplano, definido por vetor subdiferencial de uma das funções do problema avaliado em um ponto qualquer da seqüência de convergência, pode constituir uma face do poliedro $P_\zeta(\mathbf{x}_s, E_k)$.

Quando aplicado à solução de problemas de características pouco conhecidas associadas aos objetivos e restrições ou de elevado custo de avaliação, a busca deve ser realizada até o primeiro poliedro de busca gerado. Nos demais casos, como poderá ser verificado nos resultados práticos que serão apresentados no capítulo seguinte, a heurística de se estabelecer o valor de $\tau \approx 3n$ representa uma boa relação entre esforço computacional de busca com o ganho real de redução do volume. Esta discussão será retomada quando da apresentação dos resultados obtidos.

O algoritmo 10 exibe a estruturação geral do método de *Poliedro-Elipsoidal* por Seqüência de Cortes Profundos e Rasos, $MPESC$. Tal como definido para o algoritmo 8, considera-se que M_{HPk} e M_{HXk} são definidas pelo poliedro de busca $P_{\mathbf{x}_k}(\mathbf{x}_s, E_k)$. Para facilitar a representação do algoritmo, define-se que $f(\emptyset) = \infty_+$ e $C_{\mathbf{v}}(\emptyset, \Gamma) = C_{\mathbf{v}}(\mathbf{v}, \Gamma)$. Pelo mesmo motivo foi excluído o critério de parada $\mathbf{x}_k \in \Omega^*$. Deve-se ainda observar que este algoritmo mantém pelo menos a mesma classe de complexidade polinomial do algoritmo *Elipsoidal* de Shor. A cada iteração é executado o algoritmo 8 e este possui complexidade igual à complexidade do algoritmo *Elipsoidal* de Shor, resultando em um número menor ou igual de iterações.

Algorithm 10 *Historical Intersection Search Polyhedron Ellipsoid*

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^*$ será procurada de acordo com o problema definido por (6.1). O tamanho $\tau > 0$ para a busca de valores históricos.

Output: A melhor solução factível contínua $\mathbf{x}^f$ encontrada. Para um problema infactível ou para $\mathbf{x}^* \notin E_0$ ter-se-á $\mathbf{x}^f = \emptyset$.

function $Shor(E_0)$
  $k \leftarrow 0$
  $\mathbf{x}^f \leftarrow \emptyset$ ▷ Zere a melhor solução factível.
  $M_{HP} \leftarrow \emptyset$ ▷ Zere as matrizes com informações já calculadas.
  $M_{HX} \leftarrow \emptyset$
  loop
    $P_{\mathbf{x}_k}(\mathbf{x}^{\mathbf{f}}, E_k)$ ▷ Calcula o poliedro de busca a partir de $\mathbf{x}^{\mathbf{f}}, E_k, f(\cdot)$ e $g(\cdot)$ por (5.4).
    $M_{Ck} \leftarrow P_{\mathbf{x}_k}(\mathbf{x}^{\mathbf{f}}, E_k)$
    $M_{CXk} \leftarrow P_{\mathbf{x}_k}(\mathbf{x}^{\mathbf{f}}, E_k)$
    $M_{HP} \leftarrow [M_{HP} \quad M_{Ck}]$
    $M_{HX} \leftarrow [M_{HX} \quad M_{CXk}]$ ▷ Vide Definição 8 (página 79).
    if $\mathbf{x}_k \in \Omega$ e $f(\mathbf{x}_k) \prec f(\mathbf{x}^f)$ **then**
      $\mathbf{x}^f \leftarrow \mathbf{x}_k$ ▷ Novo melhor ponto factível.
    end if
    $E_{k+1} \leftarrow SCPR(E_k, (E_k, M_{HP}(:, \{k-\tau, \ldots, k\}), M_{HX}(:, \{k-\tau, \ldots, k\}))$ ▷ Vide Alg.(8)
    if vol $(E_{k+1}) \simeq 0$ ou $vol(E_{k+1}) \simeq vol(E_k)$ **then**
      return $\mathbf{x}^f$ ▷ Termine o algoritmo.
    end if
    $k \leftarrow k + 1$
  end loop
end function

6.3 Conclusões do Capítulo

Neste Capítulo foram apresentados os métodos que implementam as variantes do método *Elipsoidal* e que se fundamentam nos conceitos do poliedro de busca e do poliedro de busca com histórico de intersecções. Em seguida, os métodos propostos foram codificados nos algoritmos $MPEH$, $MPESC$ e $HISPE$.

No capítulo seguinte serão exibidos os problemas restritos e irrestritos, escalares e vetoriais utilizados como simulação de teste dos métodos propostos, bem como os resultados alcançados.

Capítulo 7

Resultados Computacionais para Problemas com Variáveis Contínuas

Neste capítulo são exibidos os problemas de simulação com variáveis contínuas, escalares ou vetoriais, quasi-convexos não necessariamente diferenciáveis propostos para a validação dos algoritmos que constituem a família dos métodos *Poliedro-Elipsoidais*.

No seu desenvolvimento são mostrados os resultados obtidos pelos algoritmos *Poliedro-Elipsoidais* apresentados no capítulo 6 e desenvolvidos a partir das definições, corolários, lemas e teoremas apresentados no capítulo 5.

Para referência da avaliação do desempenho dos algoritmos propostos, os resultados obtidos serão exibidos conjuntamente com os correspondentes resultados para os algoritmos: *Elipsoidal* proposto por *Shor* e o algoritmo dos *Dois Subgradientes Sucessivos* proposto por *Kim*. Ressalta-se aqui que o algoritmo $MPEH$ não foi incluído entre os algoritmos avaliados por ter apresentado resultados muito próximos ao algoritmo *Shor*. Já os algoritmos *Cortes Paralelos* e *Cortes em Cunha* não foram utilizados na comparação porque constituem um caso particular do algoritmo *Dois Subgradientes Sucessivos*.

Quando à implementação dos algoritmos, foi utilizada a fatorização UDU para melhorar a estabilidade numérica do algoritmo, reduzindo os erros que conduzem à perda da positividade de Q_k antes de uma convergência satisfatória[1]. Já quanto à parametrização do tamanho da busca histórica o algoritmo será indicado por $HISPE_{\{\tau\}}$. A indicação de τ como $kmax$ determina que a busca histórica ocorreu em todo o conjunto de valores históricos disponíveis em cada iteração k. Foram apresentadas duas configurações para o algoritmo $HISPE$, porque a configuração com $\tau = 3n$ representa a configuração padrão recomendada para o valor de τ, enquanto $\tau = kmax$ representa a configuração

[1]Nos testes realizados para problemas com variáveis contínuas observou-se um desempenho equivalente para a fatorização de *Cholesky* e a fatorização UDU.

recomendada para problemas onde o ECAO é insignificante em comparação ao EAIP.

Como detalhe da implementação, deve-se registrar que todos os algoritmos foram codificados utilizando-se a linguagem *MatLab*®, em sua versão 6.5, tal como todos os testes computacionais foram executados em um computador *Intel Pentium IV*® 2.40GHz com Microsoft *Windows*® *XP*.

Como critérios de parada para os laços dos algoritmos *Elipsoidais* foram utilizados a estabilização para o conjunto $\vartheta \triangleq \{\mathbf{x}_k, \ldots, \mathbf{x}_{k-4} \mid \mathbf{x} \in \mathbb{R}^n\}$ tal que a função erro definida como $\varepsilon(\cdot) = \{max(\mathbf{x}_i - \mathbf{x}_j) \mid \forall\, i \in \vartheta \mid \forall\, j \in \vartheta\}$ apresentasse valor inferior a um parâmetro $\epsilon = 10^{-6}$ fornecido, bem como a perda de positividade de Q_k.

Em todos os testes propostos, os limites de $\mathbf{x}$ foram definidos por um hipercubo $\mathbf{C} \subset \mathbb{R}^n_+$ de lado $lado^{\mathbf{C}}$ igual a 20, sendo que o elipsóide E_0 foi inicializado com a menor hiperesfera que continha o hipercubo, centralizado em um ponto $\mathbf{x}_0$ gerado aleatoriamente de modo que $(E_0 \cap \Omega^*) \neq \emptyset$. Não foi definido um número máximo de iterações, de modo que o algoritmo sempre parasse por um dos critérios definidos.

De modo a gerar uma análise estatística preliminar dos resultados, cada problema definido por uma dimensão específica foi testado com um conjunto de 10 parâmetros distintos gerados aleatoriamente, sendo que os resultados apresentados a seguir indicarão, salvo quando explicitamente informado, o valor médio dos resultados obtidos em cada teste.

Nas tabelas de resultados temos as seguintes informações apresentadas como parâmetros de avaliação de desempenho dos métodos:

Iterações (k) := Representa o número de iterações alcançado até a ocorrência de um dos critérios de parada previstos.

Tx Redução Volume := Indica a taxa de redução de volume obtida durante a convergência do algoritmo após as k iterações. Calculado por:

$$\prod_{i=2}^{k} vol(E_i)/vol(E_{i-1}))^{1/(k-1)};$$

Acessos a $f(\mathbf{x})$:= número de vezes em que a função objetivo foi avaliada. Esta informação é usada como medida do esforço computacional para avaliação do modelo matemático do problema.

Tempo Total:= tempo total em segundos para se encontrar a solução ótima $\mathbf{x}^*$. Esta informação é usada como medida da soma do esforço computacional para avaliação do modelo matemático do problema e do esforço computacional do algoritmo durante o processo de convergência para a solução ótima. Para validar este parâmetro,

a solução ótima encontrada por todos os métodos foi comparada. Considerando-se $\bar{\mathbf{x}}^*$ a melhor solução encontrada dentre todos os algoritmos testados e $\mathbf{x}^*$ a solução encontrada por um algoritmo, o erro quadrático $(\bar{\mathbf{x}}^* - \mathbf{x}^*)^T(\bar{\mathbf{x}}^* - \mathbf{x}^*)$ para cada algoritmo em cada problema testado foi sempre inferior a 10^{-4}.

% Soluções Não-Dominadas := Exclusivamente para o problema PCVQC e para o problema H_2/H_∞, representa o percentual, em relação ao número total de problemas testados, das soluções encontradas por um determinado algoritmo e que não foram dominadas por quaisquer outras soluções encontradas por qualquer ponto avaliado por qualquer algoritmo utilizado para solucionar aquele problema em específico. Para este teste, foi considerado um valor de precisão das soluções igual a 10^{-4} para o valor normalizado das funções-objetivo.

$|\frac{\nabla f_1(x^*)}{|\nabla f_1(x^*)|} + \frac{\nabla f_2(x^*)}{|\nabla f_2(x^*)|}|$:= Exclusivamente para o problema PCVQI, representa a norma da soma dos vetores $\nabla f_1(\mathbf{x})$ e $\nabla f_2(\mathbf{x})$ para cada solução encontrada por um dos métodos propostos como medida do erro desta solução. Dado o problema ser irrestrito, quadrático e composto por dois objetivos, estes vetores devem ser paralelos de sentido oposto para os pontos do conjunto *Pareto-Ótimo*.

7.1 Problemas Escalares com Variáveis Contínuas

Duas classes de problemas escalares foram utilizadas para avaliar o desempenho dos algoritmos propostos. A primeira classe é composta por problemas QCND de diversas dimensões nos quais os parâmetros foram gerados de forma aleatória. Nesta classe de problemas também foram testadas as subclasses de problemas convexos não diferenciáveis e de problemas convexos diferenciáveis. A segunda é composta por problemas de Engenharia Química. A definição destes problemas e os resultados alcançados são exibidos nas próximas seções.

7.1.1 Problema Quasi-Convexo Não Necessariamente Diferenciável com Parâmetros Aleatórios

O problema contínuo, vetorial e quasi-convexo não necessariamente diferenciável é aqui denominado PCEQC e representado por (7.1). Todas as variáveis são restritas ao hipercubo $\mathbf{C} \subset \mathbb{R}^n_+$ e canto inferior esquerdo no ponto $\{0, \ldots, 0\} \in \mathbb{R}^n$.

$$min\ f(\mathbf{x}) = max(\{-(1 + e^{(\lambda(j)(\mathbf{x}(j) - \tilde{\mathbf{x}}(j)))^2})\}) \mid j = 1, \ldots, n \tag{7.1}$$

$$s.a.\ \Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}^n\}$$

$$g(\mathbf{x}) = \begin{cases} g_i(\mathbf{x}) = (\mathbf{x} - \breve{\mathbf{x}}_i)^T (\breve{Q}_i)^{-1} (\mathbf{x} - \breve{\mathbf{x}}_i) \mid i = 1, \ldots, n+1 \\ g_{i+n+1}(\mathbf{x}) = -\mathbf{x}(i) \mid i = 1, \ldots, n \\ g_{i+2n+1}(\mathbf{x}) = \mathbf{x}(i) - lado^{\mathbf{C}} \mid i = 1, \ldots, n \end{cases}$$

onde $\tilde{\mathbf{x}}$ e $\breve{\mathbf{x}}_i \in \mathbf{C}$, $0 < \lambda_j < 10^{-1}$, e $\breve{Q}_i \in \mathbb{R}^{n \times n}$ é uma matriz simétrica definida positiva. Os centros $\breve{\mathbf{x}}_i$ dos elipsóides, o mínimo irrestrito $\tilde{\mathbf{x}}$ e o multiplicador λ são gerados de forma aleatória. As matrizes $\breve{Q}_i$ são construídas com autovalores aleatórios de modo a garantir que $(\breve{E}_1 \cap \ldots \breve{E}_{n+1}) \neq \emptyset$, onde $\breve{E}_i$ representa o elipsóide definido pelo centro $\breve{\mathbf{x}}_i$ e pela matriz $\breve{Q}_i$.

Para criar as tabelas e gráficos de resultados, os problemas PCEQC foram gerados para dimensões $n = \{5, 15, 25, 35\}$.

A partir dos resultados obtidos dos algoritmos propostos $MPESC$ e $HISPE$ e dos algoritmos de referência $Shor$ e Kim foram geradas as tabelas 7.1 a 7.4. Deve-se observar que foram utilizadas duas configurações do parâmetro tamanho da busca histórica, ou seja, $3n$ e $kmax$.

Algoritmos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
$Shor$	1121	0.9042	373	1.90
Kim	385	0.8774	118	0.71
$MPESC$	747	0.8775	303	1.23
$HISPE_{15}$	90	0.5857	48	0.28
$HISPE_{kmax}$	85	0.5648	48	0.37

Tabela 7.1: Resultados comparativos dos algoritmos $Shor$, Kim, $MPESC$ e $HISPE$ para um problema PCEQC de dimensão 5.

Algoritmos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
$Shor$	7774	0.9672	1790	101.59
Kim	3217	0.9623	739	40.83
$MPESC$	2251	0.9579	658	26.72
$HISPE_{45}$	618	0.6360	312	10.45
$HISPE_{kmax}$	561	0.6151	301	20.95

Tabela 7.2: Resultados comparativos dos métodos $Shor$, Kim, $MPESC$ e $HISPE$ para um problema PCEQC de dimensão 15.

Algoritmos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	18926	0.9802	2546	838.09
Kim	8159	0.9778	1221	339.15
MPESC	4501	0.9729	865	173.25
$HISPE_{75}$	441	0.6671	197	24.45
$HISPE_{kmax}$	435	0.6605	199	74.18

Tabela 7.3: Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 25.

Algoritmos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	35323	0.9858	5195	3971.78
Kim	15515	0.9835	2625	1599.83
MPESC	5551	0.9773	1416	499.08
$HISPE_{105}$	600	0.6704	259	76.82
$HISPE_{kmax}$	569	0.6535	254	692.36

Tabela 7.4: Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 35.

O primeiro resultado que pode ser verificado nas tabelas 7.1 a 7.4 é referente ao parâmetro de desempenho taxa de redução do volume do elipsóide. Como era esperado, vide seções 6.1 e 5.2, os métodos *MPESC* e *HISPE* apresentaram uma considerável melhora desta taxa para todas as dimensões testadas. Claramente, a definição do poliedro de busca cumpriu sua função conceitual de produzir uma região de interesse de busca menor que aquela definida pelos métodos baseados em *Elipsóide* existentes na literatura. Comparado ao algoritmo *Shor* as duas configurações do algoritmo *HISPE* produziram uma melhora média de 30% para a taxa de redução do volume do elipsóide, que se manteve com o aumento da dimensão dos problemas. Outro ponto que pode ser observado nas tabelas é que o algoritmo *HISPE*, em ambas as configurações, apresentou melhor taxa de redução do volume do elipsóide em comparação ao algoritmo *MPESC*. Conseqüentemente é plausível afirmar que a definição do poliedro de busca com histórico de intersecções efetivamente produziu uma região de busca menor que aquela definida apenas pelo poliedro de busca. Da mesma forma, observa-se que a configuração para o algoritmo $HISPE_{kmax}$ sempre apresentou melhores resultados para a taxa de redução do volume do elipsóide do que a configuração $HISPE_{3n}$. Como comentário final sobre este parâmetro de análise, registra-se os bons resultados do algoritmo *Kim* em comparação ao algoritmo *Shor*, tal como apresentados em (Kim et al. 1994).

Em concordância com a argumentação descrita acima para o parâmetro taxa de redução do volume do elipsóide tem-se os resultados do parâmetro de desempenho do número de iterações necessárias para alcançar um dos critério de parada utilizados. Dado que a proposição do poliedro de busca produz uma região de busca menor, espera-se que os algoritmos baseados nesta idéia convirjam em um número menor de iterações. Esta expectativa é confirmada pelos valores apresentados nas tabelas 7.1 a 7.4, nas quais se verifica que os algoritmos propostos $MPESC$ e $HISPE$ apresentaram valores muito inferiores aos algoritmos $Shor$ e Kim. Ressalta-se que a razão entre o número de iterações de um algoritmo proposto em relação ao número de iterações do algoritmo $Shor$ ou Kim diminui consideravelmente com o aumento da dimensão dos problemas, para ambos os algoritmos $MPESC$ e $HISPE$. Também foi observado que o algoritmo $HISPE$, em ambas as configurações, apresentou menor número de iterações em comparação ao algoritmo $MPESC$. Quanto à comparação entre a configuração do algoritmo $HISPE_{kmax}$ e a configuração $HISPE_{3n}$ para este parâmetro de desempenho, verifica-se que os resultados são similares.

O terceiro ponto passível de discussão é determinado pelos resultados apresentados nas tabelas 7.1 a 7.4 para o parâmetro de desempenho referente ao número de acessos à função $f(\cdot)$. Novamente, verifica-se que os algoritmos propostos $MPESC$ e $HISPE$ apresentaram valores muito inferiores aos dos algoritmos $Shor$ e Kim. Também pode ser constatado que a razão entre o número de acessos à função $f(\cdot)$ de um algoritmo proposto em relação ao número de acessos à função $f(\cdot)$ do algoritmo $Shor$ ou Kim diminui com o aumento da dimensão dos problemas, para ambos os algoritmos $MPESC$ e $HISPE$. Uma vez que esta informação pode ser utilizada como medida do EAIP, a grande diferença constatada gera boas perspectivas para a definição da classe de problemas para os quais os algoritmos propostos apresentarão bom desempenho. Esta afirmação se apóia na lógica de que, se a razão entre os resultados dos algoritmos propostos em relação aos algoritmos de referência reduz com o aumento da dimensão, tanto para o número de iterações quanto para o número de acessos à função $f(\cdot)$, temos efetivamente uma menor demanda por avaliações das informações de $f(\cdot)$ e $g(\cdot)$ de um problema. Por conseguinte, não apenas para problemas com demanda de esforço computacional similar aos aqui testados, mas também com demanda superior, os algoritmos $MPESC$ e $HISPE$ deverão apresentar resultados superiores aos que seriam obtidos com os algoritmos $Shor$ ou Kim. O algoritmo $HISPE$, em ambas as configurações, também foi superior ao algoritmo $MPESC$ tendo como base de comparação o parâmetro número de acessos à função $f(\cdot)$.

Finalmente, o parâmetro referente ao tempo total para convergência apresentou os resultados que mais devem ser destacados. Isto porque a proposição do poliedro de busca

e do poliedro de busca com histórico de intersecções necessariamente implica num aumento de ECAO. Portanto, para que ocorresse uma redução do parâmetro tempo total para convergência é necessário que o ganho com a redução do EAIP supere o aumento de ECAO. Dado que os problemas PCEQC aqui testados são representados por equações analíticas, não era possível pressupor, a priori, que estes problemas estariam inclusos na classe de problemas onde os algoritmos *Poliedro-Elipsoidais* produziriam tempo total para convergência inferiores aos dos métodos *Shor* e *Kim*. Este fato é ainda mais significativos para a configuração com $\tau = kmax$ do algoritmo $HISPE$, já que nesta configuração temos o máximo valor do ECAO. As mesmas análises sobre o desempenho dos algoritmos $MPESC$ e $HISPE$ em relação aos algoritmos *Shor* e *Kim* podem ser confirmadas para este parâmetro de desempenho, o que colabora para consolidar o bom desempenho dos algoritmos aqui propostos. Quanto à comparação do tempo total para convergência entre a configuração do algoritmo $HISPE_{kmax}$ e a configuração $HISPE_{3n}$, verifica-se que os valores obtidos pela configuração $HISPE_{3n}$ são sempre melhores que os obtidos $HISPE_{kmax}$. Este fato era previamente esperado, dado que o ECAO da configuração do algoritmo $HISPE_{3n}$ deve ser sempre igual ou maior que o ECAO da configuração do algoritmo $HISPE_{kmax}$. Cabe ainda comentar que no problema de dimensão 5 o algoritmo Kim apresentou um menor tempo total de convergência do que o obtido pelo algoritmo $MPESC$. Este fato se deve ao maior esforço computacional requerido pela função $SCPR$ em relação à utilização de equações analíticas pelo algoritmo Kim. Como o problema é de baixa dimensão e complexidade este resultado era esperado, tal como pode ser verificado na argumentação apresentada no fim da seção 6.1.2.

Os resultados obtidos para todos os problemas testados são mostrados em valores percentuais nas figuras 7.1 e 7.2, tendo por base de referência os resultados do algoritmo *Shor*. A figura 7.1 permite uma melhor visualização da tendência de queda da razão dos resultados do número de acessos à função $f(\cdot)$ dos algoritmos $MPESC$ e $HISPE$ em referência aos resultados *Shor*, com o aumento da dimensão do problema. Resultado similar para o parâmetro tempo total de convergência é exibido na figura 7.2. Claramente esta figura permite confirmar, para os problemas testados, que à medida que a dimensão dos problemas aumenta o EAIP, para o algoritmo $MPESC$ e para a configuração do algoritmo $HISPE_{3n}$, reduz em uma taxa muito maior que aumenta o ECAO. Como conseqüência, os valores percentuais exibidos nas figuras apresentam tendência de queda com o aumento da dimensão dos problemas. Quanto à oscilação do valor percentual para o tempo total para convergência apresentado para a configuração $HISP_{kmax}$, dada a utilização de toda a informação histórica disponível, para problemas com baixo ECAO é possível que esta relação inclusive aumente com o aumento da dimensão. Deve-se ainda

notar que para o algoritmo Kim não há a tendência de redução dos valores percentuais com o aumento da dimensão dos problemas para nenhum dos parâmetros de desempenho exibidos nas figuras.

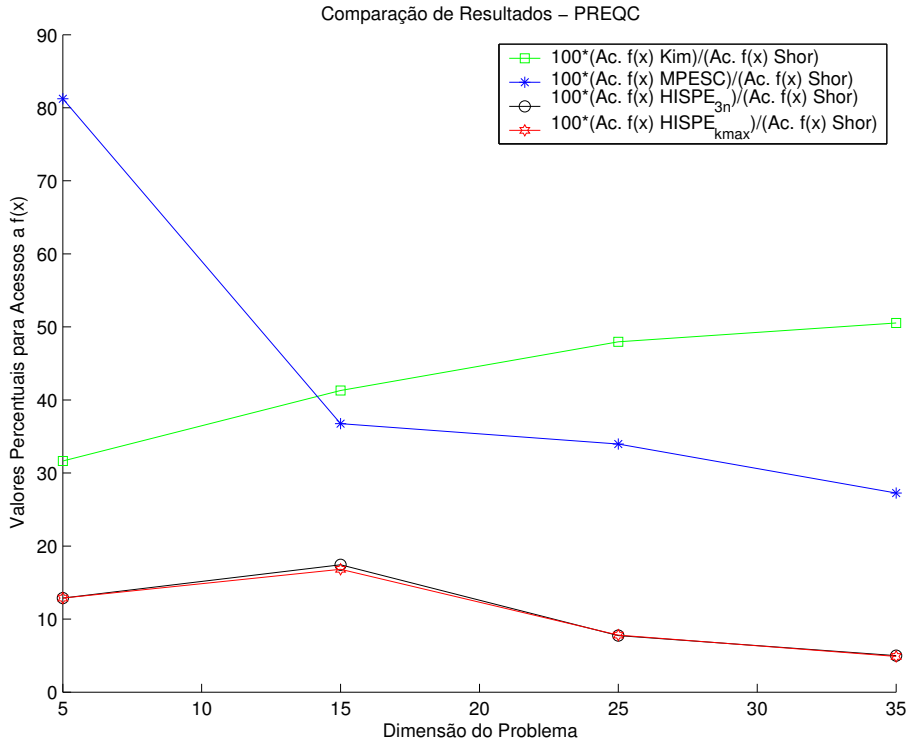


Figura 7.1: Resultados percentuais para o número de acessos à $f(\mathbf{x})$ dos algoritmos Kim, $MPESC$ e $HISPE$, em referência a $Shor$, para problemas PCEQC.

Para avaliar o desempenho do algoritmo $HISPE$ em função do tamanho da busca histórica a ser realizada, foram repetidos os testes dos problemas, cujos resultados foram exibidos nas figuras 7.1 a 7.4, variando-se o tamanho da busca por valores históricos τ. O tamanho do parâmetro τ foi variado desde zero, ou seja, correspondendo ao algoritmo $MPESC$, até um valor de τ no qual o número de acessos à função $f(\cdot)$ permanecesse estabilizado. A figura 7.3 exibe os resultados.

Cabe ressaltar que os resultados exibidos na figura 7.3 permitem definir uma heurística para o parâmetro do tamanho da busca por valores históricos τ. Embora a busca histórica possa ser realizada em todo o conjunto de informações disponíveis, o parâmetro τ que produz o melhor conjunto de resultados para os parâmetros de desempenho se encontra limitado a um valor de $\tau \approx 3n$. Claramente as subfiguras 7.3.a a 7.3.d permitem verificar que, após determinado valor de τ, a taxa de redução do volume do elipsóide se estabiliza. Portanto, o algoritmo não mais conseguiu localizar alguma informação que permitisse construir poliedros de menor volume. Concomitantemente, os valores para

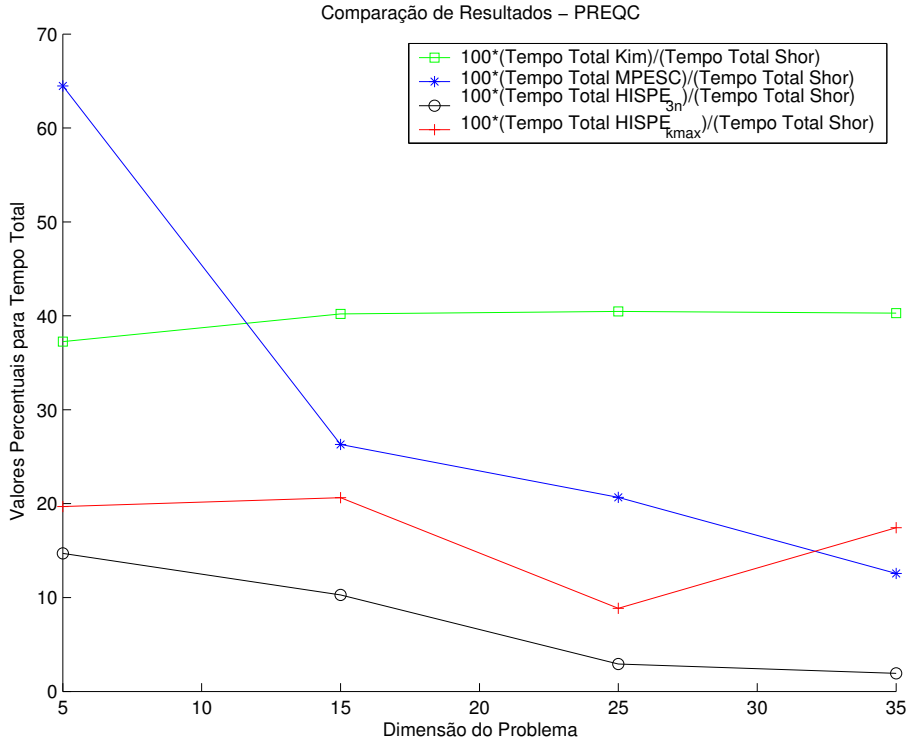
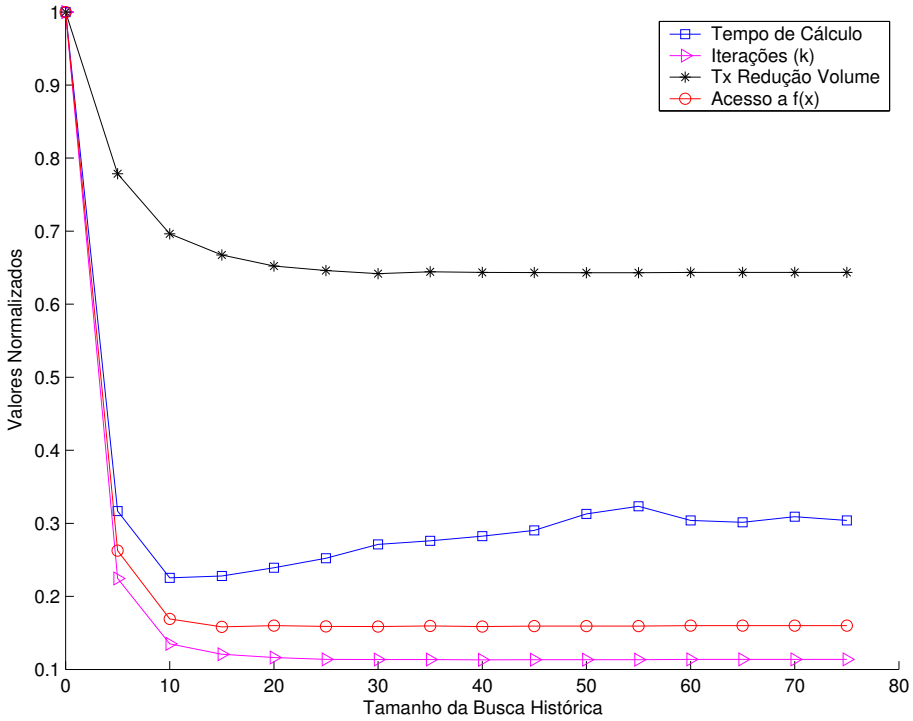
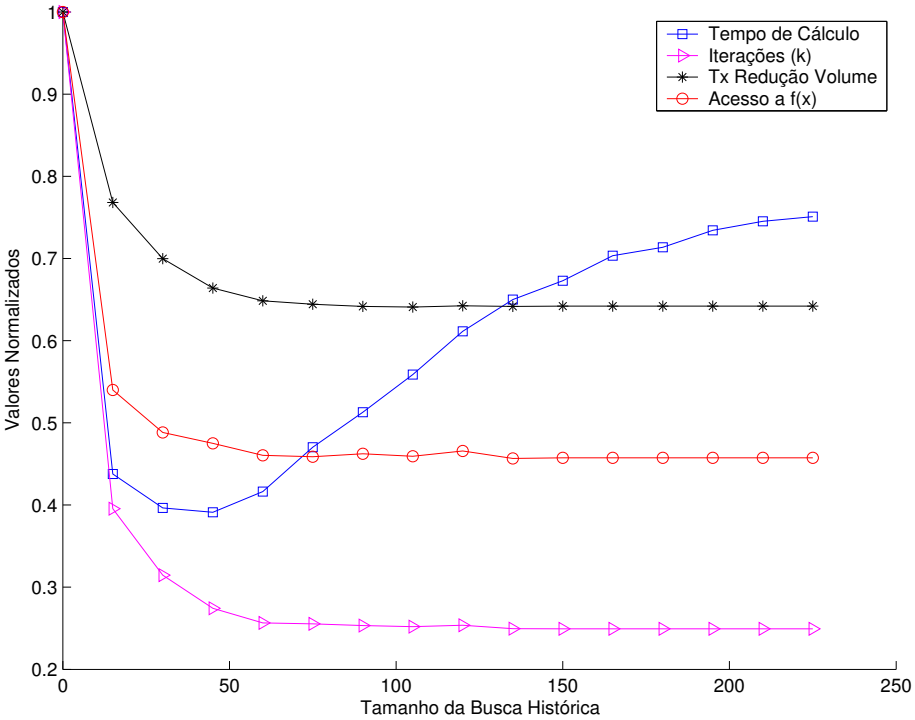


Figura 7.2: Resultados percentuais para o tempo total para convergência dos algoritmos Kim, $MPESC$ e $HISPE$, em referência a $Shor$, para problemas PCEQC.

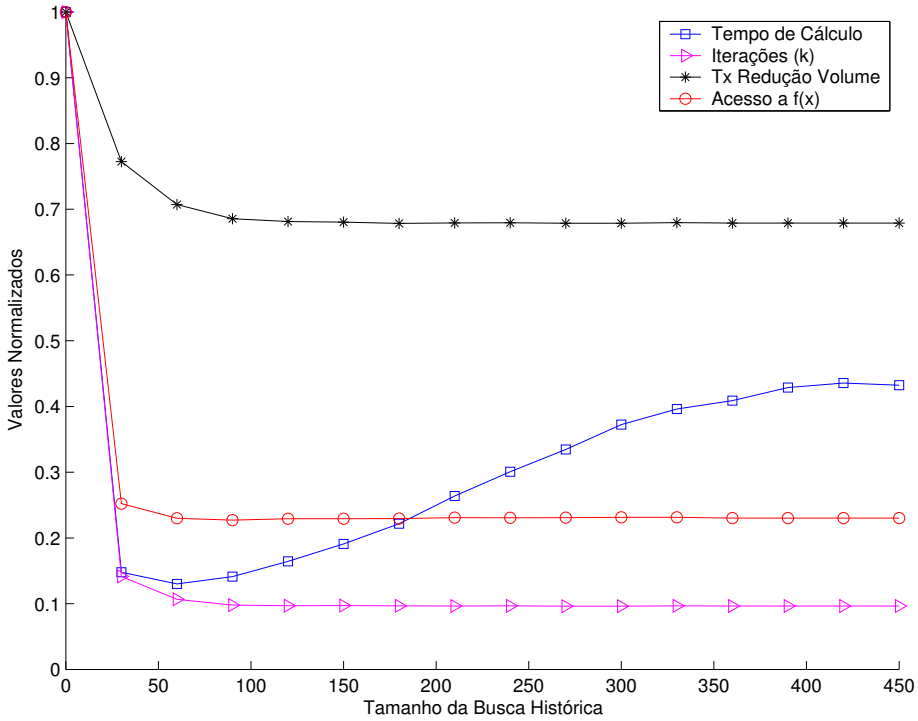
número de iterações e números de acesso à função $f(\cdot)$ também se estabilizam. Um comportamento diferente exibe o parâmetro tempo total para convergência. Uma vez que não mais existem informações históricas úteis para a construção dos poliedros após certo ponto do histórico de valores já calculados, buscar informações além deste ponto somente resulta em aumento do tempo total para convergência. Um comportamento ainda não explicado é caracterizado pelas oscilações observadas nas subfiguras 7.3.a a 7.3.d para a tendência de aumento do tempo total para convergência. Este fato foi relegado a segundo plano pois, dada a predominância de se adotar valores de $\tau \approx 3n$, os valores de τ que definem o intervalo onde ocorrem as oscilações deixam ter relevância.



a) Problema PCEQC de dimensão 5



b) Problema PCEQC de dimensão 15



c) Problema PCEQC de dimensão 25

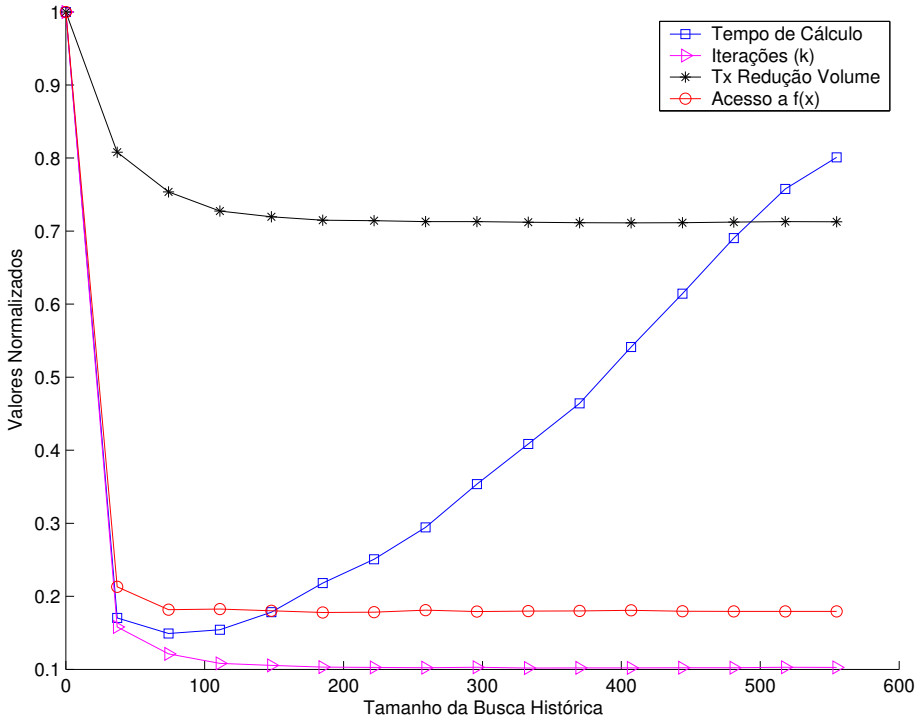


Figura 7.3: Desempenho do algoritmo HISPE mediante a variação do tamanho da busca histórica para o problema PCEQC de dimensões 5, 15, 25 e 35.

7.1.1.1 Influência do Aumento do Número de Restrições no Desempenho dos Algoritmos *Poliedro-Elipsoidais*

Devido ao fato de que os métodos *Poliedro-Elipsoidais* utilizam todas as informações dos vetores subdiferenciais avaliados no ponto corrente, deve ser também analisado o comportamento destes métodos quando do aumento do número de restrições de um problema. Para permitir esta análise, foi utilizado o problema PCEQC representado por 7.1. Todavia, o conjunto de restrições $g(\cdot)$ foi redefinido, tal como indicado a seguir.

$$g(\mathbf{x}) = \begin{cases} g_i(\mathbf{x}) = (\mathbf{x} - \breve{\mathbf{x}}_i)^T (\breve{Q}_i)^{-1} (\mathbf{x} - \breve{\mathbf{x}}_i) \mid i = 1, \ldots, \phi \\ g_{i+\phi}(\mathbf{x}) = -\mathbf{x}(i) \mid i = 1, \ldots, n \\ g_{i+n+\phi}(\mathbf{x}) = \mathbf{x}(i) - lado^{\mathbf{C}} \mid i = 1, \ldots, n \end{cases} \tag{7.2}$$

onde todos os parâmetros permanecem tal como anteriormente descritos e acrescenta-se o novo parâmetro ϕ. Desta feita, o número de restrições passa a variar controlado por este parâmetro. Deve-se ainda afirmar que as matrizes $\breve{Q}_i$ são construídas com autovalores aleatórios de modo a garantir que $(\breve{E}_1 \cap \ldots \breve{E}_{n^\phi}) \neq \emptyset$, onde $\breve{E}_i$ representa o elipsóide definido pelo centro $\breve{\mathbf{x}}_i$ e pela matriz $\breve{Q}_i$.

Para criar as tabelas e gráficos de resultados, os problemas PCEQC foram gerados para dimensões $n = \{5, 10, 15\}$ e número de restrições determinado por $\phi = \{100, 200, 300, 400, 500\}$.

A partir dos resultados obtidos dos algoritmos propostos $MPESC$ e $HISPE$ e dos algoritmos de referência *Shor* e *Kim* foram geradas as tabelas 7.5 a 7.7. Deve-se observar que foram utilizadas duas configurações do parâmetro tamanho da busca histórica, ou seja, $3n$ e $kmax$.

Como pode ser constatado nos resultados exibidos nas tabelas 7.5 a 7.7, o já esperado aumento do parâmetro tempo total para convergência ocorre para todos os algoritmos utilizados. Todavia, faz-se necessário observar que os parâmetros número de iterações, taxa de redução do volume do elipsóide e número de acessos a $f(\cdot)$ não necessariamente apresentam valores melhores à medida que o número de restrições do problema aumenta. Esta observação se deve ao fato de que um maior número de restrições não significa necessariamente um maior número de hiperplanos suporte disponíveis para definir o poliedro de busca e conseqüentemente restringir a região de interesse de busca, durante o processo de convergência. Quanto à comparação dos resultados do parâmetro tempo total para convergência obtidos pelos métodos propostos, verifica-se que a configuração do algoritmo $HISPE$ com o parâmetro $\tau = 3n$, apresentou sempre os melhores resultados. Como a configuração $HISPE_{3n}$ apresentou melhores resultados que o algoritmo $MPESC$ e como esta configuração limitou fortemente o número de informações armazenadas, é plausível

supor que seqüências de pontos factíveis devem ter existido e que as informações já calculadas destes pontos em muito colaboraram para a redução da região de interesse de busca. Esta suposição também justificaria os bons resultados do algoritmo Kim em comparação aos resultados do algoritmo $MPESC$. Ressalta-se que, caso as informações resultantes do grande número de restrições fossem sempre suficientes para gerar uma considerável redução do poliedro de busca, o algoritmo $MPESC$ deveria ter apresentado os melhores resultados para o tempo total de convergência. Isto ocorreria uma vez que este algoritmo possui um menor ECAO do que o correspondente ECAO do algoritmo $HISPE$. Outro ponto que fica claro para estes problemas é que o grande número de restrições provoca na configuração $HISPE_{kmax}$ um aumento do ECAO muito superior ao ganho obtido com a redução da região de interesse de busca resultante da utilização de todas as informações já calculadas. Por fim, os resultados das tabelas 7.5 a 7.7 permitem supor que, através do controle do parâmetro τ é possível evitar que o aumento do número de restrições restrinja a aplicabilidade do algoritmo $HISPE$.

Algorimos	ϕ	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	100	1086	0.9042	203	25.55
Kim	100	378	0.8760	63	9.03
MPESC	100	803	0.8725	190	18.48
$HISPE_{15}$	100	135	0.6471	53	2.96
$HISPE_{kmax}$	100	89	0.5582	44	5.23
Shor	200	1100	0.9042	202	55.42
Kim	200	381	0.8766	61	19.72
MPESC	200	783	0.8663	192	36.93
$HISPE_{15}$	200	135	0.6395	55	6.04
$HISPE_{kmax}$	200	95	0.5628	45	17.69
Shor	300	1106	0.9042	208	83.70
Kim	300	367	0.8744	59	29.06
MPESC	300	752	0.8659	184	55.79
$HISPE_{15}$	300	139	0.6453	57	9.06
$HISPE_{kmax}$	300	91	0.5549	45	34.52
Shor	400	1055	0.9042	229	109.00
Kim	400	372	0.8736	69	39.77
MPESC	400	842	0.8809	219	85.22
$HISPE_{15}$	400	142	0.6475	59	13.30
$HISPE_{kmax}$	400	100	0.5787	47	58.49
Shor	500	1132	0.9042	242	158.02
Kim	500	373	0.8729	69	54.11
MPESC	500	777	0.8613	225	101.27
$HISPE_{15}$	500	135	0.6375	57	15.74
$HISPE_{kmax}$	500	88	0.5576	44	121.30

Tabela 7.5: Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 5, em função do aumento do número de restrições.

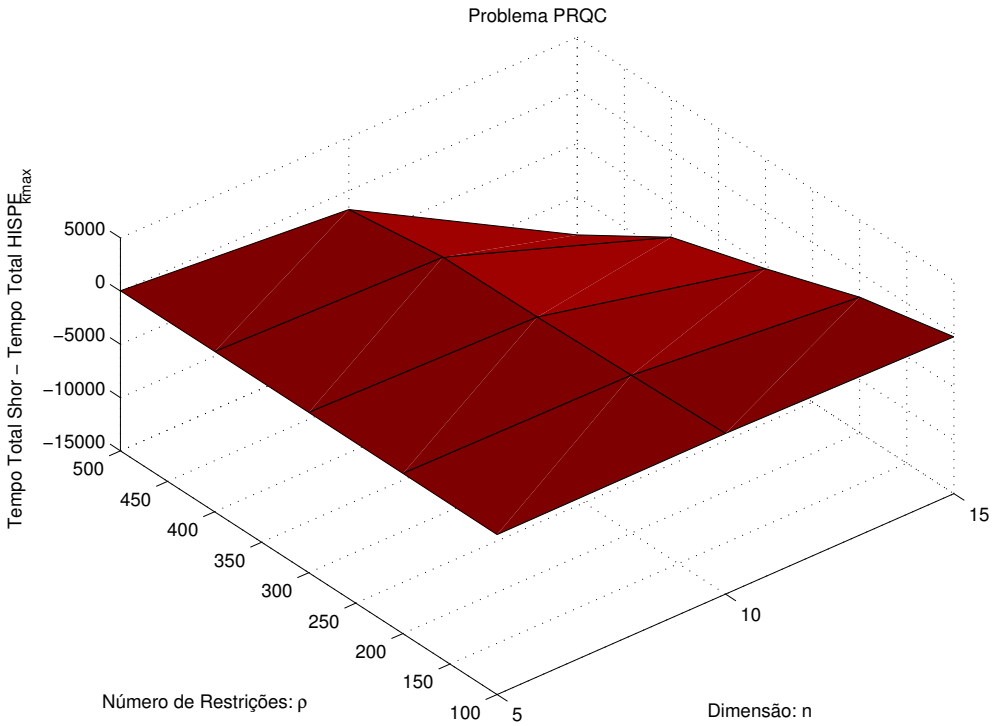
Algorimos	ϕ	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	100	4063	0.9511	591	204.50
Kim	100	1492	0.9447	198	76.68
MPESC	100	2495	0.9211	555	120.92
$HISPE_{30}$	100	249	0.6691	97	11.13
$HISPE_{kmax}$	100	150	0.5567	74	128.49
Shor	200	3982	0.9511	770	398.90
Kim	200	1474	0.9435	261	151.00
MPESC	200	2647	0.9264	736	255.05
$HISPE_{30}$	200	265	0.6664	108	23.29
$HISPE_{kmax}$	200	153	0.5615	77	575.84
Shor	300	4035	0.9511	649	662.79
Kim	300	1510	0.9443	222	251.60
MPESC	300	2543	0.9231	614	402.18
$HISPE_{30}$	300	256	0.6696	101	35.41
$HISPE_{kmax}$	300	154	0.5605	77	1085.82
Shor	400	4068	0.9511	576	1029.72
Kim	400	1505	0.9435	198	381.45
MPESC	400	2471	0.9190	548	592.81
$HISPE_{30}$	400	245	0.6596	97	50.16
$HISPE_{kmax}$	400	145	0.5389	73	1552.12
Shor	500	4267	0.9511	727	1544.40
Kim	500	1530	0.9432	244	578.29
MPESC	500	2378	0.9170	650	759.42
$HISPE_{30}$	500	254	0.6570	105	71.11
$HISPE_{kmax}$	500	145	0.5340	75	3307.12

Tabela 7.6: Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 10, em função do aumento do número de restrições.

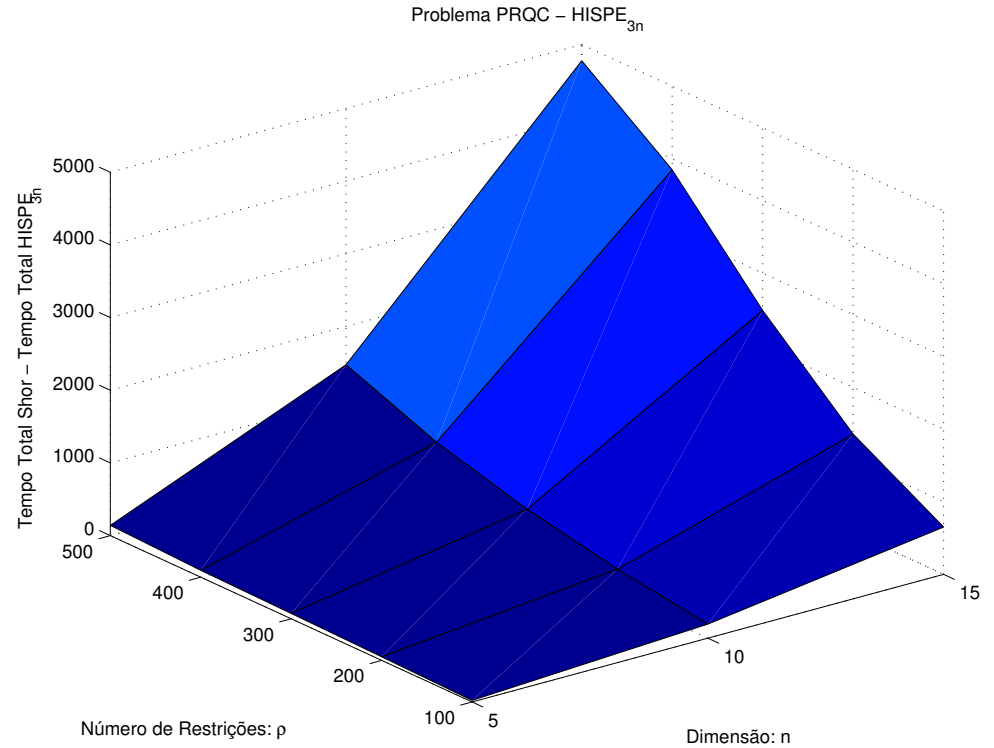
Algorimos	ϕ	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	100	8526	0.9672	1330	698.78
Kim	100	3416	0.9626	536	281.57
MPESC	100	5018	0.9474	1191	405.22
$HISPE_{45}$	100	704	0.6645	311	47.98
$HISPE_{kmax}$	100	508	0.5784	268	964.25
Shor	200	8314	0.9672	1341	1450.51
Kim	200	3688	0.9618	610	639.99
MPESC	200	4789	0.9431	1263	788.78
$HISPE_{45}$	200	684	0.6469	314	90.99
$HISPE_{kmax}$	200	493	0.5477	270	3723.67
Shor	300	8578	0.9672	998	2652.50
Kim	300	3754	0.9628	433	1180.18
MPESC	300	4443	0.9373	925	1339.13
$HISPE_{45}$	300	703	0.6536	299	165.69
$HISPE_{kmax}$	300	475	0.5414	250	7968.23
Shor	400	8662	0.9672	1182	4094.75
Kim	400	3791	0.9628	512	1816.38
MPESC	400	4600	0.9399	1074	2035.01
$HISPE_{45}$	400	703	0.6514	309	245.61
$HISPE_{kmax}$	400	476	0.5352	257	12170.35
Shor	500	8113	0.9672	1645	5104.21
Kim	500	3739	0.9625	755	2332.63
MPESC	500	5565	0.9520	1589	3183.60
$HISPE_{45}$	500	719	0.6708	328	325.93
$HISPE_{kmax}$	500	502	0.5741	280	18683.33

Tabela 7.7: Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC de dimensão 15, em função do aumento do número de restrições.

A partir dos resultados apresentados nas tabelas 7.5 a 7.7 foi gerada a figura 7.4. Como pode ser observado, embora a configuração $HISPE_{kmax}$ resulte em um tempo total para convergência cada vez maior que o tempo obtido pelo algoritmo *Shor*, à medida que o número de restrições do problema aumenta, a configuração $HISPE_{3n}$ apresenta um resultado oposto. Com o aumento do número de restrições do problema, o tempo total de convergência da configuração $HISPE_{3n}$ se torna cada vez menor que o tempo obtido pelo algoritmo *Shor*. O mesmo comportamento, de ambas as configurações do algoritmo *HISPE* em relação ao algoritmo *Shor*, se mantém para o aumento da dimensão dos problemas, porém de forma menos acentuada.



a) Diferença entre o tempo total para convergência obtido pelo algoritmo *Shor* e o tempo total para convergência obtido pelo algoritmo $HISPE_{kmax}$.



b) Diferença entre o tempo total para convergência obtido pelo algoritmo *Shor* e o tempo total para convergência obtido pelo algoritmo $HISPE_{3n}$.

Figura 7.4: Resultados comparativos para problemas PCEQC em função do aumento do número de restrições.

7.1.1.2 Problema Irrestrito

Devido ao fato de que os métodos *Poliedro-Elipsoidais* utilizam todas as informações dos vetores subdiferenciais avaliados no ponto corrente, a redução do número de informações disponíveis em um ponto pode constituir um cenário prejudicial ao desempenho deste métodos. Este raciocínio implica na necessidade de se avaliar os métodos propostos para problemas irrestritos. Uma vez que nos problemas irrestritos o número de informações já calculadas é inferior ao número de informações que existiriam caso o mesmo problema fosse restrito, deve-se analisar o desempenho dos métodos propostos neste contexto. Para permitir esta análise, foi utilizado o problema PCEQC representado por 7.1 com $p = 0$.

Para criar as tabelas e gráficos de resultados, os problemas PCEQC irrestrito foram gerados para dimensões $n = \{5, 15, 25, 35\}$. A partir dos resultados obtidos dos algoritmos propostos $MPESC$, $HISPE$, $Shor$ e Kim foram geradas as tabelas 7.8 a 7.11. Tal como realizado para o caso restrito, foram utilizadas duas configurações do parâmetro tamanho da busca histórica, ou seja, $3n$ e $kmax$.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
$Shor$	682	0.9042	682	0.27
Kim	601	0.8780	600	0.28
$MPESC$	283	0.7634	282	0.23
$HISPE_{15}$	154	0.5863	153	0.35
$HISPE_{kmax}$	133	0.5328	133	0.53

Tabela 7.8: Resultados comparativos dos métodos $Shor$, Kim, $MPESC$ e $HISPE$ para um problema PCEQC irrestrito de dimensão 5.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
$Shor$	3055	0.9672	3055	3.67
Kim	3579	0.9523	3579	4.88
$MPESC$	465	0.9253	464	0.71
$HISPE_{45}$	319	0.8733	319	2.06
$HISPE_{kmax}$	308	0.8473	307	4.57

Tabela 7.9: Resultados comparativos dos métodos $Shor$, Kim, $MPESC$ e $HISPE$ para um problema PCEQC irrestrito de dimensão 15.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	5439	0.9802	5439	14.18
Kim	7308	0.9698	7308	23.12
MPESC	519	0.9704	518	0.95
$HISPE_{75}$	499	0.9664	498	3.06
$HISPE_{kmax}$	500	0.9588	499	9.32

Tabela 7.10: Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC irrestrito de dimensão 25.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	8025	0.9858	8025	39.30
Kim	12783	0.9780	12783	81.27
MPESC	765	0.9777	764	2.47
$HISPE_{105}$	747	0.9754	746	8.42
$HISPE_{kmax}$	700	0.9716	699	26.92

Tabela 7.11: Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQC irrestrito de dimensão 35.

Como pode ser verificado nas tabelas 7.8 a 7.11, mesmo para o caso irrestrito, os métodos propostos demonstraram a validade da utilização das informações já calculadas para reduzir a região de interesse de busca. Os parâmetros número de iterações, taxa de redução do volume do elipsóide e número de acessos a $f(\cdot)$ sempre obtiveram melhora com o aumento do tamanho da busca histórica τ. Já quanto ao parâmetro tempo total para convergência, constata-se que o algoritmo *MPESC* sempre produziu os melhores resultados. Também é possível observar que à medida que o EAIP aumenta, também aumenta o ganho proporcionado pela utilização do poliedro de busca para reduzir a região de interesse de busca, durante o processo de convergência. Faz-se importante observar que para esta classe de problemas o algoritmo *Kim* é um caso particular do algoritmo *MPESC*. Isto ocorre porque ambos os algoritmos utilizam sempre os mesmos dois hiperplanos para reduzir a região de interesse de busca. A diferença entre eles ocorre por dois pontos. O primeiro se deve ao fato do algoritmo *Kim* utilizar uma equação analítica para o cálculo do novo elipsóide, enquanto o algoritmo *MPESC* utiliza a função *SCPR*. O segundo ocorre quando da existência de dois hiperplanos não paralelos, onde o algoritmo *MPESC* calcula um elipsóide de volume inferior ao calculado pelo algoritmo *Kim*. Assim, entre os métodos *Kim* e *MPESC* para esta classe de problemas, a obtenção do menor tempo total de convergência será função do EAIP do problema em questão. Cabe ainda a observação

que, embora o algoritmo $HISPE$ tenha sempre apresentado os melhores resultados para os parâmetros número de iterações, taxa de redução do volume do elipsóide e número de acessos a $f(\cdot)$, o ganho obtido com a utilização das informações já calculadas foi insuficiente para superar o esforço computacional correspondente à sua utilização, nos casos dos problemas analíticos aqui considerados.

7.1.2 Problema Convexo Não Diferenciável com Parâmetros Aleatórios

Dados os bons resultados obtidos para os problemas quasi-convexos, os métodos propostos foram também avaliados quanto ao seu desempenho para a solução de problemas convexos não diferenciáveis. O problema com variáveis contínuas, escalar e convexo não diferenciáveis é aqui denominado PCECN e representado por (7.3). Todas as variáveis são restritas ao hipercubo $\mathbf{C} \subset \mathbb{R}^n_+$ e canto inferior esquerdo no ponto $\{0,\ldots,0\} \in \mathbb{R}^n$.

$$min\ f(\mathbf{x}) = max(\{(\mathbf{x}-\tilde{\mathbf{x}}_j)^T(\tilde{Q}_j)^{-1}(\mathbf{x}-\tilde{\mathbf{x}}_j)\}) \mid j = 1,\ldots,n \tag{7.3}$$

$$s.a.\ \Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}^n\}$$

$$g(\mathbf{x}) = \begin{cases} g_i(\mathbf{x}) = (\mathbf{x}-\breve{\mathbf{x}}_i)^T(\breve{Q}_i)^{-1}(\mathbf{x}-\breve{\mathbf{x}}_i) \mid i = 1,\ldots,n+1 \\ g_{i+n+1}(\mathbf{x}) = -\mathbf{x}(i) \mid i = 1,\ldots,n \\ g_{i+2n+1}(\mathbf{x}) = \mathbf{x}(i) - lado^{\mathbf{C}} \mid i = 1,\ldots,n \end{cases}$$

onde $\tilde{\mathbf{x}}_j$ e $\breve{\mathbf{x}}_i \in \mathbf{C}$, $\tilde{Q}_j$ e $\breve{Q}_i \in \mathbb{R}^{n\times n}$ são matrizes simétricas definidas positivas. Os centros $\breve{\mathbf{x}}_i$ dos elipsóides e $\tilde{\mathbf{x}}_j$ são gerados de forma aleatória. As matrizes $\tilde{Q}_j$ são construídas com autovalores aleatórios. As matrizes $\breve{Q}_i$ são construídas com autovalores aleatórios de modo a garantir que $(\breve{E}_1 \cap \ldots \breve{E}_{n+1}) \neq \emptyset$, onde $\breve{E}_i$ representa o elipsóide definido pelo centro $\breve{\mathbf{x}}_i$ e pela matriz $\breve{Q}_i$.

Da mesma forma que definido para os problemas quasi-convexos, o elipsóide inicial E_0 foi definido de modo a garantir que $E_0 \supset \Omega$. Novamente, para criar as tabelas e gráficos de resultados, os problemas PCECN foram gerados para dimensões $n = \{5, 15, 25, 35\}$.

As tabelas 7.12 a 7.15 foram geradas a partir dos resultados obtidos dos algoritmos propostos $MPESC$ e $HISPE$ e dos algoritmos de referência $Shor$ e Kim. Também foram utilizadas duas configurações do parâmetro tamanho da busca histórica, ou seja, $3n$ e $kmax$.

Os resultados obtidos para todos os problemas testados são mostrados em valores percentuais nas figuras 7.5 e 7.6, tendo por base de referência os resultados do algoritmo $Shor$. Diferentemente da tendência observada na figura 7.1, a figura 7.5 não permite

Algoritmos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	1195	0.9042	317	2.27
Kim	374	0.8691	83	0.77
MPESC	657	0.8394	192	1.26
$HISPE_{15}$	86	0.5564	44	0.27
$HISPE_{kmax}$	76	0.5190	43	0.36

Tabela 7.12: Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQN de dimensão 5.

Algoritmos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	9843	0.9672	1501	149.81
Kim	3348	0.9608	459	48.90
MPESC	1584	0.8880	287	21.49
$HISPE_{45}$	447	0.5269	222	9.64
$HISPE_{kmax}$	383	0.4859	202	27.80

Tabela 7.13: Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQN de dimensão 15.

concluir que existe uma tendência de redução da razão dos resultados do número de acessos à função $f(\cdot)$ dos algoritmos *MPESC* e *HISPE* em referência aos resultados *Shor*. Todavia, também não existe uma tendência clara de aumento para a razão dos resultados dos algoritmos propostos, diferentemente do que é verificado para o algoritmo *Kim*. O perfil apresentado para os resultados do algoritmos *MPESC* e *HISPE* permitem pressupor que não haverá uma aproximação considerável do número de acessos à função $f(\cdot)$ destes algoritmos em relação aos número de acessos à função $f(\cdot)$ do algoritmo *Shor*, à medida que a dimensão dos problemas aumente. Já os resultados exibidos na figura 7.6 são bem próximos dos exibidos na figura 7.2. Esta figura também permite confirmar o melhor desempenho da configuração do algoritmo $HISPE_{3n}$ para o parâmetro tempo total para convergência, bem como a redução da razão deste parâmetro com referência aos respectivos resultados do algoritmo *Shor*. Na figura 7.2 para a configuração do algoritmo $HISPE_{kmax}$ a razão destes resultados, em relação aos respectivos resultados do algoritmo *Shor*, também apresenta tendência de redução. Já quanto aos resultados do método *Kim*, o perfil dos resultados apresentados apresenta uma pequena tendência de aumento. Por conseguinte, a diferença entre os resultados do algoritmo *HISPE* e os resultados do algoritmo *Kim* deve se tornar cada vez maior com o aumento da dimensão destes problemas.

Tal como realizado para os problemas PCEQC, para avaliar o desempenho do algo-

Algoritmos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	24644	0.9802	2931	1178.93
Kim	9203	0.9782	988	414.79
MPESC	4054	0.9335	570	166.05
$HISPE_{75}$	301	0.5744	141	23.36
$HISPE_{kmax}$	274	0.5467	135	114.66

Tabela 7.14: Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQN de dimensão 25.

Algoritmos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	46614	0.9858	2817	5872.12
Kim	17537	0.9852	1276	2046.21
MPESC	5551	0.9550	643	592.67
$HISPE_{105}$	539	0.6244	223	88.37
$HISPE_{kmax}$	435	0.5603	208	647.79

Tabela 7.15: Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQN de dimensão 35.

ritmo *HISPE* em função do tamanho da busca histórica a ser realizada, foram repetidos os testes dos problemas PCECN variando-se o tamanho da busca por valores históricos τ. A figura 7.7 exibe os resultados. Mais uma vez a heurística de definir um valor de $\tau \approx 3n$, para o tamanho da busca de valores já calculados demonstra sua eficácia.

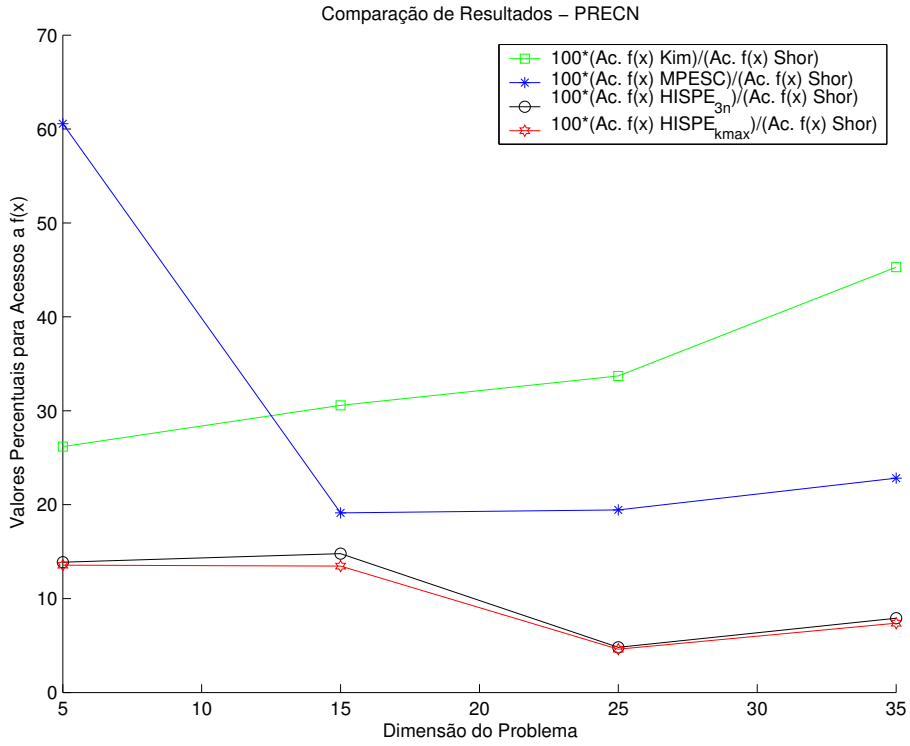


Figura 7.5: Resultados percentuais para o número de acessos à $f(\mathbf{x})$ dos algoritmos Kim, $MPESC$ e $HISPE$, em referência a $Shor$, para problemas PCECN.

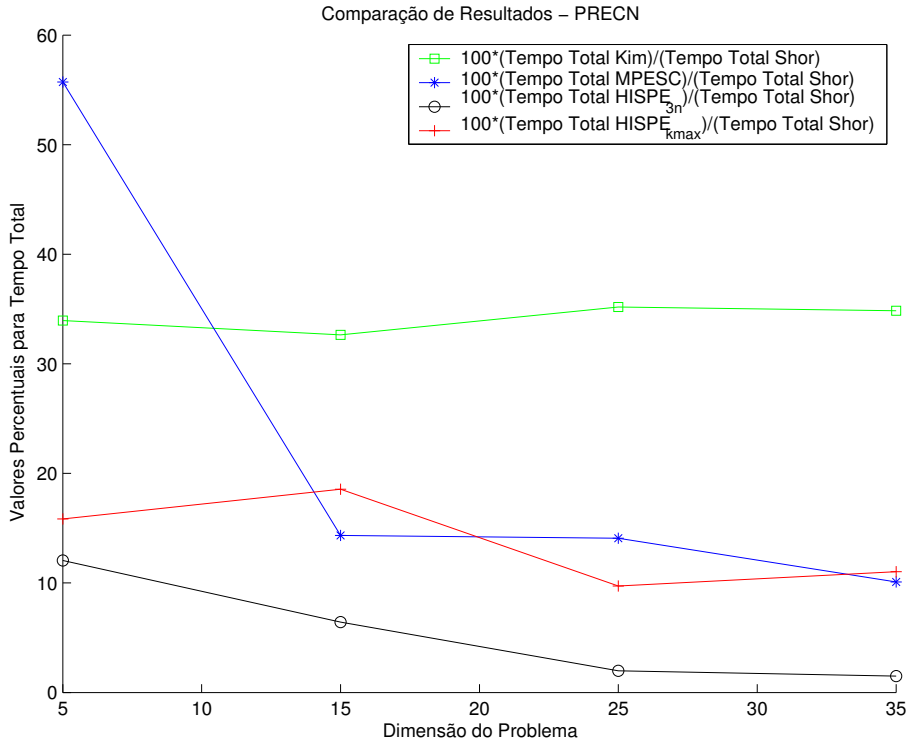
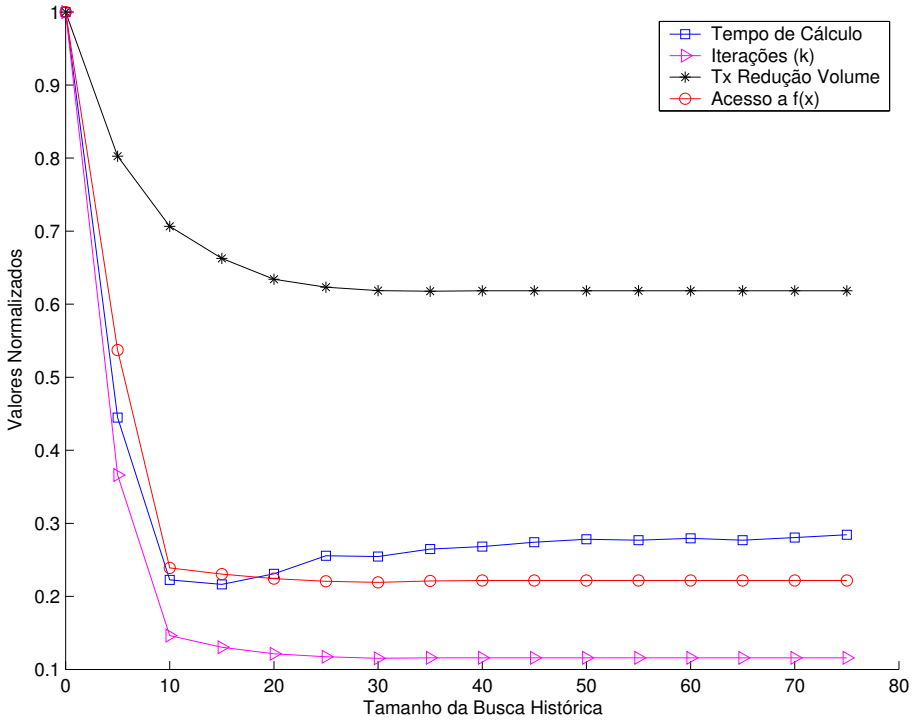
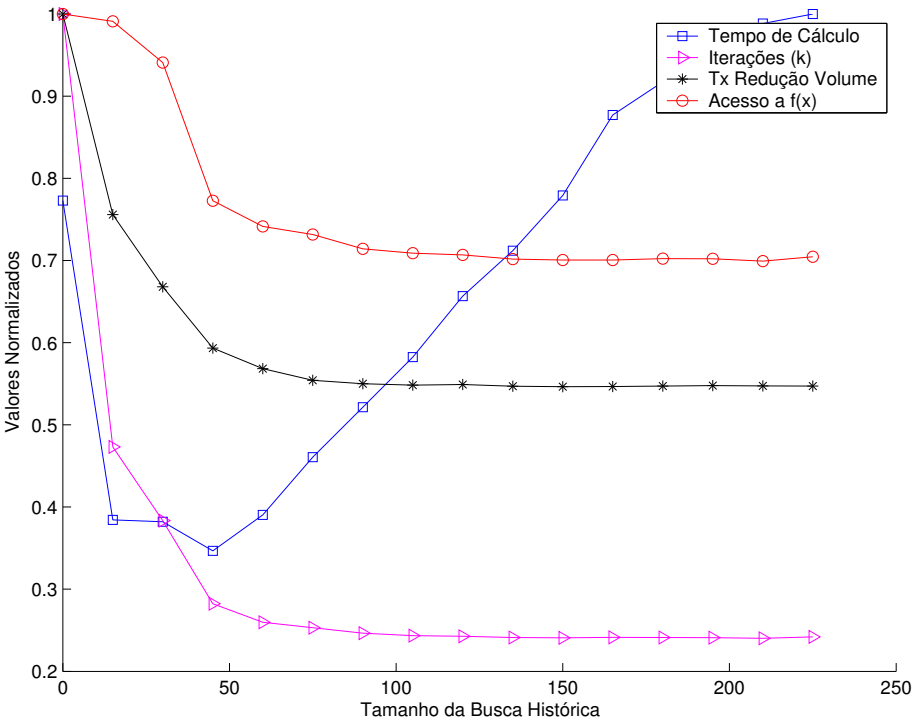


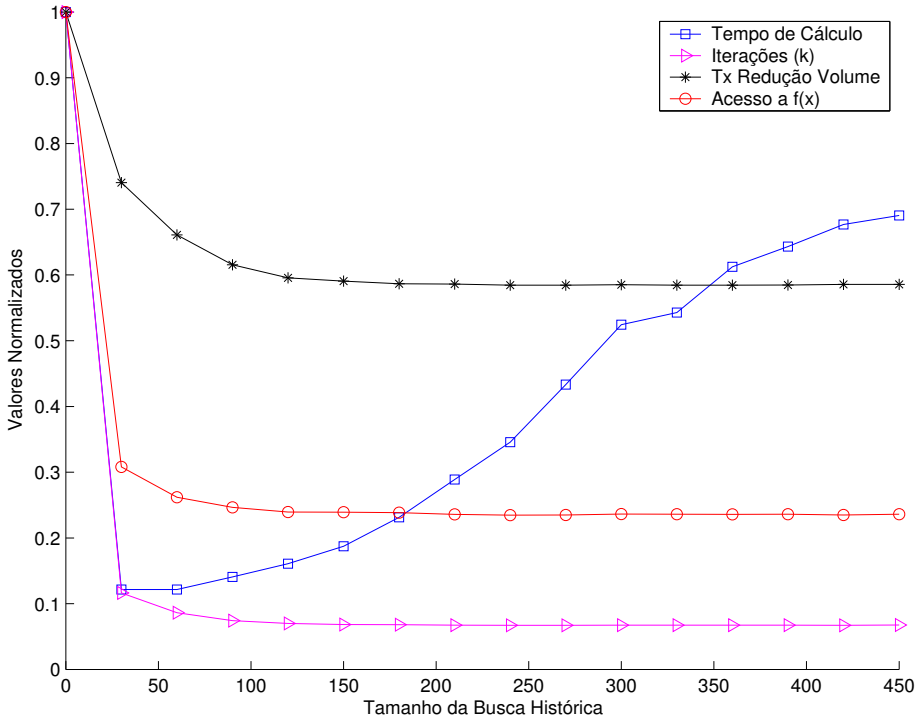
Figura 7.6: Resultados percentuais para o tempo total para convergência dos algoritmos Kim, $MPESC$ e $HISPE$, em referência a $Shor$, para problemas PCECN.



a) Problema PCECN de dimensão 5



b) Problema PCECN de dimensão 15



c) Problema PCECN de dimensão 25

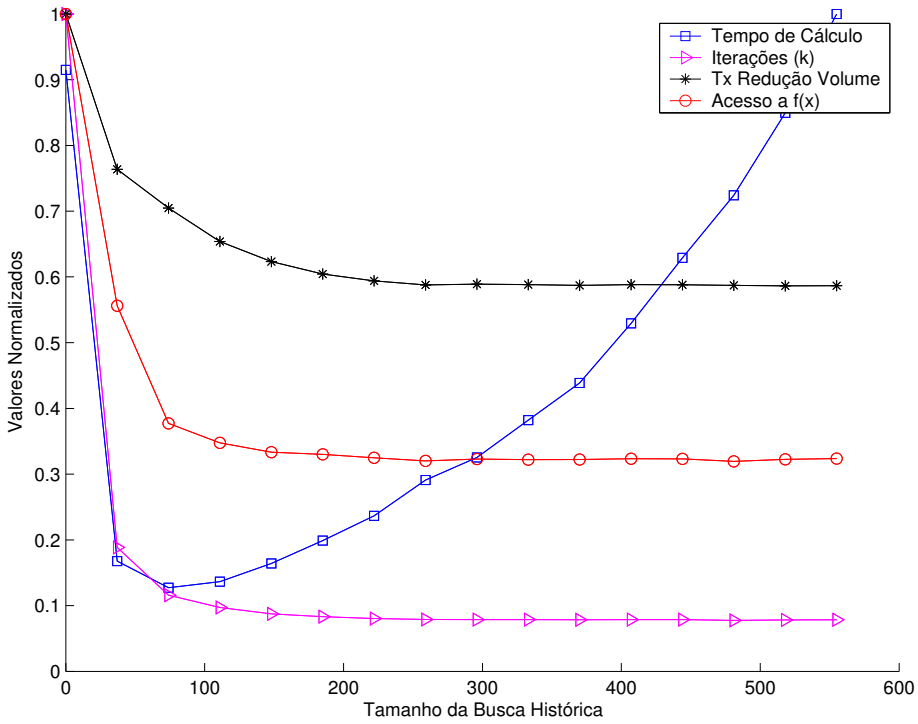


Figura 7.7: Desempenho do algoritmo HISPE mediante a variação do tamanho da busca histórica para o problema PCECN de dimensões 5, 15, 25 e 35.

7.1.3 Problema Convexo Diferenciável com Parâmetros Aleatórios

As seções anteriores deste capítulo mostraram que os métodos *Poliedro-Elipsoidais* foram efetivos em reduzir o tempo total para convergência de problemas escalares quasi-convexos e convexos não diferenciáveis. Então, para a continuidade da avaliação dos métodos propostos, também foram definidos problemas de teste escalares convexos diferenciáveis complementares, como será descrito a seguir.

7.1.3.1 Problema Quadrático com Restrições Quadráticas

O problema com variáveis contínuas, escalar e convexo diferenciáveis com função-objetivo quadrática e funções-restrição quadráticas é aqui denominado PCEQQ e representado por (7.4). Neste problema de teste todas as variáveis são restritas ao hipercubo $\mathbf{C} \subset \mathbb{R}^n_+$ e canto inferior esquerdo no ponto $\{-10, \ldots, -10\} \in \mathbb{R}^n$.

$$min\ f(\mathbf{x}) = (\mathbf{x} - \tilde{\mathbf{x}})^T (\tilde{Q})^{-1} (\mathbf{x} - \tilde{\mathbf{x}}) \tag{7.4}$$

$$s.a.\ \Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}^n\}$$

$$g(\mathbf{x}) = \begin{cases} g_i(\mathbf{x}) = (\mathbf{x} - \breve{\mathbf{x}}_i)^T (\breve{Q}_i)^{-1} (\mathbf{x} - \breve{\mathbf{x}}_i) \mid i = 1, \ldots, n+1 \\ g_{i+n+1}(\mathbf{x}) = -\mathbf{x}(i) \mid i = 1, \ldots, n \\ g_{i+2n+1}(\mathbf{x}) = \mathbf{x}(i) - lado^{\mathbf{C}} \mid i = 1, \ldots, n \end{cases}$$

onde $\tilde{\mathbf{x}}$ e $\breve{\mathbf{x}}_i \in \mathbf{C}$, $\tilde{Q}$ e $\breve{Q}_i \in \mathbb{R}^{n \times n}$ são matrizes simétricas definidas positivas. Os centros $\breve{\mathbf{x}}_i$ dos elipsóides e $\tilde{\mathbf{x}}$ são gerados de forma aleatória. A matriz $\tilde{Q}$ é construída com autovalores aleatórios. As matrizes $\breve{Q}_i$ são construídas com autovalores aleatórios de modo a garantir que $(\breve{E}_1 \cap \ldots \breve{E}_{n+1}) \neq \emptyset$, onde $\breve{E}_i$ representa o elipsóide definido pelo centro $\breve{\mathbf{x}}_i$ e pela matriz $\breve{Q}_i$.

Da mesma forma que definido para os problemas quasi-convexos, o elipsóide inicial E_0 foi definido de modo a garantir que $E_0 \supset \Omega$. Novamente, para criar as tabelas e gráficos de resultados, os problemas PCEQQ foram gerados para dimensões $n = \{5, 15, 25, 35\}$.

As tabelas 7.16 e 7.17 foram geradas a partir dos resultados obtidos dos algoritmos propostos *MPESC* e *HISPE* e dos algoritmos de referência *Shor* e *Kim*. Também foram utilizadas duas configurações do parâmetro tamanho da busca histórica, ou seja, $3n$ e $kmax$.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	3301	0.9672	2421	12.17
Kim	3304	0.9662	2448	12.63
MPESC	2251	0.9669	1672	3.62
$HISPE_{45}$	1551	0.9245	839	9.72
$HISPE_{kmax}$	639	0.8152	331	51.14

Tabela 7.16: Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQQ de dimensão 15.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	14939	0.9858	10750	338.92
Kim	16361	0.9858	11822	392.97
MPESC	5551	0.9858	3994	43.24
$HISPE_{105}$	4454	0.9455	2648	178.98
$HISPE_{kmax}$	2097	0.8632	1220	5791.00

Tabela 7.17: Resultados comparativos dos algoritmos *Shor*, *Kim*, *MPESC* e *HISPE* para um problema PCEQQ de dimensão 35.

Os resultados obtidos para todos os problemas testados são mostrados em valores percentuais nas figuras 7.8 e 7.9, tendo por base de referência os resultados do algoritmo *Shor*. A figura 7.8 permite uma melhor visualização da tendência de queda da razão dos resultados do número de acessos à função $f(\cdot)$ dos algoritmos *MPESC* e *HISPE* em referência aos resultados *Shor*, com o aumento da dimensão do problema. Porém, para os resultados do algoritmo *Kim*, a tendência de aumento da razão dos resultados deste algoritmo, em relação aos respectivos resultados do algoritmo *Shor*, é mais proeminente que a verificada nos problemas PCEQC e PCECN. Possivelmente este fato decorre da redução do EAIP para este problema, em relação ao EAIP dos problemas PCEQC e PCECN. Este mesmo fato parece determinar o perfil dos resultados apresentados na figura 7.9. Nesta figura, os resultados dos algoritmos *Kim*, *MPESC* e da configuração do algoritmo $HISPE_{3n}$ apresentam um perfil com tendência de queda. Já o resultado da configuração do algoritmo $HISPE_{kmax}$ apresentam um perfil com tendência de grande elevação. Considerando-se que os valores apresentados na figura 7.9 são valores percentuais, tendo por base de referência os resultados do algoritmo *Shor*, uma hipótese é reforçada. Esta hipótese pressupõe que à medida que o EAIP diminui, diminuem as possibilidades de que um método com maior ECAO possa produzir um menor ECT. Baseado neste raciocínio, o algoritmo *Kim* deve apresentar melhores resultados que o algoritmo *MPESC* para

problemas de baixo EAIP. Analogamente, o algoritmo $MPESC$ também deve apresentar melhores resultados que o algoritmo $HISPE$ para problemas de baixo EAIP.

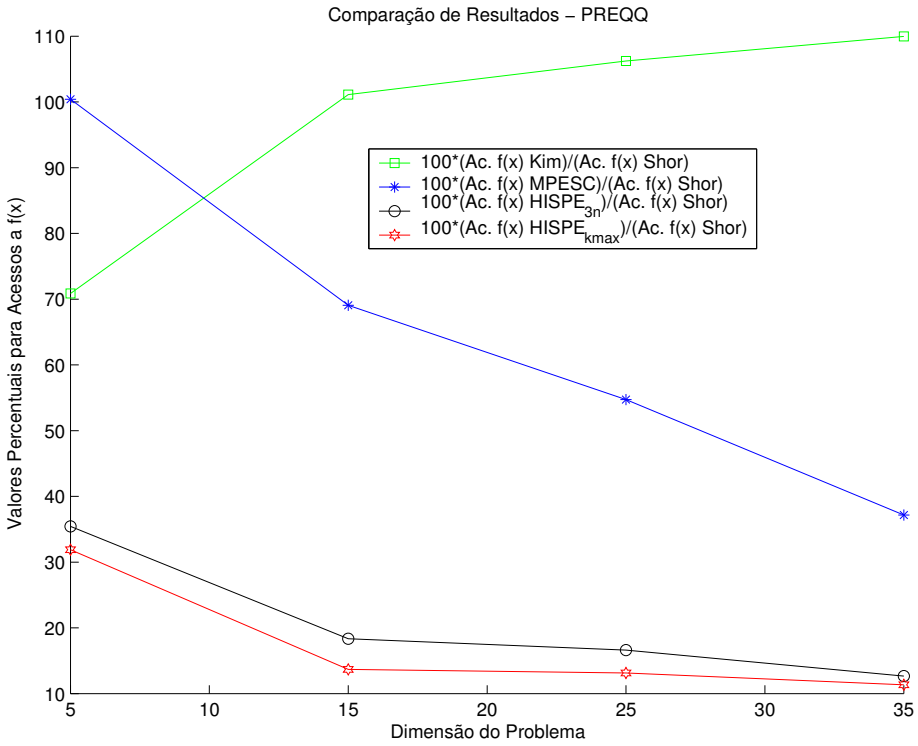


Figura 7.8: Resultados percentuais para o número de acessos à $f(\mathbf{x})$ dos algoritmos Kim, $MPESC$ e $HISPE$, em referência a $Shor$, para problemas PCEQQ.

Mais uma vez, para avaliar o desempenho do algoritmo $HISPE$ em função do tamanho da busca histórica a ser realizada, foram repetidos os testes dos problemas PCECN variando-se o parâmetro τ. Novamente, a heurística de definir um valor de $\tau \approx 3n$, para o tamanho da busca de valores já calculados, demonstra sua eficácia. A figura 7.10 exibe os resultados para problemas de dimensão 35.

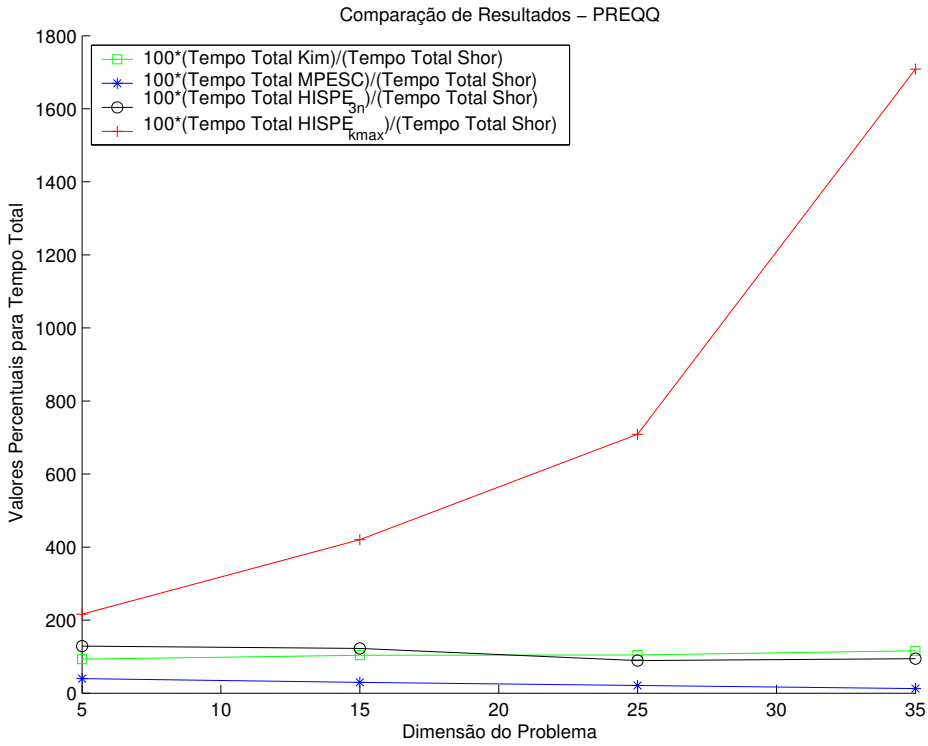


Figura 7.9: Resultados percentuais para o tempo total para convergência dos algoritmos Kim, $MPESC$ e $HISPE$, em referência a $Shor$, para problemas PCEQQ.

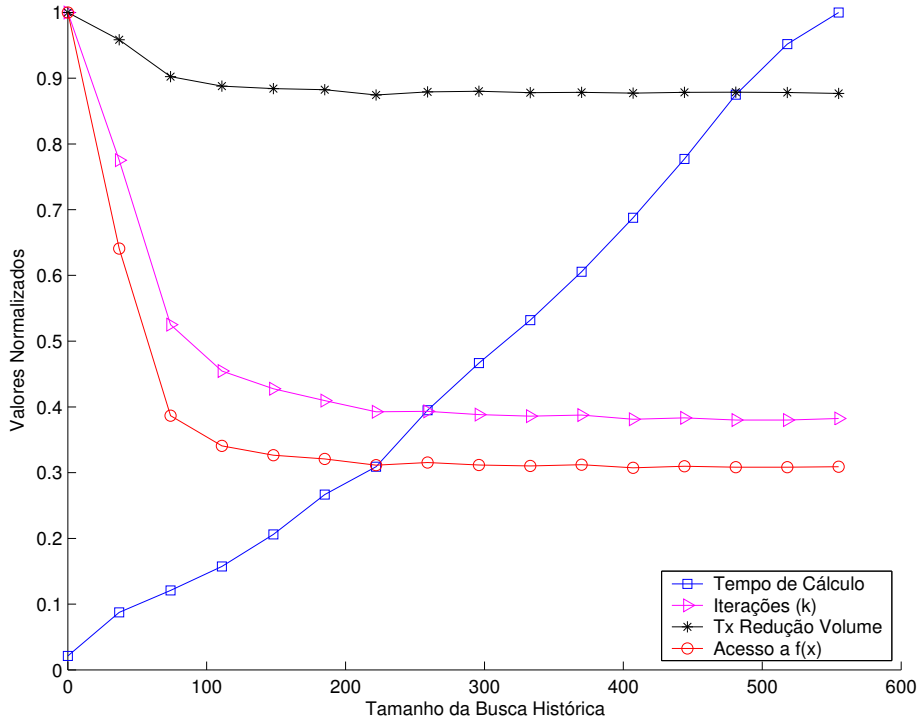


Figura 7.10: Desempenho do algoritmo HISPE mediante a variação do tamanho da busca histórica para o problema PEQQ de dimensão 35.

7.1.3.2 Problema Quadrático com Restrições Lineares

O problema com variáveis contínuas, escalar e convexo diferenciáveis com função-objetivo quadrática e funções-restrição lineares é aqui denominado PCEQL e representado por (7.5). Novamente, neste problema de teste todas as variáveis são restritas ao hipercubo $\mathbf{C} \subset \mathbb{R}^n_+$ e canto inferior esquerdo no ponto $\{-10, \ldots, -10\} \in \mathbb{R}^n$.

$$
\begin{aligned}
min\ f(\mathbf{x}) &= (\mathbf{x} - \tilde{\mathbf{x}})^T (\tilde{Q})^{-1} (\mathbf{x} - \tilde{\mathbf{x}}) \\
s.a.\ \Omega &= \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}^n\} \\
g(\mathbf{x}) &= \begin{cases} g_i(\mathbf{x}) = A(i,:)(\mathbf{x} - \mathbf{b}_i) \mid i = 1, \ldots, n+1 \\ g_{i+n+1}(\mathbf{x}) = -\mathbf{x}(i) \mid i = 1, \ldots, n \\ g_{i+2n+1}(\mathbf{x}) = \mathbf{x}(i) - lado^{\mathbf{C}} \mid i = 1, \ldots, n \end{cases}
\end{aligned}
\tag{7.5}
$$

onde $\mathbf{b}_i$ e $\tilde{\mathbf{x}} \in \mathbf{C}$, $A^i \in \mathbb{R}^{(p+1)\times n}$ e $\tilde{Q} \in \mathbb{R}^{n\times n}$. O centro $\tilde{\mathbf{x}}$ do elipsóide e os pontos $\mathbf{b}_i$ são gerados de forma aleatória. A matriz $\tilde{Q}$ é construída com autovalores aleatórios. A matriz A é uma matriz simétrica definida positiva e é construída com valores aleatórios de modo a garantir que $(H_-^{A(:,1),\mathbf{b}_1} \cap \ldots H_-^{A(:,n+1),\mathbf{b}_{n+1}}) \neq \emptyset$.

Da mesma forma que definido para os problemas das seções anteriores, o elipsóide inicial E_0 foi definido de modo a garantir que $E_0 \supset \Omega$. Novamente, para criar as tabelas e gráficos de resultados, os problemas PCEQL foram gerados para dimensões $n = \{5, 15, 25, 35\}$. As tabelas 7.18 e 7.19 foram geradas a partir dos resultados obtidos dos algoritmos propostos $MPESC$ e $HISPE$ e dos algoritmos de referência $Shor$ e Kim. Mais uma vez foram utilizadas duas configurações do parâmetro tamanho da busca histórica, ou seja, $3n$ e $kmax$.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
$Shor$	3441	0.9672	2972	11.06
Kim	3108	0.9651	2695	10.20
$MPESC$	2551	0.9670	2213	3.53
$HISPE_{45}$	1352	0.9151	906	8.92
$HISPE_{kmax}$	536	0.8092	272	32.50

Tabela 7.18: Resultados comparativos dos algoritmos $Shor$, Kim, $MPESC$ e $HISPE$ para um problema PCEQL de dimensão 15.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
$Shor$	15471	0.9858	13703	317.70
Kim	13100	0.9851	11617	240.99
$MPESC$	10501	0.9856	9304	80.29
$HISPE_{105}$	3952	0.9395	2912	149.72
$HISPE_{kmax}$	2082	0.8920	1118	3996.58

Tabela 7.19: Resultados comparativos dos algoritmos $Shor$, Kim, $MPESC$ e $HISPE$ para um problema PCEQL de dimensão 35.

Os resultados obtidos para todos os problemas testados são mostrados em valores percentuais nas figuras 7.11 e 7.12, tendo por base de referência os resultados do algoritmo $Shor$. Deve-se observar que os resultados exibidos nas figuras 7.11 e 7.12 são bem semelhantes aos discutidos para o problema PCEQQ, apresentados nas figuras 7.8 e 7.9. As principais diferenças estão nos perfis dos resultados obtidos pelos algoritmos Kim e $MPESC$ exibidos na figura 7.11. Para os problemas PCEQL, os perfis dos algoritmos Kim e $MPESC$ passam a apresentar uma tendência de estabilização da razão dos resultados do número de acessos à função $f(\cdot)$, em referência aos resultados $Shor$, com o aumento da dimensão. Novamente, a hipótese que pressupõe que à medida que o EAIP diminui, diminuem as possibilidades de que um método com com maior ECAO possa produzir um menor ECT, também pode ser observada na figura 7.12.

Mais uma vez, para validar a heurística de definir um valor de $\tau \approx 3n$, para o tamanho da busca de valores já calculados, avaliou-se o desempenho do algoritmo $HISPE$ em função do τ. A figura 7.10 exibe os resultados para problemas PCEQL de dimensão 35.

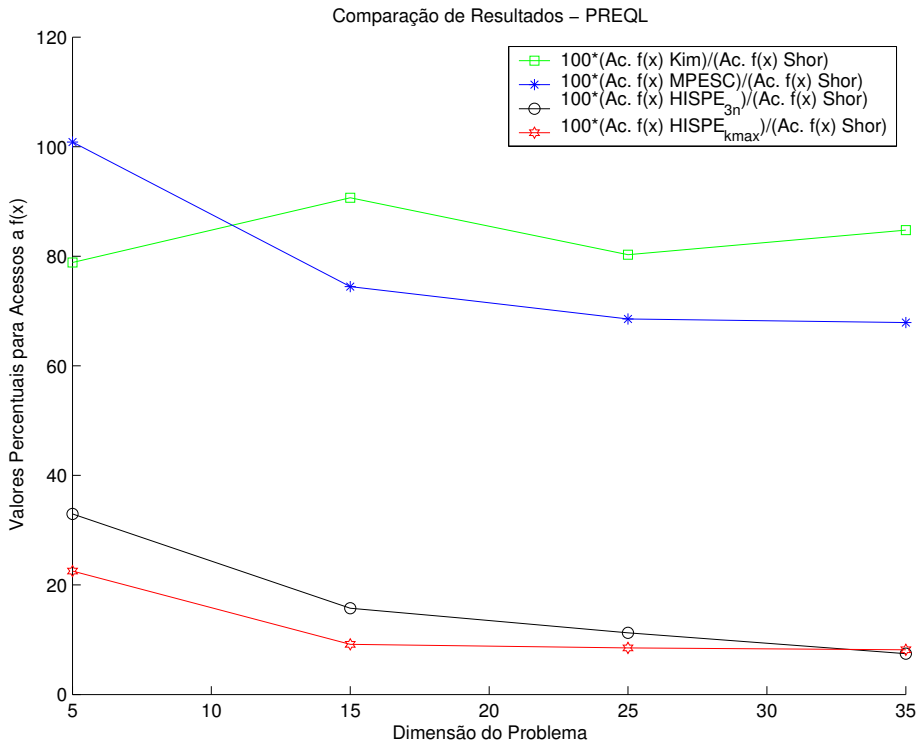


Figura 7.11: Resultados percentuais para o número de acessos à $f(\mathbf{x})$ dos algoritmos Kim, $MPESC$ e $HISPE$, em referência a $Shor$, para problemas PCEQL.

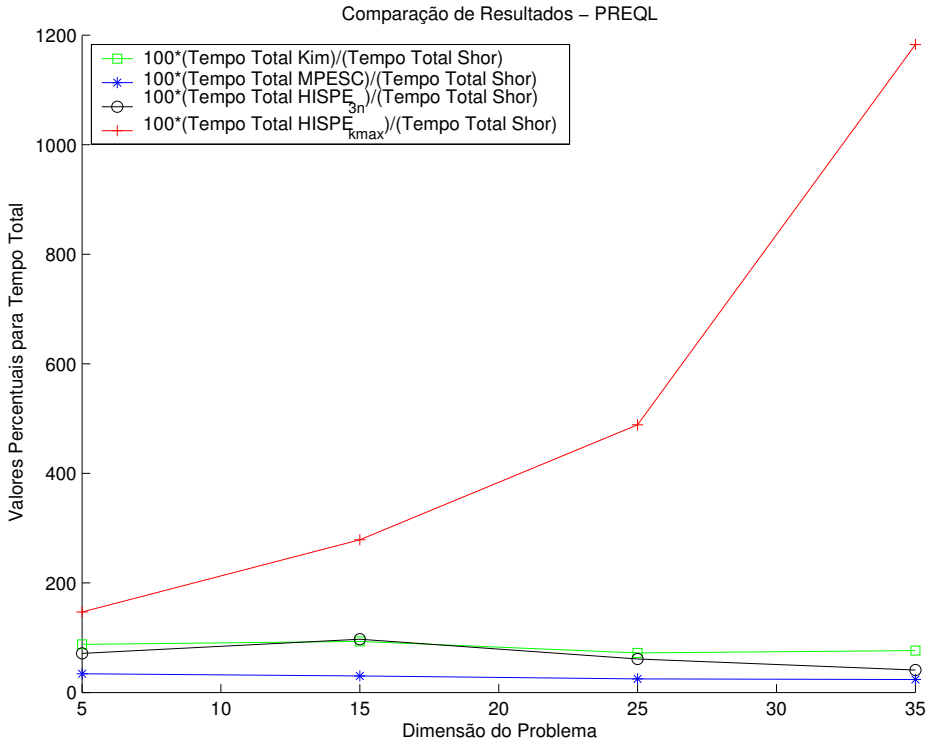


Figura 7.12: Resultados percentuais para o tempo total para convergência dos algoritmos Kim, $MPESC$ e $HISPE$, em referência a $Shor$, para problemas PCEQL.

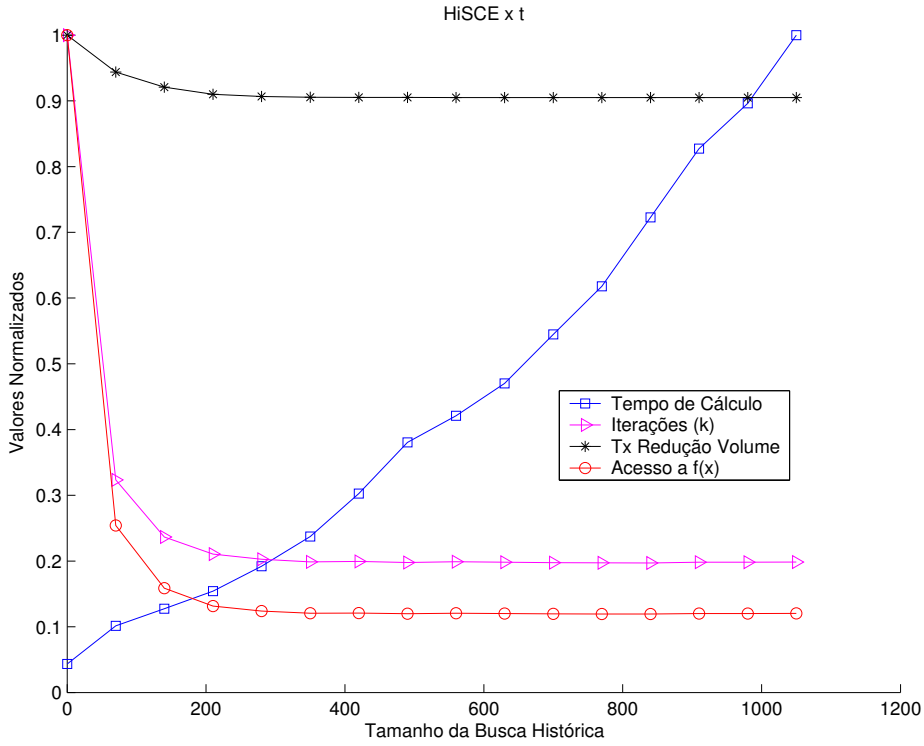


Figura 7.13: Desempenho do algoritmo HISPE mediante a variação do tamanho da busca histórica para o problema PEQL de dimensão 35.

7.1.4 Problemas com Variáveis Relaxadas para Processo Químico de Sintetização

Como será discutido detalhadamente nos capítulos 8, 9, os problemas com variáveis não exclusivamente contínuas precisam ser resolvidos com a relaxação das variáveis que devem ser discretas tomadas como variáveis contínuas. Este processo faz parte de diversos métodos de otimização discreta. Conseqüentemente, estes problemas com variáveis relaxadas podem ser utilizados como casos de teste quasi-convexos com variáveis contínuas.

Dois problemas referentes ao processo químico de sintetização (Duran & Grossmann 1986) são representados por (7.6) e (7.7). Estes problemas representam casos reais de problemas de engenharia para a classe de problemas escalares quasi-convexos, apesar do fato dos problemas serem realmente pseudo-convexos.

7.1.4.1 Processo de Sintetização 1 com Variáveis Relaxadas

O primeiro problema associado a um processo químico de sintetização é apresentado a seguir com variáveis relaxadas, ou seja, exclusivamente contínuas.

$$\begin{aligned} min\ f(\mathbf{x}) \ &= \ 5\mathbf{x}(1) + 6\mathbf{x}(2) + 8\mathbf{x}(3) + 10\mathbf{x}(4) - 18ln(\mathbf{x}(5)+1) + \\ &\quad -7\mathbf{x}(6) - 19.2ln(\mathbf{x}(4) - \mathbf{x}(5) + 1) + 10 \\ &\quad s.a.\ \Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}_+^6\} \end{aligned} \tag{7.6}$$

$$g(\mathbf{x}) \triangleq \begin{cases} g_1(\mathbf{x}) = -0.8ln(\mathbf{x}(5)+1) - 0.96ln(\mathbf{x}(4) - \mathbf{x}(5) + 1) + 0.8\mathbf{x}(6) \leq 0 \\ -ln(\mathbf{x}(5)+1) - 1.2 * ln(\mathbf{x}(4) - \mathbf{x}(5) + 1) + \mathbf{x}(6) + 2 * \mathbf{x}(3) - 2 \leq 0 \\ g_2(\mathbf{x}) = \mathbf{x}(5) - \mathbf{x}(4) \leq 0 \\ g_3(\mathbf{x}) = \mathbf{x}(5) - 2\mathbf{x}(1) \leq 0 \\ g_4(\mathbf{x}) = \mathbf{x}(4) - \mathbf{x}(5) - 2\mathbf{x}(2) \leq 0 \\ g_5(\mathbf{x}) = \mathbf{x}(1) + \mathbf{x}(2) \leq 1 \\ g_{5+i} = -\mathbf{x}(i) \mid i = 1, \ldots, 6 \\ g_{11+i} = \mathbf{x}(i) - 2 \mid i = 1, 2 \\ g_{14} = \mathbf{x}(6) - 1 \end{cases}$$

O elipsóide inicial E_0 foi definido de modo a garantir que $E_0 \supset \Omega$. A tabela 7.20 exibe os resultados para o problema de sintetização 1 alcançados pelos algoritmos $MPESC$, $HISPE$, $Shor$ e Kim, tal como nos testes apresentados nas seções anteriores.

Os resultados apresentados na tabela 7.20 deixam claro que os métodos *Poliedro-*

Algoritmos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
Shor	1604	0.9308	1630	2.05
Kim	785	0.9191	799	1.06
MPESC	725	0.8926	745	1.13
$HISPE_{21}$	193	0.5558	204	7.84
$HISPE_{kmax}$	142	0.4801	152	11.67

Tabela 7.20: Resultados comparativos dos métodos *Shor*, *Kim*, *MPESC* e *HISPE* do problema de sintetização 1.

Elipsoidais conseguiram reduzir a região de busca durante o processo de convergência. Comparando-se os resultados dos parâmetros número de iterações, taxa de redução do volume do elipsóide e número de acessos a $f(\cdot)$, é notório o aumento da eficiência em reduzir a região de busca nas iterações com a utilização de um maior número de informações históricas. Entretanto, como o modelo deste processo de síntese é baseado em equações analíticas de baixa complexidade e dimensão, o aumento do ECAO não foi compensado pela redução do EAIP. Assim, na ordem direta em que aumentou a eficiência em reduzir a região de busca, também aumentou o tempo total para convergência. Cabe ressaltar que, apesar da simplicidade do modelo, o algoritmo *MPESC* convergiu em um tempo 55% inferior ao algoritmo *Shor*. Todavia, o algoritmo *MPESC* apresentou um tempo total para convergência similar ao obtido pelo algoritmo *Kim*, mas superior. Deve-se ainda observar que, para problemas desta mesma classe e cujos modelos exijam maior ECAO, provavelmente os métodos *Poliedro-Elipsoidais* apresentariam menor tempo total de convergência.

7.1.4.2 Processo de Sintetização 2 com Variáveis Relaxadas

O segundo problema associado a um processo químico de sintetização é apresentado a seguir com variáveis relaxadas.

$$\begin{aligned} min\ f(\mathbf{x}) \ &= \ 5\mathbf{x}(1) + 8\mathbf{x}(2) + 6\mathbf{x}(3) + 10\mathbf{x}(4) + 6\mathbf{x}(5) + \\ &-10\mathbf{x}(6) - 15\mathbf{x}(7) + 15\mathbf{x}(8) + 5\mathbf{x}(9) - 15\mathbf{x}(10) - 20\mathbf{x}(11) \\ &s.a.\ \Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}_+^{11}\} \end{aligned} \tag{7.7}$$

$$\Omega \triangleq \begin{cases} -ln(\mathbf{x}(8)+\mathbf{x}(9)+1) \leq 0 \\ -\mathbf{x}(6)-\mathbf{x}(7)-\mathbf{x}(10)+\mathbf{x}(9)+2\mathbf{x}(11) \leq 0 \\ -\mathbf{x}(6)-\mathbf{x}(7)-0.75\mathbf{x}(10)+\mathbf{x}(8)+2\mathbf{x}(11) \leq 0 \\ \mathbf{x}(10)-\mathbf{x}(11) \leq 0 \\ 2\mathbf{x}(10)-\mathbf{x}(8)-2\mathbf{x}(11) \leq 0 \\ -0.5\mathbf{x}(8)+\mathbf{x}(9) \leq 0 \\ 0.2\mathbf{x}(8)-\mathbf{x}(9) \leq 0 \\ e^{\mathbf{x}(6)}-10\mathbf{x}(1) \leq 0 \\ e^{\mathbf{x}(7)/1.2}-10\mathbf{x}(2) \leq 0 \\ 1.25\mathbf{x}(10)-10\mathbf{x}(8) \leq 0 \\ -2\mathbf{x}(8)+2\mathbf{x}(11)-10\mathbf{x}(5) \leq 0 \\ \mathbf{x}(1)+\mathbf{x}(2)-1=0 \\ \mathbf{x}(4)+\mathbf{x}(5)-1 \leq 0 \\ 0 \leq \mathbf{x}(6) \leq 2 \\ 0 \leq \mathbf{x}(7) \leq 2 \\ 0 \leq \mathbf{x}(8) \\ 0 \leq \mathbf{x}(9) \\ 0 \leq \mathbf{x}(10) \leq 2 \\ 0 \leq \mathbf{x}(11) \leq 3 \end{cases}$$

Similarmente ao realizado para o problema 1, o elipsóide inicial E_0 foi definido de modo a garantir que $E_0 \supset \Omega$. A tabela 7.21 exibe os resultados para o problema de sintetização 2 alcançados pelos algoritmos pelos algoritmos $MPESC$, $HISPE$, $Shor$ e Kim

Algoritmos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total
$Shor$	3838	0.9555	4184	10.09
Kim	1718	0.9483	1941	4.55
$MPESC$	1541	0.9172	4695	4.19
$HISPE_{33}$	386	0.6638	2860	56.25
$HISPE_{kmax}$	221	0.4924	2306	89.97

Tabela 7.21: Resultados comparativos dos métodos $Shor$, Kim, $MPESC$ e $HISPE$ do problema de sintetização 2.

A argumentação, apresentada para o problema de sintetização 1, na qual é esperado dos métodos *Poliedro-Elipsoidais* uma melhoria do tempo de convergência total mediante o aumento do ECAO é reforçada pelos resultados apresentados na tabela 7.21. Embora o problema de sintetização 2 continue sendo definido por equações analíticas de baixa complexidade, o aumento da dimensão no problema de sintetização 2 foi suficiente para a melhora dos resultados apresentados pelos algoritmos *Poliedro-Elipsoidais*. Comparando-se os resultados dos parâmetros número de iterações, taxa de redução do volume do elipsóide e número de acessos a $f(\cdot)$, novamente é notório o aumento da eficiência em reduzir a região de busca nas iterações com o aumento da utilização de informações históricas. Por fim, deve ainda ser ressaltado que os valores percentuais para o tempo total para convergência também melhoraram, sendo que, para este teste, o melhor desempenho foi apresentado pelo algoritmo $MPESC$.

7.2 Problemas Vetoriais com Variáveis Contínuas

Três classes de problemas vetoriais foram utilizadas para avaliar o desempenho dos algoritmos propostos. A primeira classe é composta por problemas QCND de diversas dimensões nos quais os parâmetros foram gerados de forma aleatória. Nesta classe de problemas também foi testada a subclasse de problemas convexos diferenciáveis. A segunda classe de problemas é composta por problemas de Engenharia de Controle. A terceira classe de problemas é composta por problemas de factibilidade. A definição destes problemas e os resultados alcançados são exibidos nas próximas seções.

7.2.1 Problema Quasi-Convexo Não Necessariamente Diferenciável com Parâmetros Aleatórios

O problema contínuo, vetorial e quasi-convexo não necessariamente diferenciável é aqui denominado PCVQC e representado por (7.8). Todas as variáveis são restritas ao hipercubo $\mathbf{C} \subset \mathbb{R}^n_+$ e canto inferior esquerdo no ponto $\{0, \ldots, 0\} \in \mathbb{R}^n$.

$$min\ f(\mathbf{x}) = [\ f_1(\mathbf{x})\ \ldots\ f_m(\mathbf{x})\ ]^T; \tag{7.8}$$

$$s.a.\ \Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}^n\}$$

$$f_j(\mathbf{x}) = max(\{-(1 + e^{(\lambda_j(t)(\mathbf{x}(t) - \tilde{\mathbf{x}}_j(t)))^2})\}) \mid t = 1, \ldots, n \mid \forall\ j$$

$$g(\mathbf{x}) = \begin{cases} g_i(\mathbf{x}) = (\mathbf{x} - \breve{\mathbf{x}}_i)^T (\breve{Q}_i)^{-1} (\mathbf{x} - \breve{\mathbf{x}}_i) \mid i = 1, \ldots, n+1 \\ g_{i+n+1}(\mathbf{x}) = -\mathbf{x}(i) \mid i = 1, \ldots, n \\ g_{i+2n+1}(\mathbf{x}) = \mathbf{x}(i) - lado^{\mathbf{C}} \mid i = 1, \ldots, n \end{cases}$$

onde $\tilde{\mathbf{x}}_j$ e $\breve{\mathbf{x}}_i \in \mathbf{C}$, $\lambda_j \in \mathbb{R}^n \mid 0 < \lambda_j(t) < 10^{-1}$, e $\breve{Q}_i \in \mathbb{R}^{n \times n}$ é uma matriz simétrica definida positiva. Os centros $\breve{\mathbf{x}}_i$ dos elipsóides, os mínimos irrestritos $\tilde{\mathbf{x}}_j$ e os multiplicadores λ_j são gerados de forma aleatória. As matrizes $\breve{Q}_i$ são construídas com autovalores aleatórios de modo a garantir que $(\breve{E}_1 \cap \ldots \breve{E}_{n+1}) \neq \emptyset$, onde $\breve{E}_i$ representa o elipsóide definido pelo centro $\breve{\mathbf{x}}_i$ e pela matriz $\breve{Q}_i$.

Para criar as tabelas e gráficos de resultados, os problemas PCVQC foram gerados para dimensões $n = \{5, 15, 25\}$ e para $m = 2$. Deve-se observar que foram utilizadas duas configurações do parâmetro tamanho da busca histórica, ou seja, $3n$ e $kmax$. A partir dos resultados obtidos dos algoritmos propostos $MPESC$ e $HISPE$ foram geradas as tabelas 7.22 a 7.24.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total	% Soluções Não-Dominadas
$MPESC$	924	0.9042	923	1.43	100.00
$HISPE_{15}$	37	0.3051	38	0.25	100.00
$HISPE_{kmax}$	37	0.3054	38	0.28	100.00

Tabela 7.22: Resultados comparativos dos algoritmos $MPESC$ e $HISPE$ para um problema PCVQC de dimensão 5.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total	% Soluções Não-Dominadas
$MPESC$	8380	0.9672	8379	42.92	100.00
$HISPE_{45}$	352	0.4733	351	11.28	100.00
$HISPE_{kmax}$	342	0.4653	341	18.96	100.00

Tabela 7.23: Resultados comparativos dos algoritmos $MPESC$ e $HISPE$ para um problema PCVQC de dimensão 15.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total	% Soluções Não-Dominadas
$MPESC$	20639	0.9802	20638	235.54	100.00
$HISPE_{75}$	288	0.5393	289	26.18	100.00
$HISPE_{kmax}$	284	0.5370	284	45.36	100.00

Tabela 7.24: Resultados comparativos dos algoritmos $MPESC$ e $HISPE$ para um problema PCVQC de dimensão 25.

Os resultados apresentados nas tabelas 7.22 a 7.24 reforçam algumas observações já realizadas para os problemas escalares apresentados na seção 7.1. Tal como esperado, a utilização de informações já calculadas permitiu a redução da região de interesse de busca durante o processo de convergência do algoritmo $HISPE$. Assim, pode-se verificar nas tabelas que os parâmetros número de iterações, taxa de convergência e número de acessos a $f(\cdot)$ apresentaram melhores resultados à medida que aumentou-se o valor do parâmetro τ, relacionado ao tamanho da busca histórica. Conseqüentemente, para estes três parâmetros, temos que os resultados do algoritmo $HISPE$ com a configuração $\tau = kmax$ foram melhores que os do mesmo algoritmo com a configuração $\tau = 3n$ que, por sua vez, também foram melhores que os resultados do algoritmo $MPESC$. Quanto ao parâmetro tempo total de convergência, novamente a heurística de definir o parâmetro $\tau = 3n$ como melhor custo benefício entre EAIP e ECAO se mostrou adequada. Em todos os casos, esta configuração do algoritmo $HISPE$ obteve os menores tempos totais para convergência. Deve-se ainda ressaltar que o algoritmo $HISPE$ com a configuração $\tau = kmax$ apresentou, para todos os problemas testados, menor tempo total para convergência do que o tempo obtido pelo algoritmo $MPESC$. Este resultado mais uma vez reforça a hipótese de que o algoritmo $HISPE$ produz melhores resultados que o algoritmo $MPESC$ à medida que o EAIP aumenta. Estas mesmas observações podem ser verificadas graficamente nas figuras 7.14 e 7.15 mostradas a seguir.

Por fim, dada a utilização de parâmetros gerados aleatoriamente, não é possível prédefinir se um ponto factível pertence ou não ao conjunto *Pareto-Ótimo*. Assim, o critério adotado para avaliar se uma solução encontrada por um método pertence ao conjunto *Pareto-Ótimo* foi determinar se a solução em questão não é dominada por algum ponto avaliado por qualquer outro algoritmo que também solucionou o problema. Este critério, quantificado pelo parâmetro percentual de soluções não-dominadas, permite concluir que, em todos os testes realizados, os algoritmos propostos encontraram 100% de soluções não-dominadas.

A figura 7.16 exibe os pontos avaliados pelos algoritmos $MPESC$ e $HISPE$ para um PCVQC testado com $n = 5$ e $m = 2$. Durante a convergência dos métodos, verifica-se que as duas configurações do algoritmo $HISPE$ convergiram para a mesma solução, enquanto o algoritmo $MPESC$ convergiu para uma solução mais afastada. Todavia, em todos os casos as soluções encontradas não são dominadas por qualquer outro ponto avaliado por qualquer dos outros algoritmos. Esta afirmação é mais perceptível pela observação de que não existe qualquer ponto no interior da região demarcada por linhas pontilhadas a partir das soluções encontradas pelos algoritmos. Assim, verifica-se que as soluções encontradas são pontos não-dominados por quaisquer dos outros pontos exibidos na figura 7.16.

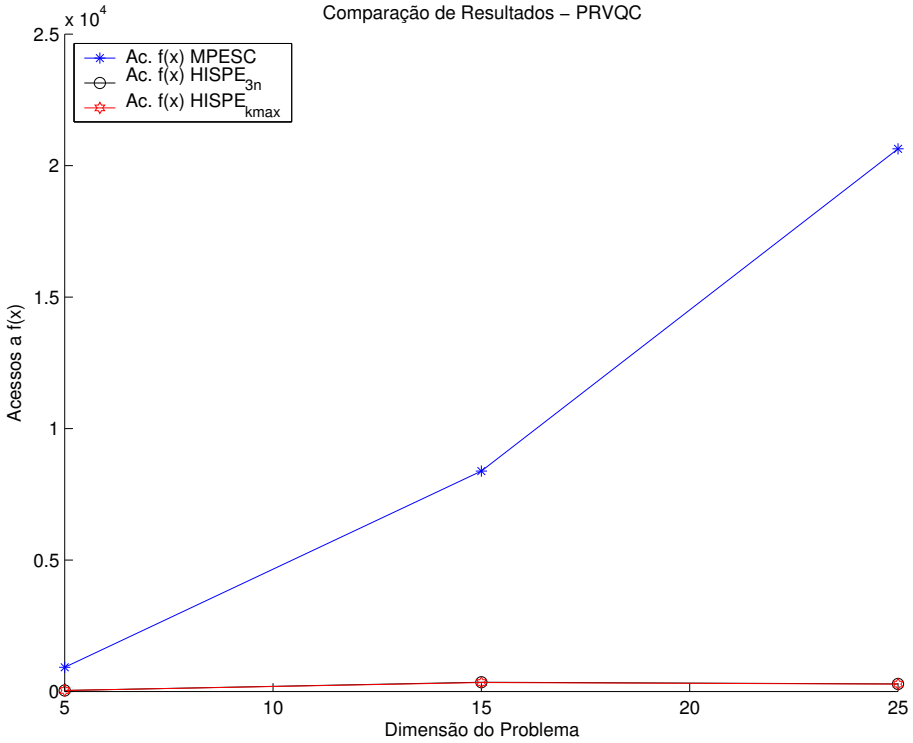


Figura 7.14: Resultados para o número de acessos à $f(\mathbf{x})$ dos algoritmos $MPESC$ e $HISPE$ para problemas PCVQC.

Uma vez demonstrada na seção 7.1 a validade da heurística proposta para o parâmetro de busca histórica τ, não foram realizados os testes de desempenho do algoritmo $HISPE$ em função da variação do parâmetro τ da busca histórica.

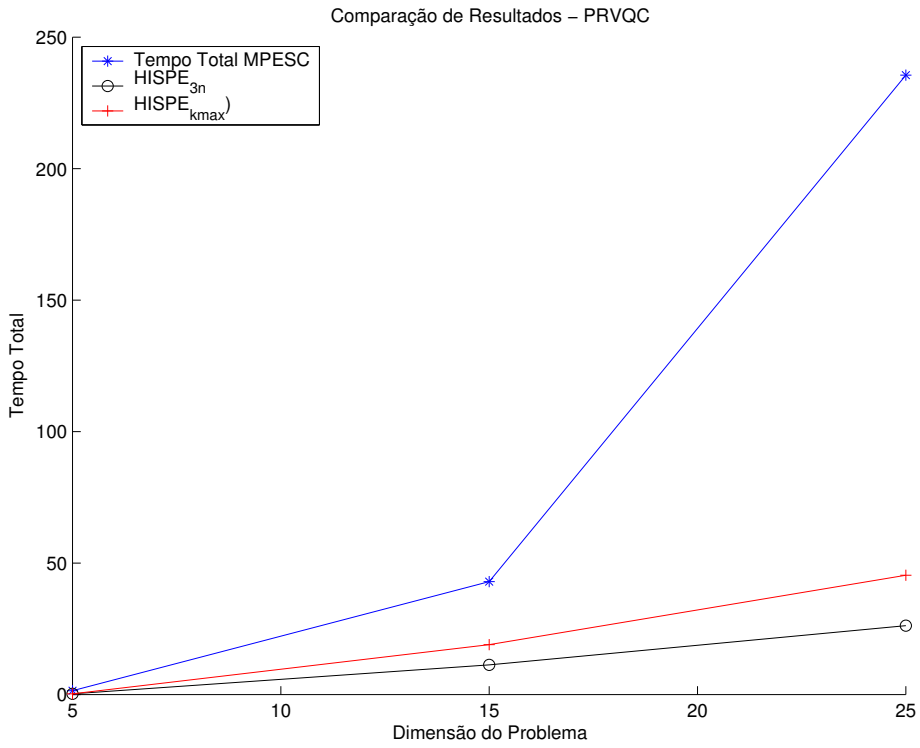


Figura 7.15: Resultados para o tempo total para convergência dos algoritmos $MPESC$ e $HISPE$ para problemas PCVQC.

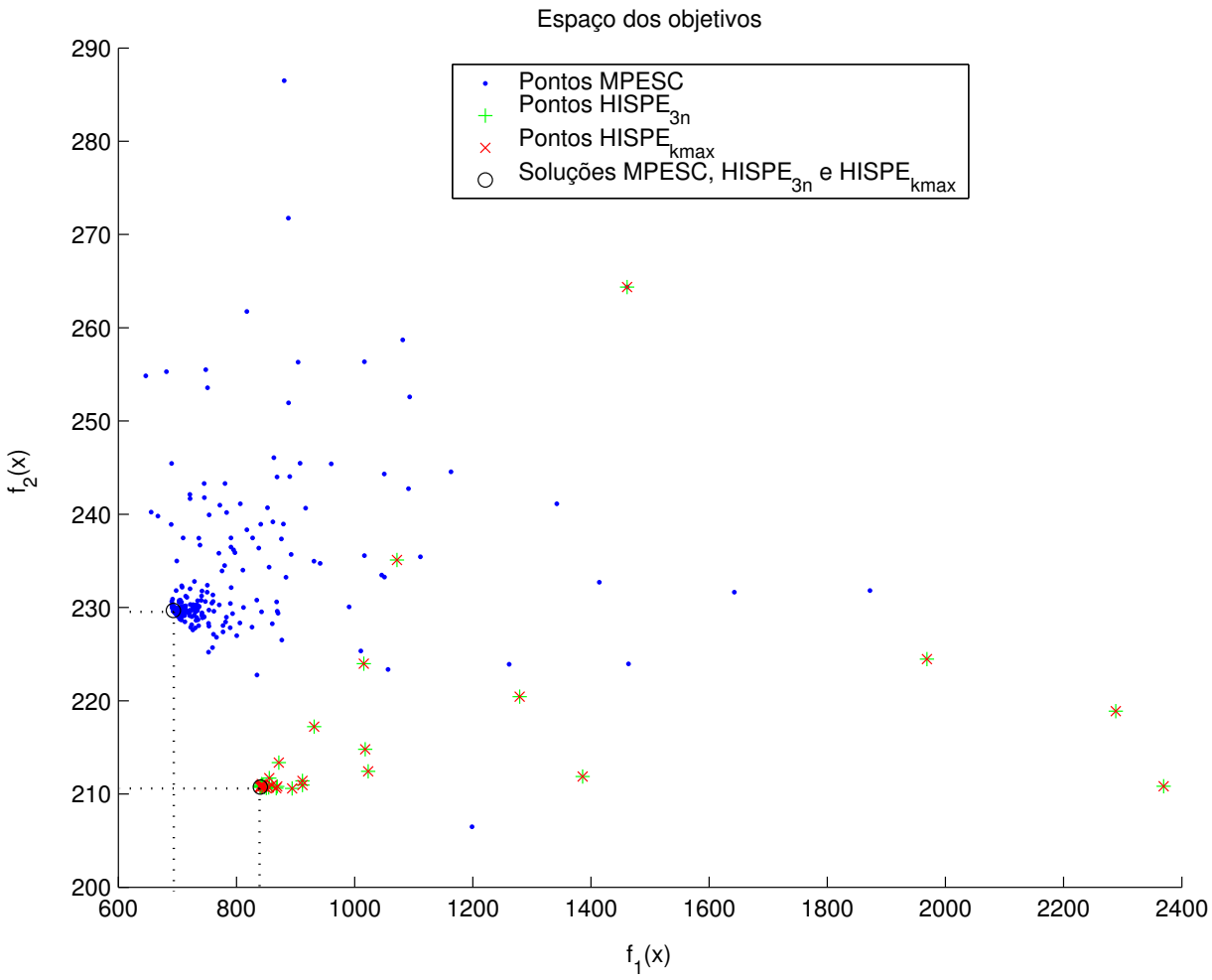


Figura 7.16: Conjunto dos pontos avaliados durante a convergência dos métodos $MPESC$ e $HISPE$ para um problema PCVQC, $m = 2$.

7.2.1.1 Problema Quadrático Irrestrito

Com o objetivo de avaliar por uma métrica diferente da apresentada na seção anterior a capacidade dos algoritmos $MPESC$ e $HISPE$ encontrar pontos não-dominados, foi proposto o problema com variáveis contínuas vetorial quadrático irrestrito, aqui denominado por PCVQI, é representado por (7.9).

$$min\ f(\mathbf{x}) = [\ f_1(\mathbf{x})\ \ldots\ f_m(\mathbf{x})\ ]^T :\ |:\ \mathbf{x} \in \mathbf{R}^n; \tag{7.9}$$
$$f_j(\mathbf{x}) = (\mathbf{x} - \tilde{\mathbf{x}}_j)^T(\tilde{Q}_j)^{-1}(\mathbf{x} - \tilde{\mathbf{x}}_j)\ |\ \forall\ j$$

onde $\tilde{\mathbf{x}}_j \in \mathbf{C}$, $\tilde{Q}_j \in \mathbb{R}^{n\times n}$ são matrizes simétricas definidas positivas. Os centros $\breve{\mathbf{x}}_i$ dos elipsóides são gerados de forma aleatória. As matrizes $\tilde{Q}_j$ são construídas com autovalores aleatórios.

Para criar as tabelas e gráficos de resultados, os problemas PCVQC foram gerados para dimensões $n = \{5, 15, 25\}$ e para $m = 2$. Novamente foram utilizadas duas configurações do parâmetro tamanho da busca histórica, ou seja, $3n$ e $kmax$. A partir dos resultados obtidos dos algoritmos propostos $MPESC$ e $HISPE$ foram geradas as tabelas 7.25 a 7.27.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total	Erro $\lvert\frac{\nabla f_1(x^*)}{\lvert\nabla f_1(x^*)\rvert} + \frac{\nabla f_2(x^*)}{\lvert\nabla f_2(x^*)\rvert}\rvert$
$MPESC$	452	0.9042	451	0.16	1.09e-003
$HISPE_{15}$	175	0.7454	174	0.85	8.72e-004
$HISPE_{kmax}$	58	0.4067	57	3.10	9.23e-004

Tabela 7.25: Resultados comparativos dos algoritmos $MPESC$ e $HISPE$ para um problema PCVQI de dimensão 5.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total	Erro $\lvert\frac{\nabla f_1(x^*)}{\lvert\nabla f_1(x^*)\rvert} + \frac{\nabla f_2(x^*)}{\lvert\nabla f_2(x^*)\rvert}\rvert$
$MPESC$	3132	0.9672	3131	3.92	7.64e-004
$HISPE_{45}$	830	0.8659	829	16.04	5.58e-004
$HISPE_{kmax}$	177	0.5094	176	98.61	4.93e-004

Tabela 7.26: Resultados comparativos dos algoritmos $MPESC$ e $HISPE$ para um problema PCVQI de dimensão 15.

Algorimos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total	Erro $\lvert\frac{\nabla f_1(x^*)}{\lvert\nabla f_1(x^*)\rvert} + \frac{\nabla f_2(x^*)}{\lvert\nabla f_2(x^*)\rvert}\rvert$
$MPESC$	8178	0.9802	8177	30.05	4.59e-004
$HISPE_{75}$	1650	0.8976	1649	75.14	4.75e-004
$HISPE_{kmax}$	335	0.5908	334	644.49	4.54e-004

Tabela 7.27: Resultados comparativos dos algoritmos $MPESC$ e $HISPE$ para um problema PCVQI de dimensão 25.

Os resultados apresentados nas tabelas 7.25 a 7.27 apresentam um comportamento para os parâmetros número de iterações, taxa de convergência e número de acessos bem similar aos resultados já discutidos para o problema PCVQC. Todavia, para o problema PCVQI as configurações do algoritmo $HISPE$ para $\tau = 3n$ e $\tau = kmax$ apresentam maior tempo de convergência total ao se comparar os resultados com os obtidos pelo algoritmo $MPESC$. Uma vez que existe uma menor quantidade de informações já calculadas disponíveis para a redução do poliedro de busca, dado o problema ser irrestrito, este melhor desempenho do algoritmo $MPESC$ não constitui um fato novo. Para permitir a visualização gráfica destas mesmas observações, são exibidas as figuras 7.17 e 7.18.

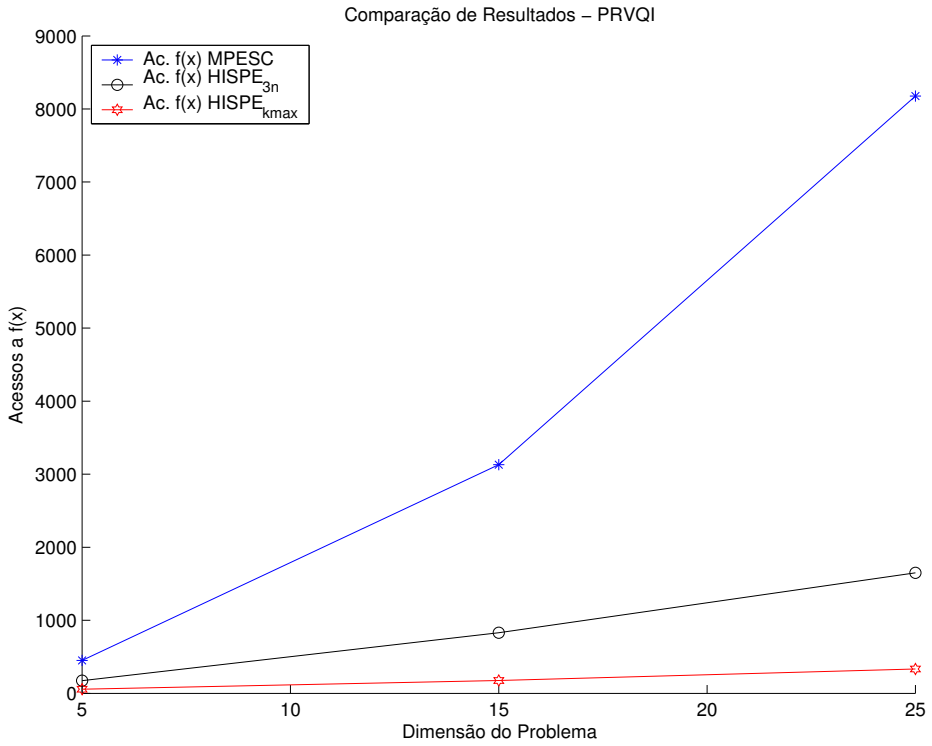


Figura 7.17: Resultados para o número de acessos à $f(\mathbf{x})$ dos algoritmos $MPESC$ e $HISPE$ para problemas PVEQI.

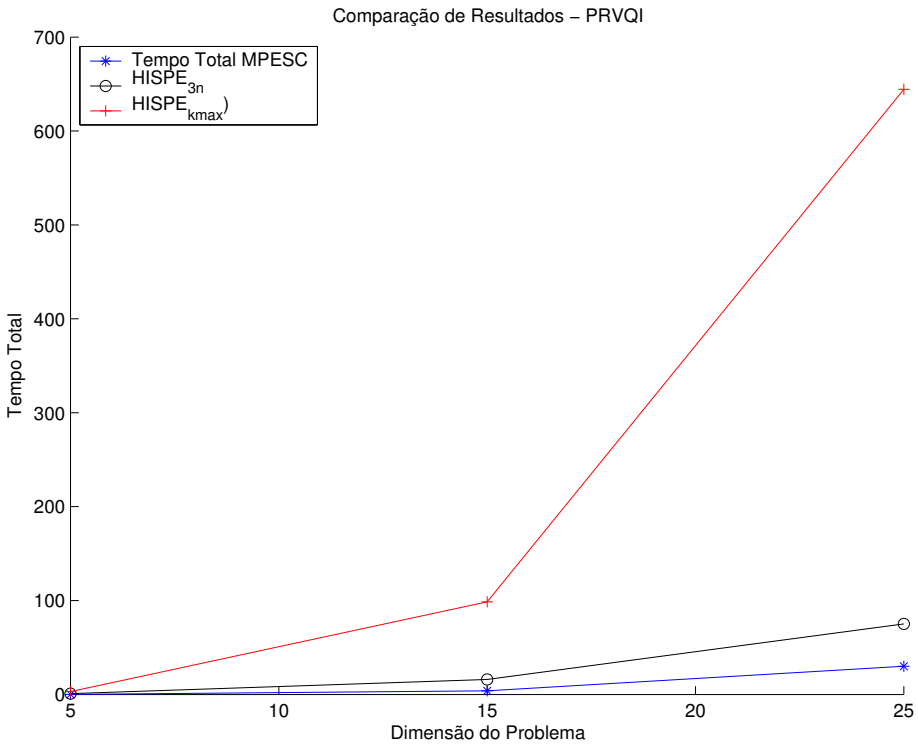


Figura 7.18: Resultados para o tempo total para convergência dos algoritmos $MPESC$ e $HISPE$ para problemas PVEQI.

Para a análise dos resultados exibidos pelo parâmetro $|\frac{\nabla f_1(x^*)}{|\nabla f_1(x^*)|} + \frac{\nabla f_2(x^*)}{|\nabla f_2(x^*)|}|$, ressalta-se que os vetores $\nabla f_1(\mathbf{x})$ e $\nabla f_2(\mathbf{x})$ devem ser paralelos para os pontos do conjunto *Pareto-Ótimo*. Este fato é decorrente do problema ser irrestrito, quadrático e composto por dois objetivos. Por conseguinte, para todos os testes realizados, verifica-se que os métodos $MPESC$ e $HISPE$ apresentaram resultados com boa precisão, já os valores exibidos nas tabelas 7.25 a 7.27 são quase sempre da ordem de $5\text{x}10^{-4}$.

7.2.2 Controlador com Norma H_2/H_∞

Durante o processo de projeto de um sistema de controle a especificação simultânea das características de robustez e desempenho pode ser acompanhada pelo critério das normas H_∞ e H_2. Por conseguinte, este problema pertence à classe dos problemas vetoriais. A realização de um espaço de estados para uma planta $P(s)$ é representada por (7.10). O espaço de estados para o sistema de controle em malha fechada para $\mathbf{u} = K\mathbf{v}$ é representado por (7.11).

$$\begin{cases} \dot{\mathbf{v}} & = A\mathbf{v} + B_1\mathbf{w} + B_2\mathbf{u} \\ z_\infty & = C_1\mathbf{v} + D_{11}\mathbf{w} + D_{12}\mathbf{u} \\ z_2 & = C_2\mathbf{v} + D_{22}\mathbf{u} \end{cases} \tag{7.10}$$

onde $\mathbf{w} \in \mathbb{R}^{\frac{n}{2}}$ é o vetor das variáveis de entrada externa, $\mathbf{u} \in \mathbb{R}^{\frac{n}{2}}$ é o vetor de variáveis de entrada por realimentação , $\mathbf{v} \in \mathbb{R}^{\frac{n}{2}}$ é o vetor de variáveis de estado, z_∞ é a variável de saída para o objetivo H_∞e z_2 é a variável de saída para o objetivo H_2. As matrizes A, B_1, B_2, C_1, C_2, D_{11}, D_{12} e D_{12} dizem respeito à planta $P(s)$.

$$\begin{cases} \dot{\mathbf{v}} & = (A + B_2K)\mathbf{v} + B_1\mathbf{w} \\ z_\infty & = (C_1 + D_{12}K)\mathbf{v} + D_{11}\mathbf{w} \\ z_2 & = (C_2 + D_{22}K)\mathbf{v} \end{cases} \tag{7.11}$$

O problema com variáveis contínuas, vetorial e não-diferenciável proposto, denominado por problema H_2/H_∞, é representado por (7.12).

$$min\ f(\mathbf{x}) = \{\ z_2 \quad z_\infty\ \} \mid \mathbf{x} = [\ \mathbf{v}^T \quad \mathbf{u}^T\ ]^T \tag{7.12}$$

onde $\mathbf{x} \in \mathbb{R}^n$, $A \in \mathbb{R}^{\frac{n}{2}\times\frac{n}{2}}$, $B_1 \in \mathbb{R}^{\frac{n}{2}\times\frac{n}{2}}$, $B_2K \in \mathbb{R}^{\frac{n}{2}\times\frac{n}{2}}$, $C_1 \in \mathbb{R}^{1\times\frac{n}{2}}$, $C_2 \in \mathbb{R}^{1\times\frac{n}{2}}$, $D_{11} \in \mathbb{R}^{1\times\frac{n}{2}}$, $D_{12}K \in \mathbb{R}^{1\times\frac{n}{2}}$ e $D_{22}K \in \mathbb{R}^{1\times\frac{n}{2}}$. Todos os valores das matrizes são gerados aleatoriamente, mas de modo a garantir um sistema em malha fechada estável.

Para a solução do problema de controle com ganho H_2/H_∞ definido por (7.10) e (7.12), duas abordagens distintas são utilizadas.

A primeira abordagem utiliza as bem conhecidas funções do módulo de controle do *MatLab*® baseadas em LMI. Nesta abordagem, primeiramente os mínimos escalares dos problemas individuais H_2 e H_∞ são encontrados. Em seguida, dado que a abordagem por LMI permite que um parâmetro γ seja utilizado como uma restrição do problema H_∞, uma seqüência de 20 valores crescentes de γ é utilizada para se obter um conjunto de soluções por esta abordagem.

A segunda abordagem utilizou o proposto algoritmo $HISPE$ para solucionar o problema H_2/H_∞. Um conjunto de 20 elipsóides iniciais E_0 é utilizado para gerar diferentes soluções para esta abordagem. Cada elipsóide E_0 é definido como uma hiperesfera que contém um hipercubo de lado 10^6 e canto inferior esquerdo $\{0, \ldots, 0\} \in \mathbb{R}^n$. Os centros dos elipsóides iniciais E_0 foram distribuídos aleatoriamente sobre a linha que conecta as soluções dos problemas individuais H_2 e H_∞.

Neste caso em particular, os testes foram executados em um computador com processador *AMD Athlon*^{TM} 64.

A figura 7.19 mostras os resultados para um problema de dimensão $n = 8$ e solucionado

pelas abordagens LMI e $HISPE$. Nesta figura, uma marca representada por um ponto foi desenhada para definir as soluções não-dominadas. Como pode ser verificado, não existe sequer uma solução obtida pela abordagem $HISPE$ que foi dominado por uma solução obtida pela abordagem LMI. Devido ao fato do módulo de controle do $MatLab^{®}$ baseado em LMI ser compilado, não é possível a comparação do desempenho dos métodos com respeito ao parâmetro tempo total para convergência.

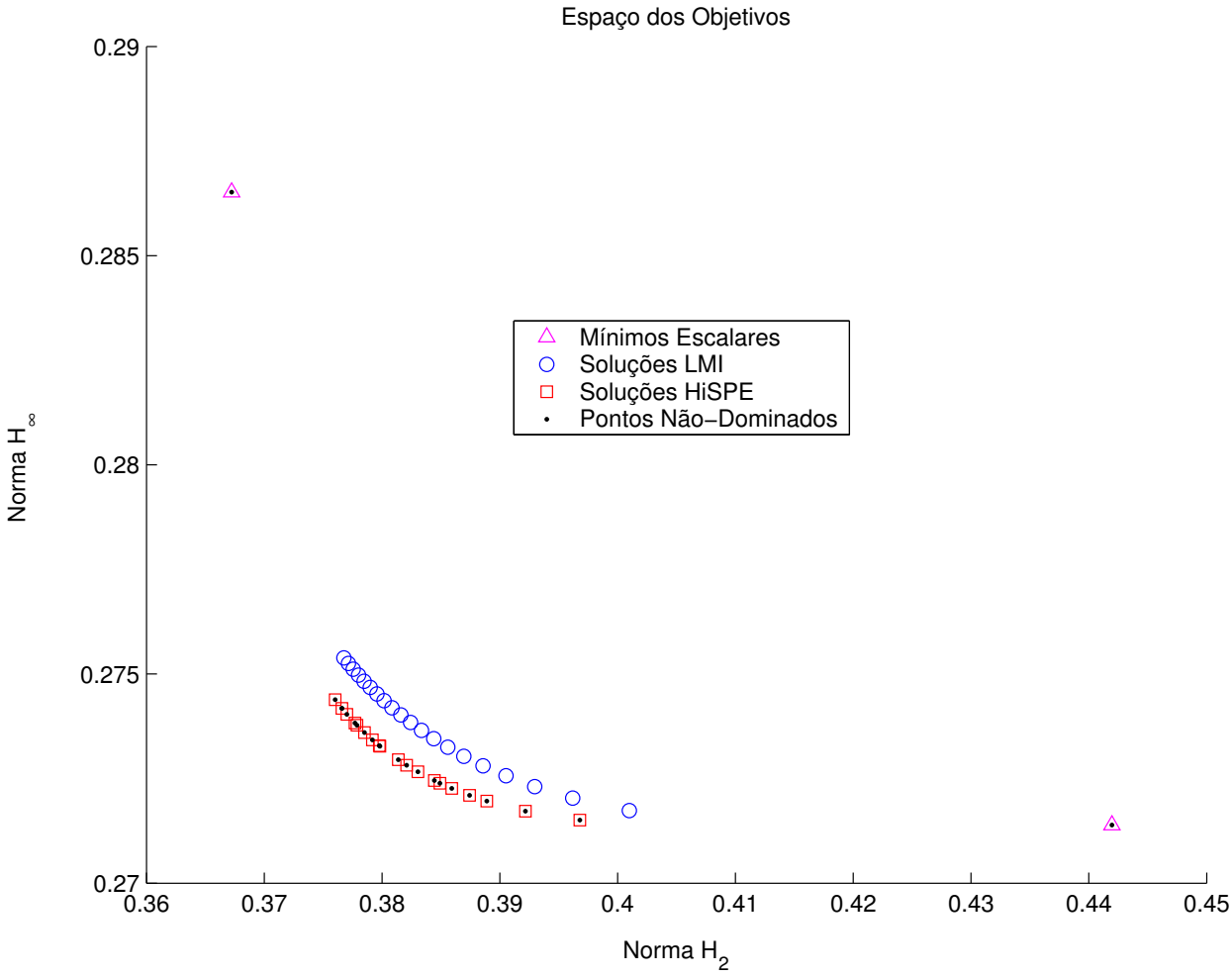


Figura 7.19: conjunto *Pareto-Ótimo* das soluções obtidas pelos algoritmos LMI e $HISPE$ para o problema H_2/H_∞.

Uma vez verificado que a abordagem $HISPE$ apresentou bons resultados para o problema H_2/H_∞, um conjunto de 10 problemas H_2/H_∞ de dimensão $n = 8$ foi utilizado para avaliar o desempenho dos algoritmos propostos. Novamente foram utilizadas duas configurações do parâmetro tamanho da busca histórica, ou seja, $3n$ e $kmax$. A partir dos resultados obtidos dos algoritmos propostos $MPESC$ e $HISPE$ foi gerada a tabela

7.28.

Algoritmos	Iterações (k)	Tx Redução Volume	Acessos $f(\mathbf{x})$	Tempo Total	% Soluções Não-Dominadas
$MPESC$	1816	0.9393	1815	69.80	100.00
$HISPE_{24}$	181	0.5739	178	16.00	100.00
$HISPE_{\infty}$	156	0.3603	155	27.80	100.00

Tabela 7.28: Resultados comparativos dos algoritmos $MPESC$ e $HISPE$ para um problema H_2/H_∞ de dimensão 8.

Os resultados apresentados na tabela 7.28 apresentam o mesmo comportamento já descrito nos testes para o problema PCVQC. Mais uma vez a configuração do algoritmo $HISPE$ para $\tau = kmax$ apresentou os melhores resultados para os parâmetros número de iterações, taxa de convergência e número de acessos. Já para o parâmetro tempo total para convergência, o menor valor foi obtido pela configuração do algoritmo $HISPE$ para $\tau = 3n$, o que reafirma a heurística proposta para o valor do parâmetro τ. Por fim, o parâmetro percentual de soluções não-dominadas permite concluir que, em todos os testes realizados, os algoritmos propostos encontraram 100% de soluções não-dominadas.

7.2.3 Problema de Factibilidade

Para avaliar a capacidade do método $HISPE$ de identificar a infactibilidade de um problema, foi definido um problema vetorial quadrático com restrições quadráticas que se tornaria infactível, ou não, dependendo de um parâmetro gerado aleatoriamente. Este problema vetorial de factibilidade com variáveis contínuas é denominado de PCVF e definido por (eq.PCVF).

$$
\begin{aligned}
min\ f(\mathbf{x}) &= [\ f_1(\mathbf{x})\ \ldots\ f_m(\mathbf{x})\ ]^T; \qquad (7.13)\\
s.a.\ \Omega &= \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{R}^n\}\\
f_j(\mathbf{x}) &= (\mathbf{x} - \tilde{\mathbf{x}}_j)^T (\tilde{Q}_j)^{-1} (\mathbf{x} - \tilde{\mathbf{x}}_j) \mid \forall\ j\\
g(\mathbf{x}) &= \begin{cases} g_i(\mathbf{x}) = (\mathbf{x} - \breve{\mathbf{x}}_i)^T (\breve{Q}_i)^{-1} (\mathbf{x} - \breve{\mathbf{x}}_i) \mid i = 1, \ldots, n+1 \\ g_{i+n+1}(\mathbf{x}) = -\mathbf{x}(i) \mid i = 1, \ldots, n \\ g_{i+2n+1}(\mathbf{x}) = \mathbf{x}(i) - lado^{\mathbf{C}} \mid i = 1, \ldots, n \end{cases}
\end{aligned}
$$

onde $\tilde{\mathbf{x}}_j$ e $\breve{\mathbf{x}}_i \in \mathbf{C}$, $\tilde{Q}_j \in \mathbb{R}^{n \times n}$ são matrizes simétricas definidas positivas e $\breve{Q}_i \in \mathbb{R}^{n \times n}$ são matrizes diagonais. Os centros $\tilde{\mathbf{x}}_j$ dos elipsóides são gerados de forma aleatória. As

matrizes $\tilde{Q}_j$ e $\breve{Q}_i$ são construídas com autovalores aleatórios. Os centros $\breve{\mathbf{x}}_i$ dos elipsóides são calculados em função de um parâmetro aleatório ρ de modo que:

$$\begin{cases} \Omega = \emptyset & \text{se } \rho < 0; \\ \Omega \neq \emptyset & \text{se } \rho \geq 0; \end{cases} \tag{7.14}$$

Para estes problemas de factibilidade, os elipsóides iniciais E_0 foram definidos de modo a garantir que $E_0 \supset \Omega$. Para os problemas de factibilidade o algoritmo $HISPE$ foi sempre configurado com o parâmetro $\tau = 3n$. Para criar a tabela 7.29, os problemas PCVF foram gerados para dimensões no intervalo $n = [5, 10]$. Cada um dos dez problemas testados para cada dimensão é exibido em uma das colunas P_1 a P_{10}. Para cada problema testado, quando o algoritmo $HISPE$ encontrou um ponto pertencente ao conjunto Λ, definido por (4.3) para a nova condição de infactibilidade proposta, a tabela 7.29 exibe o resultado $\Lambda \neq \emptyset$. Para os demais resultados, a tabela 7.29 exibe o resultado $\Omega \neq \emptyset$. Como pode ser claramente observado na tabela 7.29, tendo por referência os valores gerados para o parâmetro ρ e a definição de Ω por (7.14), em todos os problemas testados o algoritmo $HISPE$ determinou corretamente se o problema era factível ou não. Este fato permite afirmar que a nova condição necessária e suficiente de infactibilidade se mostrou adequada para definir a infactibilidade de um problema, para o conjunto dos problemas testados.

Testes	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
n = 5										
ρ	< 0	≥ 0	≥ 0	< 0	≥ 0	< 0	< 0	≥ 0	≥ 0	< 0
$HISPE$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$
n = 6										
ρ	≥ 0	≥ 0	< 0	< 0	≥ 0	≥ 0	< 0	< 0	< 0	< 0
$HISPE$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Lambda \neq \emptyset$	$\Lambda \neq \emptyset$	$\Lambda \neq \emptyset$
n = 7										
ρ	≥ 0	≥ 0	< 0	< 0	≥ 0	< 0	≥ 0	< 0	≥ 0	≥ 0
$HISPE$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$
n = 8										
ρ	≥ 0	≥ 0	≥ 0	≥ 0	< 0	≥ 0	< 0	≥ 0	≥ 0	≥ 0
$HISPE$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$
n = 9										
ρ	< 0	< 0	< 0	≥ 0	< 0	< 0	< 0	≥ 0	≥ 0	< 0
$HISPE$	$\Lambda \neq \emptyset$	$\Lambda \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Lambda \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$
n =10										
ρ	≥ 0	≥ 0	< 0	≥ 0	≥ 0	≥ 0	≥ 0	< 0	≥ 0	< 0
$HISPE$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$	$\Omega \neq \emptyset$	$\Lambda \neq \emptyset$

Tabela 7.29: Resultados dos problemas PCVF paras as dimensões n=[5:10] obtidos pelo algoritmo $HISPE$.

7.3 Conclusões do Capítulo

Neste Capítulo foram apresentados os resultados obtidos mediante a aplicação dos algoritmos que implementam a família dos métodos *Poliedro-Elipsoidais* e dos algoritmos utilizados como referência para a avaliação do desempenho associados à solução dos problemas QCND com variáveis exclusivamente contínuas.

Inicialmente, foi definida a classe de problemas PCEQC para a análise dos resultados dos algoritmos propostos $MPESC$ e $HISPE$. Os resultados exibidos na seção 7.1.1 demonstraram o bom desempenho dos algoritmos propostos, com destaque à configuração do algoritmo $HISPE$ com o parâmetro $\tau = 3n$. Para esta configuração do algoritmo $HISPE$, os resultados indicaram ainda uma tendência de melhora destes resultados, em comparação aos resultados do algoritmo *Shor*, à medida que a dimensão dos problemas aumenta. Em seqüência, a seção 7.1.2 definiu um conjunto de problemas PCECN para testes e confirmou os bons resultados apresentados para os algoritmos propostos. Já a seção 7.1.3 apresentou os resultados obtidos para os problemas PCEQQ e PCEQL. Para estes casos, embora a configuração do algoritmo $HISPE$ com o parâmetro $\tau = 3n$ sempre apresentasse resultados superiores aos resultados dos algoritmos *Shor* e Kim, os melhores resultados para o parâmetro tempo total para convergência foram obtidos com o algoritmo $MPESC$. Este fato é atribuído à redução do EAIP para os problemas de teste. Para encerrar os testes com problemas escalares, a seção 7.1.4 apresentou os resultados dos algoritmos propostos para dois problemas de Engenharia Química.

Em continuidade aos testes, a classe de problemas PCVQC foi definida na seção 7.2.1. Nesta mesma seção, duas métricas distintas foram apresentadas para testar a capacidade dos métodos *Poliedro-Elipsoidais* em encontrar pontos do conjunto *Pareto-Ótimo*. A primeira avaliou a dominância entre as soluções encontradas pelos métodos propostos. A segunda mediu a norma do valor do vetor $|\frac{\nabla f_1(x^*)}{|\nabla f_1(x^*)|} + \frac{\nabla f_2(x^*)}{|\nabla f_2(x^*)|}|$ para as soluções encontradas pelos métodos propostos, tendo o valor zero como referência. Para os problemas testados, os melhores resultados se alternaram entre o algoritmo $MPESC$ e a configuração do algoritmo $HISPE$ com o parâmetro $\tau = 3n$, em função do EAIP dos problemas testados. Para problemas com maior EAIP, destacou-se a configuração do algoritmo $HISPE$ com o parâmetro $\tau = 3n$. Para os problemas com menor EAPI, destacou-se o algoritmo $MPESC$. Em seguida, a seção 7.2.2 exibiu os resultados para o problema de Engenharia de Controle, referente ao projeto de um controlador considerando-se simultaneamente as normas H_2 e H_∞. Esta seção demonstrou as vantagens de se utilizar uma abordagem vetorial com a utilização do algoritmo $HISPE$ em relação à abordagem com utilização de LMI. Por fim a seção 7.2.3 demonstrou que a nova condição necessária e suficiente de infactibilidade, proposta no capítulo 4, pode ser utilizada como critério de parada de um

algoritmo por infactibilidade do problema testado.

No capítulo seguinte serão apresentados os fundamentos para o entendimento dos métodos de otimização que solucionam problemas onde as variáveis associadas não são exclusivamente variáveis contínuas.

Parte II

Otimização com Variáveis Discretas e Mistas Discretas e Contínuas

Capítulo 8

Métodos Baseados em Elipsóide Aplicados à Programação Discreta e à Programação Mista

Neste capítulo são apresentados os fundamentos para o entendimento dos métodos de otimização que solucionam problemas onde as variáveis associadas não são exclusivamente variáveis contínuas.

Em seguida são discutidas as limitações da aplicação direta dos métodos elipsoidais para a solução de problemas discretos e mistos contínuos e discretos, bem como apresentadas as definições dos limites de convergência superior e inferior. Estes limites serão utilizados para a implementação de um método *Elipsoidal* capaz de solucionar problemas discretos e mistos discretos e contínuos.

Maiores detalhes sobre os temas tratados nas seções apresentadas neste capítulo podem ser conseguidos nas referências (Salkin & Mathur n.d., Floudas n.d., Papadimitriou & Steiglitz n.d.).

8.1 Introdução à Programação Discreta e Mista Contínua e Discreta

Um grande número de problemas de otimização possui a restrição de que parte, ou todas, as variáveis envolvidas devem assumir valores discretos. Problemas associados ao agendamento de equipes ou de rotas, processos químicos de síntese, alocação e estocagem de materiais, agendamento e planejamento de processos em batelada, desenvolvimento de produtos bioquímicos e determinação de topologia de transportes em redes, sistemas

de controle, seleção do portifólio de investimentos, processo de evacuação de pessoas em emergência e propagação da frente de chama de incêndios descrevem a vastidão da presença das variáveis discretas nas áreas da ciência (Wolsey n.d., Floudas n.d., Nemhauser & Wolsey n.d., Parker. & Rardin 1988, Papadimitriou & Steiglitz n.d., Parker. & Rardin 1988, Li & Sun n.d., Smith n.d.).

Apesar das últimas duas décadas terem sido memoráveis para o desenvolvimento da capacidade de encontrar, ou de aproximar, soluções ótimas para diversos dos problemas práticos supracitados, ainda hoje muitas aplicações práticas consistem apenas em formalizar e encontrar uma solução factível. Este procedimento acaba sempre por conduzir a perdas desnecessárias, as quais poderiam ser reduzidas mediante a melhor compreensão do porque alguns problemas são mais difíceis de resolver do que outros, de como algumas das possíveis formulações de um problema podem ser melhores que as outras e de como a utilização de diferentes algoritmos podem produzir soluções mais efetivas (Wolsey n.d.).

A estrutura geral dos problemas mono-objetivos nos quais existe a presença de variáveis discretas é descrito por (8.1).

$$min\ f(\mathbf{x}) \tag{8.1}$$

$$s.a.\ \mathbf{x} \in \Omega$$

$$\Omega \triangleq \{\mathbf{x}(1:z) \in \mathbb{I}_+^z\ e\ \mathbf{x}(z+1:z+d) \in \mathbb{R}^d \mid g_i(\mathbf{x}) \leq 0\ \forall\, i = 1, \ldots, p\}$$

onde, $z + d = n$, $f(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}$ e $g(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}^p$.

Define-se como o problema relaxado para (8.1), um sub-problema no qual permite-se que parte do conjunto das variáveis discretas, ou todo este conjunto, assuma valores contínuos. A solução de um sub-problema, obtida por relaxação, é indicada por $\mathbf{x}^*$.

A partir de (8.1) definem-se quatro grandes subclasses de problemas. A primeira e mais genérica subclasse constitui a classe onde existem simultaneamente variáveis contínuas e discretas nos problemas e é denominada de Programação Mista Discreta e Contínua ou MIP. Se considerarmos que o conjunto de variáveis contínuas é vazio, ou seja $d = 0$, teremos os problemas associados à programação discreta ou IP. No caso especial em que todas as variáveis discretas são binárias, $\mathbf{x} \in \{0, 1\}^z$, teremos os problemas de programação binária BIP ou também chamada de programação zero-um. Por razões históricas do desenvolvimento da área de pesquisa operacional, os nomes IP e MIP são comumente associados ao caso particular onde todas as funções envolvidas são lineares, ficando os problemas discretos não lineares conhecidos por NIP e os problemas mistos não lineares conhecidos por MINLP (Wolsey n.d., Floudas n.d.).

Deve-se ainda observar que todo problema MIP ou IP pode ser transcrito para um problema equivalente BIP. O Teorema 6 formaliza esta afirmação:

Teorema 6 - *Equivalente BIP (Salkin & Mathur n.d.) : Suponha em um problema MIP (ou IP) que cada variável* $\mathbf{x}(j) \leq u_j$*, sendo* u_j *correspondente a um discreto positivo e* $j \leq z$*. Então o programa BIP equivalente será obtido pela substituição de cada* $\mathbf{x}(j)$ *por: (a)* $\sum_{k=1}^{u_j} t_{kj}$*, onde* t_{kj} *representa uma variável binária, e pela omissão da restrição* $\mathbf{x}(j) \leq u_j$ *dado que* $\sum_{k=1}^{u_j} t_{kj}$ *poderá no máximo assumir o valor* u_j*, ou (b)* $\sum_{k=0}^{l_j} 2^k t_{kj}$*, onde* t_{kj} *representa uma variável binária e* l_j *é o menor inteiro tal que* $\sum_{k=0}^{l_j} 2^k = (2^{l_j+1} - 1)$*.*

A estrutura do problema descrito por (8.1) pode induzir a idéia da solução deste através de métodos de otimização para variáveis contínuas e do posterior arredondamento das variáveis que precisam ser discretas para a obtenção da solução mista ou discreta final. Todavia, deve ser observado que normalmente este processo demonstra ser insuficiente para a obtenção da solução ótima do problema. A figura 8.1 exibe graficamente um problema onde a solução discreta $\mathbf{x}^{z*}$, encontra-se visivelmente afastada da solução contínua $\mathbf{x}^*$ obtida pelo problema relaxado. Sendo assim qualquer solução na vizinhança de $\mathbf{x}^*$ não corresponderá a uma boa aproximação da solução $\mathbf{x}^{z*}$. Por vizinhança, entende-se neste contexto, o hipercubo de aresta unitária e de vértices puramente compostos por valores discretos que contém o ponto $\mathbf{x}^*$. Em problemas com variáveis binárias este afastamento é, normalmente, ainda mais significativo para a obtenção da solução por arredondamento. Todavia, para os problemas onde o intervalo de variação das variáveis discretas seja consideravelmente longo, a ponto de que a fração correspondente ao valor arredondado produza um valor insignificante no valor da função-objetivo, o processo de arredondamento da solução contínua irá produzir soluções discretas com aproximação aceitável.

Uma vez que a factibilidade de um problema MIP ou IP depende da existência de uma solução $\mathbf{x}^{z*}$ internamente à região factível Ω, a figura 8.1 também demonstra que para os problemas onde as funções envolvidas são convexas, ou quasi-convexas, obrigatoriamente teremos a solução do problema MINP ou NIP em uma curva de nível da função-objetivo de valor superior[1] ao valor correspondente à solução $\mathbf{x}^*$. O teorema 7 e seu corolário formalizam esta afirmação:

Teorema 7 - *Limite Inferior ("Lower Bound") (Salkin & Mathur n.d.) : O mínimo valor da função-objetivo de um problema MIP ou IP solucionado como um problema de variáveis puramente contínuas, ou seja* $\mathbf{x}(1 : z) \in \mathbb{R}^z$*, constitui o limite inferior de qualquer solução mista, ou discreta, factível.*

[1]Para problemas de maximização a solução ocorrerá obrigatoriamente em uma curva de nível de valor inferior ao valor correspondente à solução $\mathbf{x}^*$.

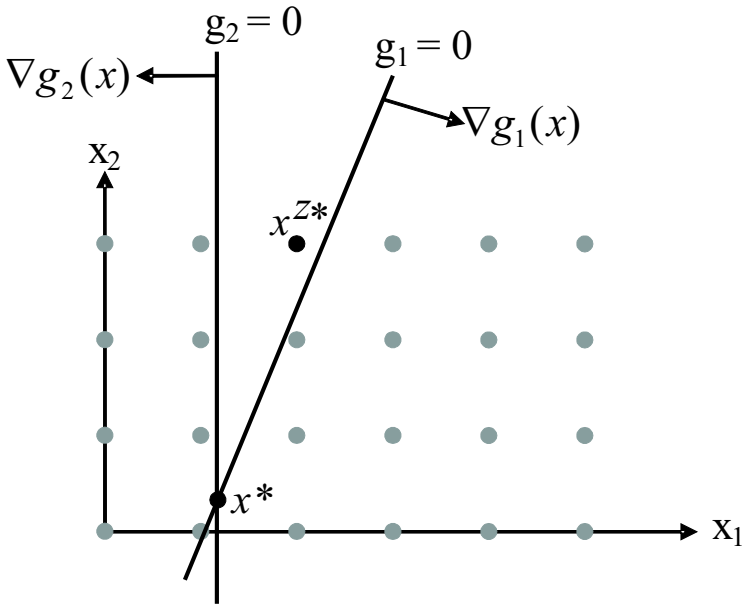


Figura 8.1: Exemplo comparativo da solução discreta de um problema $\mathbf{x}^{z*}$ e da solução obtida pelo arredondamento da solução contínua $\mathbf{x}^*$.

Corolário 6 - *Factibilidade do MIP ("Feasibility of MIP") (Salkin & Mathur n.d.) : Se um problema MIP ou IP solucionado como um problema de variáveis puramente contínuas é infactível, então também será infactível o problema MIP ou IP.*

Embora a enumeração por valores 0 ou 1 dos elementos que compõem as variáveis $\mathbf{x}(j) \,|\, j \leq z$ possa conduzir a obtenção do resultado ótimo de um problema MIP ou IP, existe uma dificuldade intrínseca a ser considerada. Esta dificuldade é traduzida no grande esforço computacional para a solução do problema em virtude da característica combinatória associada às variáveis discretas. Desta forma em muitos casos a abordagem por força bruta para a enumeração das soluções se torna proibitiva. Como exemplo, supondo a existência de 100 variáveis binárias para a solução de um problema, isto implicará na solução das 2^{100} possibilidades de solução por enumeração completa.

A despeito da complexidade da análise do grande número de soluções parciais, decorrentes da natureza combinatória dos problemas MIP ou IP, diversas classes de algoritmos têm sido propostas com êxito para a solução destes problemas. Para a solução de problemas MIP ou IP (Nemhauser & Wolsey n.d.) se destacam:

i) Métodos Branch and Bound: nestes métodos uma árvore binária é utilizada para a representação das variáveis discretas por sua combinação 0-1, sendo então a região factível particionada subsequentemente em subdomínios com a definição de limites válidos de limites superiores e inferiores da solução em diferentes níveis desta árvore

binária[2].

ii) Métodos de Plano de Corte: nestes métodos a região factível não é dividida em subdomínios, mas sim novas restrições, usualmente denominadas de cortes, são adicionadas ao problema inicial de modo a reduzir a região factível até a obtenção da solução discreta ótima[3].

iii) Métodos de Decomposição: nestes métodos a estrutura matemática dos modelos é explorada através do particionamento de variáveis, dualidade e métodos de relaxação.

iv) Métodos Baseados em Lógica: nestes métodos restrições disjuntivas ou técnicas de inferência simbólica são utilizadas pela sua expressão através de variáveis binárias.

Tal como ocorrido para os problemas lineares, significativos progressos também foram obtidos para a solução de problemas MINLP ou NIP do ponto de vista teórico e computacional. Como resultado, um grande número de problemas, que permeiam por diferentes disciplinas da ciência, têm sido solucionados pelos diversos algoritmos propostos. Uma coleção representativa destes algoritmos propostos é apresentada a seguir:

i) Decomposição Generalizada de Benders - GDE: neste algoritmo duas seqüências de atualização de limites, um superior obtido pela solução do problema que considera as variáveis $\mathbf{x}(1:z) \in \mathbb{R}^z$ e um inferior obtido pela teoria da dualidade, são criadas e convergem para um valor limite desejado em um número finito de iterações.

ii) Branch and Bound - BB: este algoritmo considera a solução da relaxação do problema MIP em variáveis contínuas e realiza a subseqüente enumeração implícita na forma de árvore onde uma combinação do conjunto de variáveis discretas é fixada em cada nó[4].

iii) Aproximação Externa - OA: este algoritmo trata de problemas com desigualdades não lineares e cria uma seqüência de limites inferiores e superiores de forma similar ao algoritmo GDE. Contudo, utiliza uma linearização das funções objetivo e restrições no ponto ótimo da solução primal para a definição dos limites superiores. Já a obtenção dos limites inferiores ocorre pela linearização das funções objetivo e restrições nos pontos da solução ótima primal generalizada.

[2] O método Branch and Bound será detalhado no capítulo 10.
[3] Os métodos de Plano de Corte serão detalhados no capítulo 9.
[4] O método Branch and Bound será detalhado no capítulo 10.

iv) Abordagem por Factibilidade - FA: este algoritmo arredonda a solução relaxada do problema não linear com a menor degradação local por sucessivas imposições de que as variáveis super-básicas se tornem variáveis não-básicas da função-objetivo reduzida.

v) Aproximação Externa com Relaxação de Igualdades OA/ER: este algoritmo estende o algoritmo OA para incluir a capacidade deste em lidar com igualdades não lineares por meio da relaxação destas em desigualdades.

vi) Aproximação Externa com Relaxação de Igualdades e Aumento de Penalidades OA/ER/AP: este algoritmo introduz uma função de penalidade aumentada nos subproblemas de limite inferior para a abordagem AO/ER.

vii) Aproximação Externa Generalizada - GOA: este algoritmo estende o algoritmo OA com a introdução de funções de penalidade exatas.

viii) Decomposição Cruzada Generalizada - GCD: este algoritmo simultaneamente utiliza as informações dos problemas primal e dual pela utilização das vantagens das decomposições Dantzig-Wolfe e Benders generalizada.

Deve-se ressaltar que, não obstante à existência de um número considerável de problemas MINP classificados como quasi-convexos (Roberts & Varberg n.d., Floudas n.d.), há apenas uns poucos algoritmos capazes de resolver estes problemas com garantia de convergência global (Westerlund & Pörn 2002). Dentre os algoritmos capazes de garantir convergência global para problemas MINP classificados como quasi-convexos, destacam-se os algoritmos baseados em *Branch-and-Bound* e *Planos de Corte*. Por conseguinte, o desenvolvimento de algoritmos com a capacidade de solucionar problemas MINP quasi-convexos e não necessariamente diferenciáveis constituiu um ponto chave da motivação do desenvolvimento desta pesquisa.

8.2 Restrição da Aplicação Direta dos Métodos Baseados em Elipsóide a Problemas Inteiros e Mistos Inteiros e Contínuos

Excluindo-se os casos em que é válida a heurística de se arredondar o valor resultante da solução contínua de um problema ($\mathbf{x}^*$) para a obtenção da solução discreta ou mista ($\mathbf{x}^{z*}$), a família dos algoritmos derivados do método *Elipsoidal* não pode ser aplicada

diretamente na solução de problemas MINP ou NIP. De forma ainda mais genérica, os métodos que utilizam exclusivamente a direção de um vetor pertencente ao conjunto subdiferencial para determinar a direção de busca ou restringir o espaço de busca não podem ser aplicados diretamente na solução de problemas MINP ou NIP. Entende-se aqui que estes métodos que não podem ser aplicados a problemas MINP ou NIP não incluem duas classes de métodos. A primeira classe é formada pelos métodos que adicionam restrições para garantir a existência de uma solução mista (ou discreta) válida no semi-espaço a ser utilizado. A segunda classe é formada pelos métodos que garantem que as soluções parciais de todas as iterações devem apresentar valores discretos para as variáveis que devem ser discretas.

A principal limitação da utilização de métodos que dependem da direção de um vetor pertencente ao conjunto subdiferencial na solução de problemas MINP e NIP é resultante da não linearidade intrínseca à obrigatoriedade de que parte, ou todas, as variáveis do problema sejam discretas. Ao se adotar a direção oposta a um vetor pertencente ao conjunto subdiferencial existe uma pressuposição implícita. Esta pressuposição é definida pela idéia de que, dada a existência de um conjunto não nulo resultante da intersecção da região factível Ω com o semi-espaço definido pela direção oposta ao vetor pertencente ao conjunto subdiferencial $H_{-}^{\bar{\nabla} f(\mathbf{x}),\mathbf{x}_k}$, é sobre este conjunto intersecção que deve ser realizada a busca de uma solução melhor que a existente no ponto atual. Todavia, para os problemas MINP ou NIP deve-se acrescentar à intersecção do semi-espaço $H_{-}^{\bar{\nabla} f(\mathbf{x}),\mathbf{x}_k}$ e da região factível Ω o subconjunto das variáveis obrigatoriamente discretas $(\mathbf{x}(1:z) \in \mathbb{I}^z)$ de forma a garantir que a direção utilizada possa conter uma solução que domine a atual.

A figura 8.2 ilustra um exemplo de um problema NIP. As equações de restrição estão indicadas na figura, sendo a região factível composta por um setor do elipsóide $(\mathbf{x}-\mathbf{x}_c)^T Q^{-1}(\mathbf{x}-\mathbf{x}_c) \leq 1$ com $\mathbb{I}^n$. O subconjunto definido pelo semi-elipsóide $E_k \cap H_{-}^{\bar{\nabla} f(\mathbf{x}_k),\mathbf{x}_k}$ não contém qualquer ponto discreto, tal como exibido na figura 8.2a. O próximo elipsóide E_{k+1} deve ser construído pelo semi-elipsóide $E_k \cap H_{+}^{\bar{\nabla} f(\mathbf{x}_k),\mathbf{x}_k}$, tal como mostrado na figura 8.2b.

8.3 Princípios Básicos da Aplicação de Métodos Baseados em Elipsóide a Problemas Inteiros e Mistos Inteiros e Contínuos

A existência de uma restrição para a utilização direta de métodos elipsoidais a problemas MINP ou NIP não deve ser entendida como um impossibilidade definitiva. Esta restrição

implicará apenas na necessidade de aumento da complexidade dos algoritmos elipsoidais de modo a que a convergência global para a solução discreta, ou mista discreta e contínua, seja assegurada.

Durante a convergência de um algoritmo baseado em *Elipsóide* para a solução de um problema MNIP ou NIP torna-se necessária a definição de limites superiores e inferiores. Estes limites são relativos ao valor das soluções existentes em qualquer iteração k do algoritmo *Elipsoidal* para os sub-problemas obtidos por relaxação. Estes limites balizarão a convergência do algoritmo na convergência para a solução discreta, ou mista discreta e contínua.

A seguir, são apresentadas as definições que formalizam o conceito dos limites, inferior e superior, associados à convergência dos métodos elipsoidais para a solução discreta, ou mista discreta e contínua.

Definição 9 *: Limite Inferior de Convergência (LIC): Para um problema definido por (8.1), considere o conjunto X composto pelas soluções ótimas factíveis de um conjunto não nulo de sub-problemas obtidos por relaxação. Considere o ponto $\mathbf{x}^{lb} = \{\mathbf{x}^{lb} \in X \mid f(\mathbf{x}^{lb}) \geq f(\mathbf{x}) \; \forall \; (\mathbf{x} \in X) \; e \; (\mathbf{x} \neq \mathbf{x}^{lb})\}$. O limite inferior de convergência é definido como o semi-espaço $H_{+}^{\bar{\nabla} f(\mathbf{x}^{lb}),\mathbf{x}^{lb}}$.*

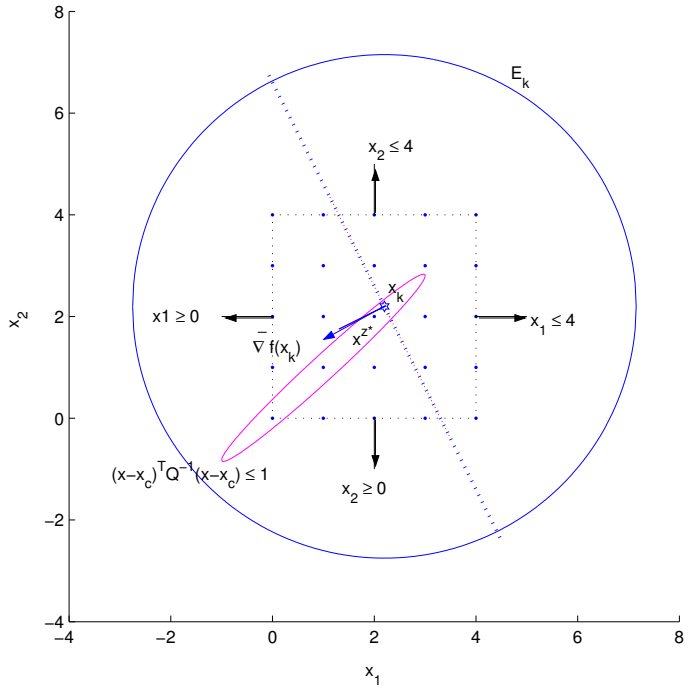
Definição 10 *: Limite Superior de Convergência (LSC): Para um problema definido por (8.1), considere o conjunto $\hat{X} = \{\mathbf{x} \in \Omega \mid \mathbf{x}(1:z) \in \mathbb{I}^z\}$. Considere o ponto $\mathbf{x}^{ub} = \{\mathbf{x}^{ub} \in \hat{X} \mid f(\mathbf{x}^{ub}) \leq f(\mathbf{x}) \; \forall \; (\mathbf{x} \in \hat{X}) \; e \; (\mathbf{x} \neq \mathbf{x}^{ub})\}$. O limite superior de convergência é definido como o semi-espaço $H_{-}^{\bar{\nabla} f(\mathbf{x}^{ub}),\mathbf{x}^{ub}}$.*

Como pode ser observado pela Definição 10, para o caso em que $\hat{X} = \Omega$, o ponto $\mathbf{x}^{ub}$ corresponderá a $\mathbf{x}^{z*}$. Para os demais casos, onde $\hat{X}$ é um subconjunto de Ω, o ponto $\mathbf{x}^{ub}$ representa a melhor solução no subconjunto $\hat{X}$. Deve-se ainda ser ressaltado que, durante a convergência de um método *Elipsoidal* para a solução discreta, ou mista discreta e contínua, a intersecção entre os semi-espaços definidos pelos limites inferior e superior de convergência e pela região factível, $(H_{+}^{\bar{\nabla} f(\mathbf{x}^{LIC}),\mathbf{x}^{LIC}} \cap H_{-}^{\bar{\nabla} f(\mathbf{x}^{LSC}),\mathbf{x}^{LSC}} \cap \Omega)$, define a região do espaço de variáveis onde pode ser encontrada a solução ótima do problema.

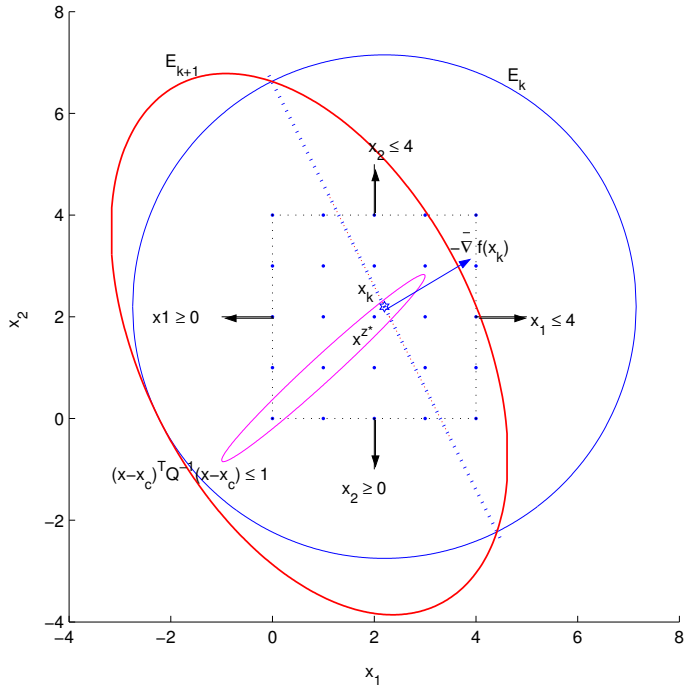
Conseqüentemente, a estratégia a ser adotada para a adaptação dos métodos elipsoidais aos problemas MINP ou NIP consiste em identificar os valores dos limites durante o processo de convergência e prover a contínua contração da região definida por $(H_{+}^{\bar{\nabla} f(\mathbf{x}^{LIC}),\mathbf{x}^{LIC}} \cap H_{-}^{\bar{\nabla} f(\mathbf{x}^{LSC}),\mathbf{x}^{LSC}} \cap \Omega)$ até que seja possível garantir que não exista nesta região de interesse de busca qualquer solução discreta, ou mista discreta e contínua, de valor inferior à definida pelo limite superior da iteração corrente.

Seguindo esta linha de raciocínio, no capítulo 9 será apresentada uma variação do método *Elipsoidal* que permitirá a obtenção da solução ótima para problemas NIP baseada na movimentação do LIC, pela inclusão de novas restrições definidas por planos de corte, e da movimentação do LSC, pela avaliação dos pontos discretos que se encontram na vizinhança do centro atual do elipsóide. Um fluxograma geral de um algoritmo *Elipsoidal* alterado para que ocorra a inclusão de novas restrições baseadas em planos de cortes é apresentado na figura 8.3. Pode-se observar que os objetos sombreados representam as alterações implementadas no algoritmo HISPE para permitir que este solucione problemas NIP.

Em seqüência, no capítulo 10 serão apresentadas três variações do método HISPE que possibilitarão encontrar a solução ótima tanto para problemas NIP quanto para problemas MINP com base na movimentação de ambos os limites LIC e LSC através da enumeração de um ponto discreto, ou misto discreto e contínuo, existente em uma determinada região de interesse de busca. Um fluxograma geral de um algoritmo *Elipsoidal* alterado para enumerar um ponto discreto, ou misto discreto e contínuo, é apresentado na figura 8.4. Novamente pode-se observar que os objetos sombreados representam as alterações implementadas no algoritmo HISPE para permitir que este solucione problemas MINP e NIP.



a) Semi-elipsóide incorreto definido por $H_{-}^{\bar{\nabla} f(\mathbf{x}_k), \mathbf{x}_k}$.



b) Correto elipsóide E_{k+1} que contém a solução ótima $\mathbf{x}^{z*}$ definido por $H_{+}^{\bar{\nabla} f(\mathbf{x}_k), \mathbf{x}_k}$.

Figura 8.2: Falha do método *Elipsoidal* quando aplicado a problemas MINP e NIP.

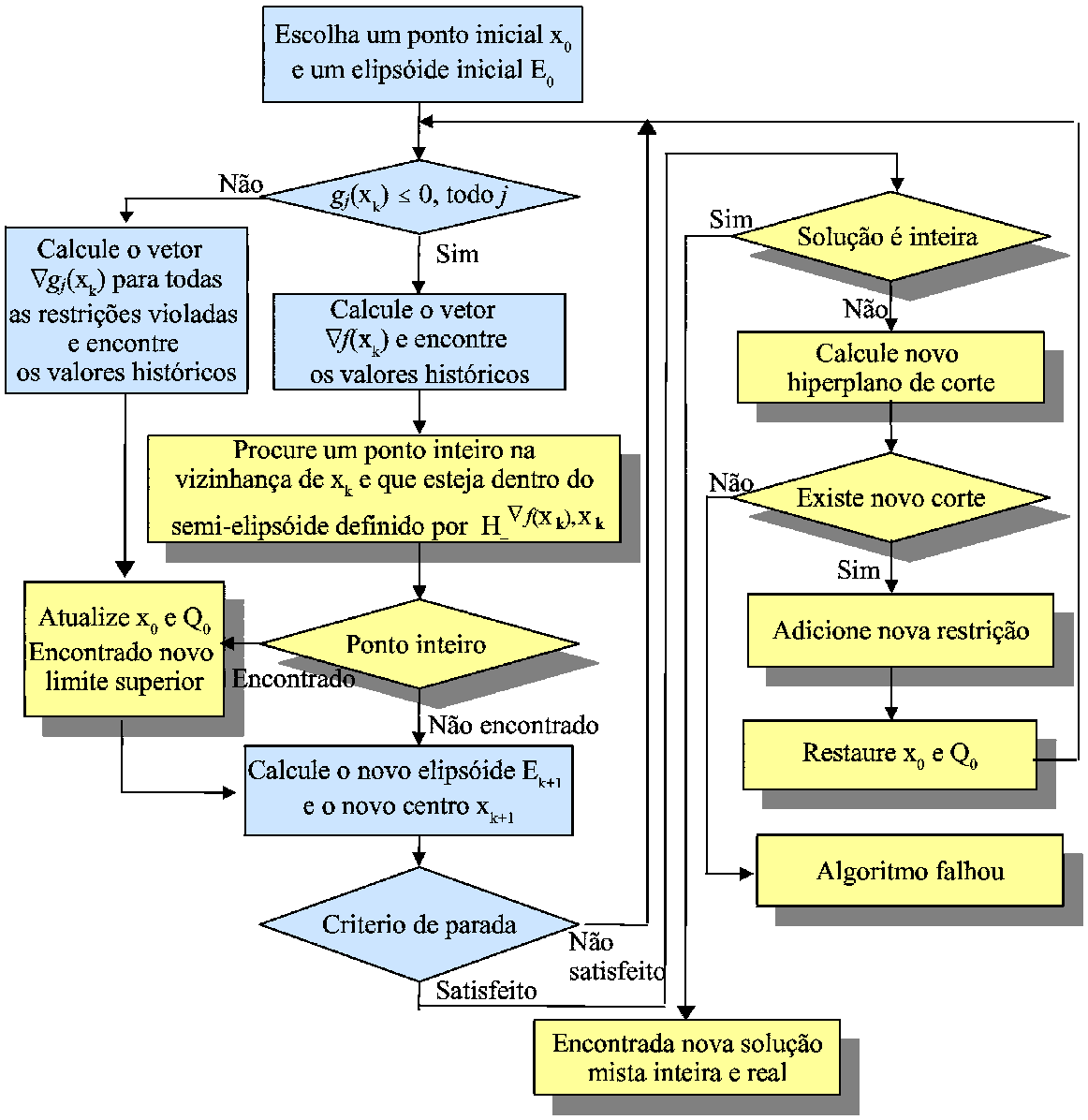


Figura 8.3: Fluxograma geral de um algoritmo *Elipsoidal* alterado para a inclusão de novas restrições baseadas em planos de cortes.

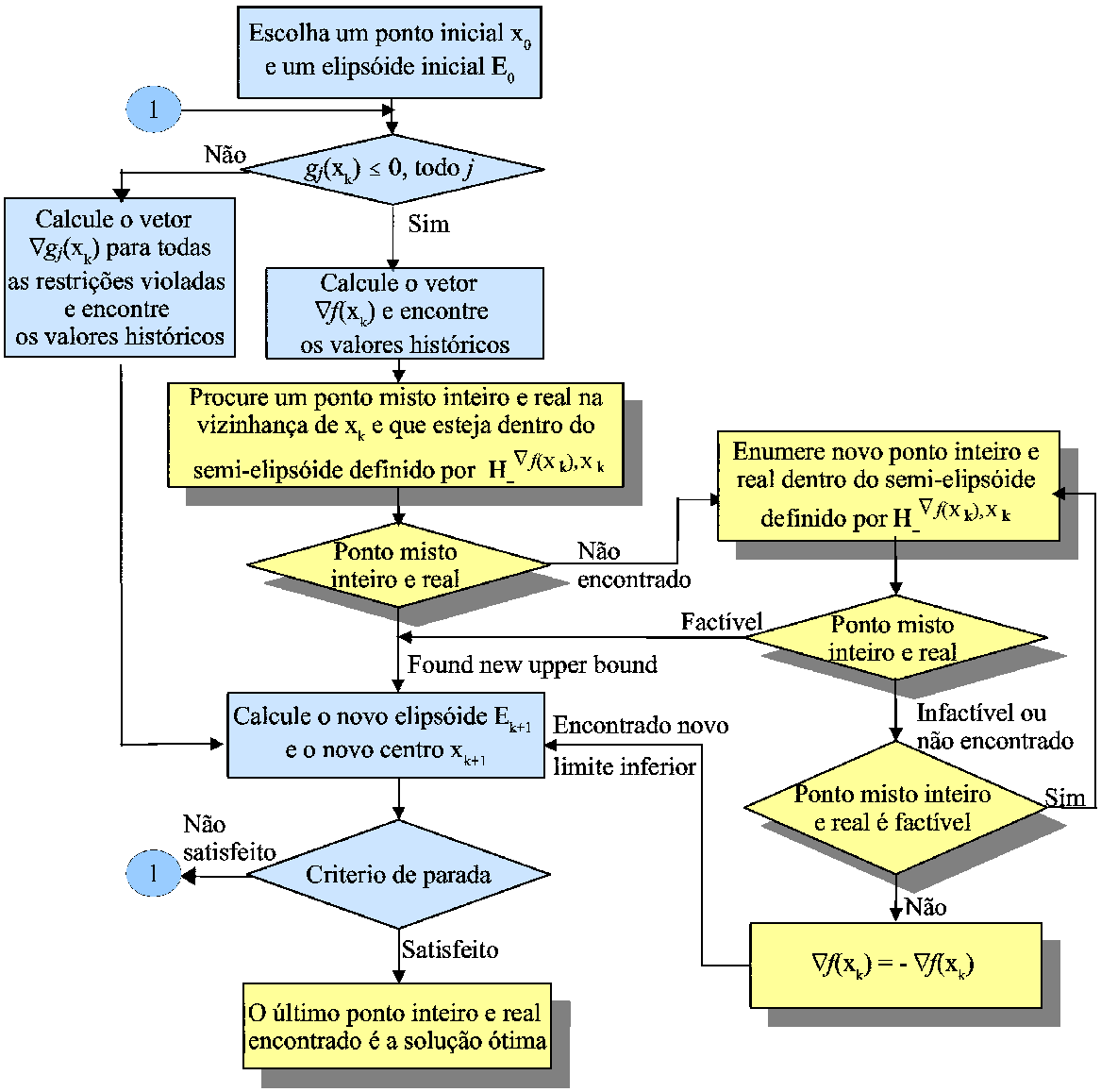


Figura 8.4: Fluxograma geral de um algoritmo *Elipsoidal* alterado para enumerar um ponto discreto, ou misto discreto e contínuo.

8.4 Conclusões do Capítulo

Neste capítulo foram apresentados os fundamentos para o entendimento dos métodos de otimização que solucionam problemas onde as variáveis associadas não são exclusivamente variáveis contínuas.

Em seqüência foram discutidas as limitações da aplicação direta dos métodos baseados em *Elipsóide* para a solução de problemas MINP e NIP, bem como apresentadas as definições dos limites de convergência superior e inferior, os quais devem ser utilizados para a implementação de um método baseado em *Elipsóide* capaz de solucionar problemas MINP e NIP.

No capítulo seguinte será apresentada a teoria para a obtenção de novas restrições que permitem a obtenção da solução discreta de um problema NIP a partir da movimentação do limite inferior de convergência pela solução contínua obtida por relaxação das variáveis discretas.

Capítulo 9

Métodos Elipsoidal e Poliedro-Elipsoidal com Hiperplanos de Corte Aplicados a Problemas Discretos

Neste capítulo é apresentada a teoria para a geração de novas restrições, na forma de hiperplanos de corte, que permitem a obtenção da solução de um problema com variáveis discretas a partir da movimentação do limite inferior de convergência correspondente à solução obtida por relaxação.

No decorrer do capítulo também é apresentada uma variação de um método baseado em *Elipsóide* capaz de resolver problemas discretos com restrições quasi-convexas não necessariamente diferenciáveis baseada em hiperplanos de corte. Este algoritmo é a primeira contribuição desta Tese para a solução de problemas NIP.

Maiores detalhes sobre os temas tratados nas seções apresentadas neste capítulo podem ser conseguidos nas referências (Salkin & Mathur n.d., Floudas n.d., Papadimitriou & Steiglitz n.d., Wolsey n.d., Tui 1964, Young 1968, Marchand, Martin, Weismantel & Wosley 2002).

9.1 Hiperplanos de Corte Aplicados a Problemas MIP e IP

A teoria da geração e adição de novos cortes a um determinado problema, para permitir a obtenção da solução discreta, remonta aos trabalhos de Gomory (Gomory 1958, Gomory 1960), Young (Young 1968), Balas (Balas 1969), Chvátal (Chvátal 1973) e Glover (Glover 1973).

Nesta seção serão apresentados os cortes oriundos da visão geométrica de Chvátal-Gomory e da derivação geométrica para cortes convexos generalizada por Glover. Estes cortes apresentam a característica comum de que não são baseados em qualquer particularidade de um problema específico, sendo portanto aplicados a qualquer estrutura de problemas lineares. Em seguida, como contribuição deste trabalho, estes cortes serão implementados em conjunto para a obtenção de um algoritmo capaz de solucionar problemas com função de custo discreta e restrições quasi-convexas.

9.1.1 Cortes Convexos e Derivados Geometricamente

Através da análise geométrica pode-se construir restrições de desigualdade capazes de permitir a solução de problemas lineares MIP ou IP. Estas desigualdades foram inicialmente propostas por (Tui 1964), embora em um contexto distinto. Posteriormente estas desigualdades foram estendidas pelo chamado corte hipercilíndrico (i.e., do inglês *Hypercylindrical Cut*) desenvolvido por Young (Young 1968) para a solução de uma determinada classe de problemas BIP e pelos chamados cortes de intersecção (i.e., do inglês *Intersection Cuts*) desenvolvidos por Balas (Balas 1969). Estes últimos, fundamentados na idéia de que os pontos discretos, que estão ao redor de uma solução contínua relaxada, podem possibilitar a descrição de cortes que conduzirão à solução discreta.

A figura 9.1 exibe o princípio fundamental da construção destes cortes derivados geometricamente (GDC). Partindo da obtenção da solução relaxada para variáveis contínuas de um problema linear MIP ou IP, representado por $\mathbf{x}^*$, obtém-se o hipercubo de aresta unitária que contém esta solução. Do centro deste hipercubo é definida a hiperesfera de raio unitário. Dado o problema ser linear, existirão ao menos n restrições cujo valor na solução relaxada contínua é zero, uma vez que a solução de um problema linear se encontra sempre na extremidade do politopo que define a região factível. Sendo assim, considerando-se a solução do problema linear pelo algoritmo Simplex (Dantzig 2002, Luenberger 1984, Salkin & Mathur n.d.) a tabela ótima resultante irá exibir as n variáveis de folga (i.e do inglês *slack variables*) em função das variáveis originais do problema. Cada variável de folga definirá uma reta, a qual será interceptada pela hiperesfera. Desta intersecção serão resultados n pontos $\mathbf{u}_j$, a partir dos quais será definida a equação da nova desigualdade, exibida na figura 9.1 como $g_{p+1}(\mathbf{x})$. Esta nova desigualdade se somará às já existentes restrições do problema. A repetição sucessiva do procedimento de obtenção da solução relaxada contínua e do acréscimo de uma nova restrição derivada geometricamente conduzirá à convergência da solução relaxada contínua para a solução ótima do problema $\mathbf{x}^{z*}$.

Pela descrição acima do método, verifica-se que cada ponto $\mathbf{u}_j$ é obtido pela solução

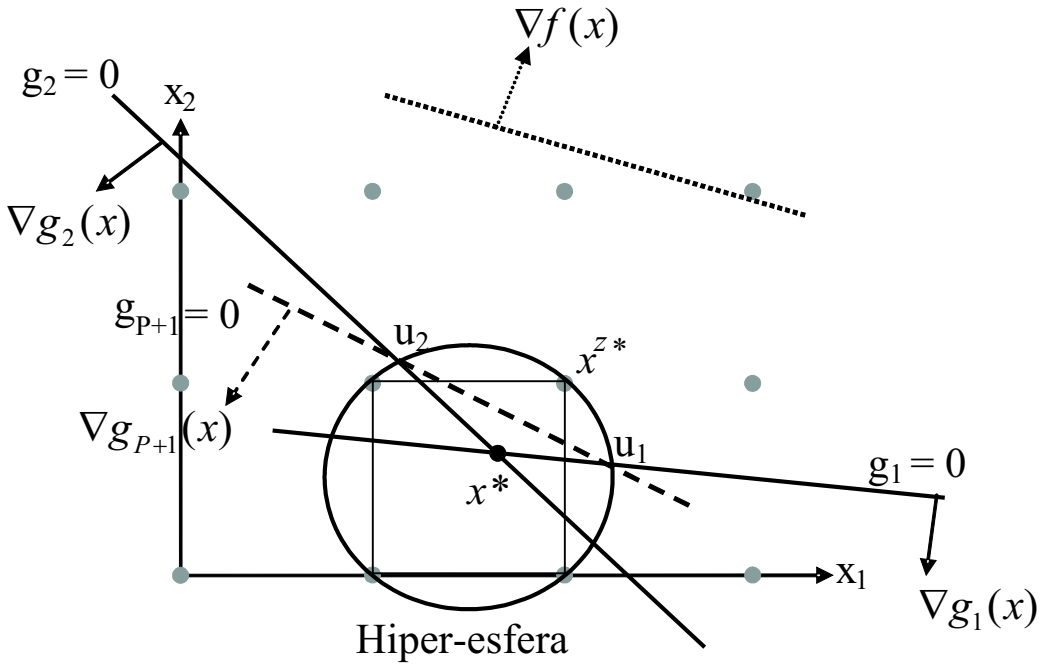


Figura 9.1: Corte derivado geometricamente baseado em uma hiperesfera.

de um sistema de $n-1$ equações lineares e uma equação quadrática correspondente à esfera. Uma opção alternativa, a qual comporá um sistema formado apenas por equações lineares para a determinação dos pontos de intersecção $\mathbf{u}_j$, pode ser obtida utilizando um hiperoctaedro, aqui denominado de hiperdiamante. A figura 9.2 exibe a ilustração gráfica desta variação do método GDC.

De maneira análoga à utilização da esfera, num processo iterativo os cortes são gerados pela intersecção do hiperdiamante com $n-1$ retas obtidas pelas equações de folga. A solução dos consecutivos problemas relaxados fará com que a solução relaxada contínua convirja para a solução ótima do problema $\mathbf{x}^{z*}$.

Posteriormente, Glover (Glover 1973) acabou por desenvolver uma teoria mais genérica, a qual engloba as idéias propostas por Young e Balas, aplicada a uma classe mais extensa de problemas matemáticos. A principal idéia foi generalizar as regiões de intersecção para qualquer função convexa que não possua qualquer solução discreta no seu interior. O Lema a seguir descreve os cortes convexos de Glover no contexto da obtenção da solução contínua relaxada pelo método *Simplex*.

Lema 6 *:Corte Convexo de Glover: Seja S um dado conjunto de pontos factíveis discretos de uma região factível. Se R representa um conjunto convexo em cujo interior não contém pontos de S e se* $\mathbf{x} = \mathbf{x}^*$ *existe uma vizinhança que pode ser removida e que se encontra no interior de R, então para qualquer constante* $\mathbf{w}_j^* > 0$, $j \in$ *conjunto de variáveis não básicas NB, tal que os pontos*

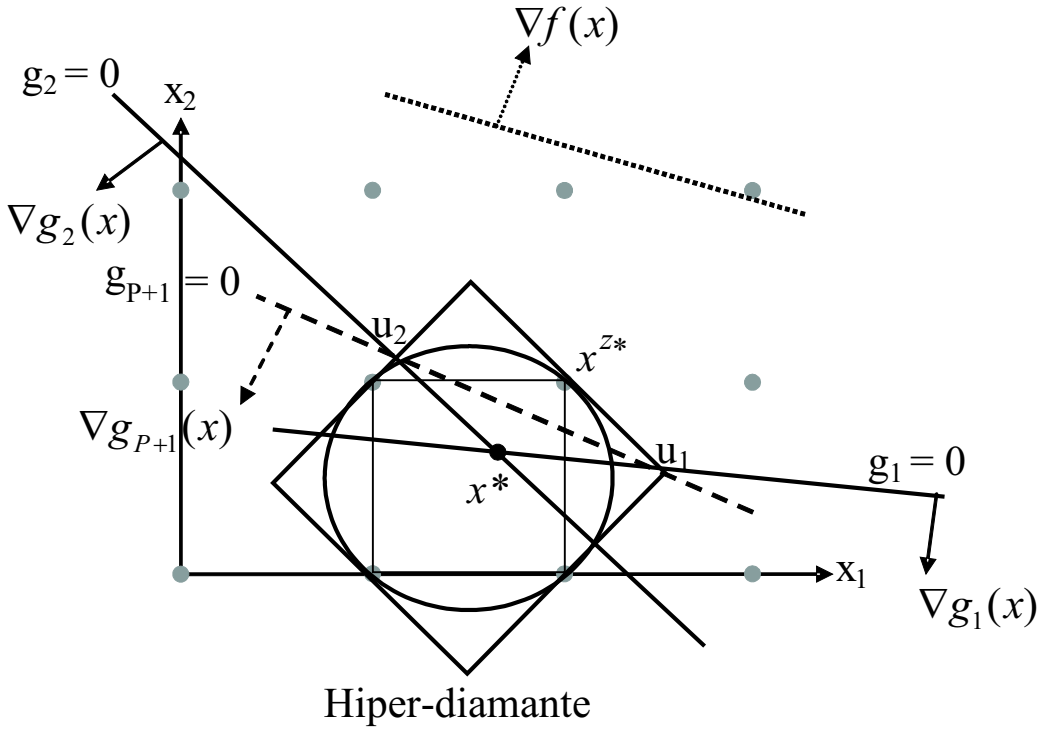


Figura 9.2: Corte derivado geometricamente baseado em um hiperdiamante.

$$\xi_j^* = (\mathbf{x}^* - \alpha_j \mathbf{w}_j^*) \in \mathbb{R}, \quad \forall\, j \in NB$$

onde α_j *são constantes, o corte convexo*

$$\sum_{j \in NB} \mathbf{w}_j / \mathbf{w}_j^* \geq 1 \qquad (c - cut)$$

excluirá o ponto $\mathbf{x} = \mathbf{x}^*$*, mas nunca qualquer ponto em* S*.*

9.1.2 Cortes Chvátal-Gomory

Considerando a descrição de um problema discreto, tal como apresentado a seguir, este problema será denominado puramente discreto quando as constantes A e b forem discretas.

$$\begin{aligned} \mathbf{y} &= c^T \mathbf{x} \mid \mathbf{x} \in \Omega \\ \Omega &\triangleq \{\mathbf{x} \in \mathbb{R}_+^n \mid A\mathbf{x} = b \, e \, \mathbf{x} \in \mathbb{I}^n\} \end{aligned} \tag{9.1}$$

onde, A é uma matriz n por p e b é um vetor de p linhas.

Por definição tem-se que o subconjunto $A\mathbf{x} = b$ restrito ao cone positivo é um politopo.

O subconjunto das soluções $A\mathbf{x} = b$, tal que a equação $A(i,:)\mathbf{x} = b(i)$ é formada por valores puramente discretos para algum i, é formado por politopos que contêm ao menos um ponto discreto em uma das faces. A casca convexa $conv(\mathbf{x})$ também é um politopo onde a equação $A\mathbf{x} = b$ é formada por valores puramente discretos. Isto corresponde ao fato de que cada plano suporte de $conv(\mathbf{x})$ contém um vetor discreto. A visão geométrica de Chvátal consiste em avaliar os planos suportes de $A\mathbf{x} = b$ e movê-los de forma a que se aproximem de $conv(\mathbf{x})$ até que contenham um ponto discreto (Chvátal 1973). Assim, Chvátal demonstrou que o politopo $conv(\mathbf{x})$ pode ser obtido após um número finito de iterações.

De toda forma, o problema remanescente da geração seqüencial dos diversos hiperplanos persistiu até a proposição do algoritmo de Gomory. Este algoritmo gera cortes a partir da solução relaxada contínua, os quais excluem a atual solução relaxada e, paulatinamente, conduzem à solução discreta.

A idéia de Gomory baseia-se no simples fato de que, considerando $A \in \mathbb{R}^{p\times n}$ e o vetor de constantes $u \in \mathbb{R}^p_+$, (9.2) é válida.

$$\sum_{j=1}^{n} \mathbf{u} .* A(:,j)\,\mathbf{x} \leq \mathbf{u} .* \mathbf{b} \tag{9.2}$$

onde $\mathbf{u} .* A(:,j) = [u(1)A(1,j), \ldots, u(p)A(p,j)]^T$ e $\mathbf{u} .* \mathbf{b} = [u(1)\mathbf{b}(1), \ldots, u(p)\mathbf{b}(p)]^T$.

Todavia, dada a desigualdade, (9.3) também é válida.

$$\sum_{j=1}^{n} \lfloor \mathbf{u} .* A(:,j) \rfloor\, \mathbf{x} \leq \mathbf{u} .* \mathbf{b} \tag{9.3}$$

onde, $\lfloor w \rfloor$ representa o arredondamento de w para o menor inteiro próximo à w.

Por fim, (9.4) também é válida, dado que o lado direito da desigualdade (9.3) representa um valor discreto.

$$\sum_{j=1}^{n} \lfloor \mathbf{u} .* A(:,j) \rfloor\, \mathbf{x} \leq \lfloor \mathbf{u} .* \mathbf{b} \rfloor \tag{9.4}$$

A seqüência a seguir mostra um exemplo do corte Chvátal-Gomory.

Considere o poliedro formado pelas equações abaixo:

$$\begin{aligned} 0.6x_1 + 0.8x_2 &\leq 3.6 \\ 4x_1 - x_2 &\leq 11 \\ -x_1 &\leq 1 \\ -x_2 &\leq 1 \end{aligned}$$

Dividindo as inequações por um dos coeficientes que as compõem, desde que diferente de um, teremos:

$$\begin{aligned} x_1 + \lfloor \frac{0.8x_2}{0.6} \rfloor &\leq \frac{3.6}{0.6} \\ \lfloor \frac{0.6x_1}{0.8} \rfloor + x_2 &\leq \frac{3.6}{0.8} \\ x_1 + \lfloor \frac{-x_2}{4} \rfloor &\leq \frac{11}{4} \end{aligned}$$

Arredondando os valores do lado esquerdo das inequações teremos:

$$\begin{aligned} x_1 + x_2 &\leq \lfloor \frac{3.6}{0.6} \rfloor \\ x_2 &\leq \lfloor \frac{3.6}{0.8} \rfloor \\ x_1 - x_2 &\leq \lfloor \frac{11}{4} \rfloor \end{aligned}$$

Por fim arredondando o valor do lado direito das inequações teremos cortes definidos por Chvátal-Gomory.

$$\begin{aligned} x_1 + x_2 &\leq 6 \\ x_2 &\leq 4 \\ x_1 - x_2 &\leq 2 \end{aligned}$$

A figura 9.3 exibe a região factível do problema original Ω, a casca convexa $conv(\mathbf{x})$,

os cortes obtidos e o novo poliedro P_1 definido por estes cortes.

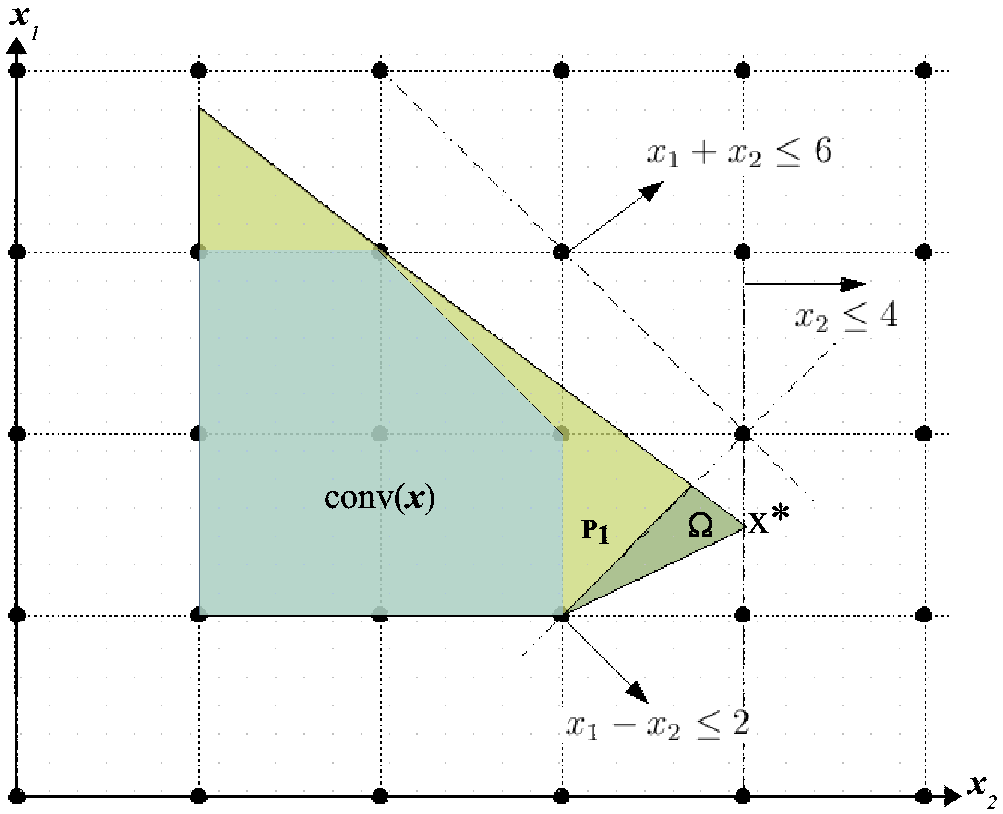


Figura 9.3: Cortes pela visão de Chvátal-Gomory.

9.2 Variação do Método Elipsoidal com Hiperplanos de Corte: MEHC

Uma vez que o método *Elipsoidal* permite a solução de problemas lineares, o conceito dos cortes derivados geometricamente facilmente pode ser implementado para criar uma variação do método capaz de solucionar problemas lineares discretos.

Todavia, a criação de um algoritmo baseado em *Elipsóide* para a solução de problemas IP não constitui um avanço. Isto ocorre porque os métodos já existentes baseados em *Simplex* ou *Pontos Interiores* terão desempenho muito superior em função do tempo gasto para encontrar a solução dos problemas relaxados.

Já para a solução de problemas NIP com função-objetivo linear e restrições quasi-convexas, um algoritmo baseado em *Elipsóide* passa a ter valor, já que existem poucos métodos capazes de garantir a convergência global destes problemas (Westerlund & Pörn 2002), especialmente com base em hiperplanos de corte.

Como primeiro ponto a ser considerado, uma vez que tanto os cortes derivados geometricamente, quanto os cortes Chvátal-Gomory baseiam-se na existência de equações lineares, um conjunto destas equações deve ser utilizado para a determinação do politopo inicial. Qualquer politopo que contenha a região factível não linear pode ser utilizado. Para problemas em que a determinação deste politopo seja complexa, o hiperparalelepípedo definido pelos intervalos de variação das variáveis constitui uma opção.

Em seqüência, após solucionado o problema relaxado, em virtude da existência de restrições não lineares é provável que a solução ótima do problema relaxado $\mathbf{x}^*$ não se encontre na extremidade do politopo inicial. Neste caso, deve-se utilizar exclusivamente os cortes Chvátal-Gomory para a geração da nova desigualdade a ser adicionada ao problema a partir do semi-espaço $H_-^{\bar{\nabla} f(\mathbf{x}^*),\mathbf{x}^*}$.

Quando a solução ótima do problema relaxado $\mathbf{x}^*$ se encontrar na extremidade do politopo corrente, ambos os cortes GDC e Chvátal-Gomory devem ser utilizados na obtenção das novas restrições. Dado que a proposição original dos cortes GDC utiliza as variáveis de folga para a determinação dos pontos $\mathbf{u}_j$, estas variáveis devem ser substituídas pela combinação das n restrições, que apresentam valor zero no ponto $\mathbf{x}*$. Quando combinadas em conjuntos de $n-1$ restrições, esta combinação das $n-1$ restrições definirá uma reta, a qual será interceptada por uma face do hiperdiamante e permitirá determinar os n pontos $\mathbf{u}_j$. A partir dos pontos $\mathbf{u}_j$ são gerados $n-1$ vetores que definirão o hiperplano da nova restrição de desigualdade a ser acrescentada ao problema.

A figura 9.4 exibe uma seqüência de criação de uma nova restrição para um algoritmo baseado em *Elipsóide*, a partir do algoritmo GDC. O processo começa com uma solução obtida por relaxação e um politopo inicial P apresentados na subfigura 9.4a. Em seguida, a subfigura 9.4b exibe o hipercubo unitário que contém a solução relaxada. Através dos vetores que são definidos pelos cantos do hipercubo em direção ao centro deste, as faces do hiperdiamante são definidas como exibido na subfigura 9.4c. Na subfigura 9.4d são apresentadas as n restrições que passam pelo solução relaxada e, portanto, apresentam valor nulo. Cada conjunto de $n-1$ destas restrições que intercepta uma face do hiperdiamante irá definir um ponto $\mathbf{u}_j$, tal como mostrado na subfigura 9.4e. Por fim, a subfigura 9.4f exibe os n pontos $\mathbf{u}_j$, os quais definirão o hiperplano cujo vetor normal é ∇g_{p+1}. Conseqüentemente, a nova restrição será definida como $\nabla g_{p+1}^T(\mathbf{x}-\mathbf{u}_j) \leq 0$.

O problema (8.1) será aqui tratado e é representado abaixo para a comodidade do leitor:

$$min\ f(\mathbf{x}) \tag{9.5}$$

$$s.a.\ \mathbf{x} \in \Omega$$

$$\Omega \triangleq \{\mathbf{x}(1:z) \in \mathbb{I}_+^z\ e\ \mathbf{x}(z+1:z+d) \in \mathbb{R}^d \mid g_i(\mathbf{x}) \leq 0\ \forall\ i = 1,\ldots,p\}$$

onde, $z+d=n$, $f(\cdot):\mathbb{R}^n \longmapsto \mathbb{R}$ e $g(\cdot):\mathbb{R}^n \longmapsto \mathbb{R}^p$.

O Algoritmo 11 representa a codificação do método *Elipsoidal* com hiperplanos de corte, $MEHC$, baseado no fluxograma geral exibido na figura 8.3. Como um processo de aceleração da busca da solução do problema obtido por relaxação das variáveis, foi incluída a heurística da vizinhança e a redução do elipsóide $\hat{E}$. A heurística tenta encontrar um ponto factível próximo ao centro do elipsóide corrente, na direção do gradiente[1] da função-objetivo. Este ponto pode ser entendido como um LSC. Uma vez encontrado um LSC no semi-elipsóide que será preservado, pode-se reduzir o elipsóide $\hat{E}$ que será utilizado como elipsóide inicial no próximo problema obtido por relaxação. Este processo garante que a solução ótima do problema $\mathbf{x}^{z*}$ será preservada, caso se encontre no elipsóide inicial E_0 fornecido como parâmetro. Também deve ser observado que o problema relaxado é solucionado para todas as restrições, lineares ou não, enquanto que apenas as restrições lineares são utilizadas para a adição de novas restrições por duas funções. A primeira é a função $ChvatalGomory_{Cut}$ que adiciona restrições segundo a proposição de Chvátal-Gomory descrita na seção 9.1.2 pela seqüência de equações descritas por (9.2), (9.3) e (9.4). A segunda é a função GDC_{Cut} que adiciona restrições segundo a proposição dos cortes descritos geometricamente descrita na seção 9.1.1 e exemplificada na seqüência exibida na figura 9.4. O critério de parada do algoritmo verifica se a solução do problema obtido por relaxação é uma solução factível. Para os casos em que durante a convergência do algoritmo não foi possível adicionar uma nova restrição para a criação de um novo problema relaxado, o algoritmo terminará com erro. Por fim, uma vez que a adição de novas restrições constitui um dos maiores limitadores para os métodos de plano de corte, foi incluído no algoritmo um critério de exclusão de restrições baseado no fato destas restrições terem, necessariamente, que interceptar o elipsóide $\hat{E}$.

Deve-se ainda observar que a solução do problema obtido por relaxação pode ser adicionada ao algoritmo $HISPE$ para criar uma versão do algoritmo $MEPC$ que considera valores já calculados para o cálculo do novo elipsóide E_{k+1}.

[1] Nos problemas tratados por este método a função-objetivo é linear.

9.3 Limitações da Aplicação Prática do $MEHC$

Embora Gomory tenha provado que um algoritmo *Simplex* adicionado de uma seqüência de restrições definidas por cortes Chvátal-Gomory converge globalmente para um problema linear IP e MIP, o mesmo não se pode dizer quanto ao $MEHP$ aplicado a problemas NIP. Assim, não existe prova de convergência global para o método $MEHC$.

Além disto, as restrições por cortes GCD não asseguram a convergência global, mesmo quando estas restrições são associadas ao algoritmo *Simplex*. A maior razão para este fato é que muitas restrições adicionadas não geram redução considerável entre o politopo atual que descreve a região factível e a casca convexa do problemas em questão. Este fato acarreta a existência de uma seqüência de novas restrições que, embora excluam a extremidade do politopo atual que define a solução do problema contínuo $\mathbf{x}^*$, se aproximam da forma de uma curvatura convexa que se aproxima muito vagarosamente da solução discreta $\mathbf{x}^{z*}$.

No caso particular do algoritmo $MEHC$ os dois problemas supracitados são exacerbados por alguns fatos. O primeiro decorre da imprecisão da solução $\mathbf{x}^*$ encontrada pelo laço do algoritmo *Elipsoidal* à medida que o politopo na região próxima $\mathbf{x}^*$ apresenta diversos vértices muito próximos. Embora o volume do elipsóide possa ser reduzido de acordo com a precisão desejada, nada garante que para todos os eixos esta redução será similar ou que não haverá perda de positividade da matriz Q_k. Outro fato é a determinação das restrições que são nulas no ponto $\mathbf{x}^*$. A teoria inicial dos cortes parte das variáveis de folga, enquanto no método proposto verifica-se o valor das restrições. Este procedimento aumenta o número de cálculos e conseqüentemente o erro por arredondamento. Pode ainda ocorrer a existência de um número maior do que n restrições com valor zero, sendo que as n restrições que definem o vértice do politopo devem ser determinadas. Finalmente, no corte GDC existe a solução do sistema que define a intersecção das faces do hiperdiamante com as restrições com valor zero, implementado aqui pelo cálculo de uma matriz inversa, e do sistema que calcula o vetor ortogonal aos pontos $\mathbf{u}_j$ que define a restrição a ser adicionada. Apesar deste procedimento não possuir restrições teóricas, na implementação prática ocorreram diversos erros por arredondamento durante o processo de convergência à medida que a dimensão foi aumentada.

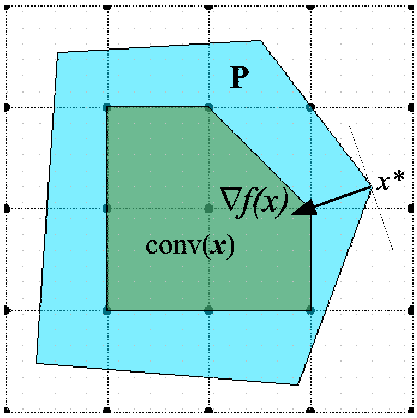
Como observação final sobre o método $MEHC$ proposto, independentemente das limitações aqui apresentadas, o conceito de definir hiperplanos para colaborar no processo de solução de problemas NIP e MINP é de fundamental importância para esta pesquisa, dado que esta idéia permitirá a construção de novos algoritmos baseados no conceito de *Branch-and-Cut*, o qual será apresentado no capítulo 10.

9.4 Conclusões do Capítulo

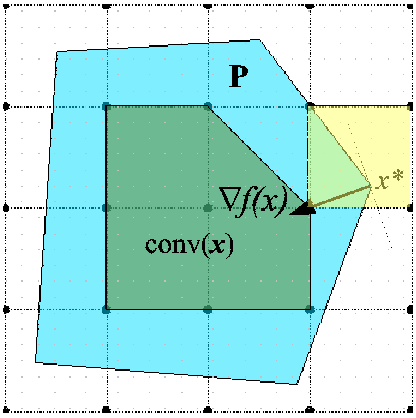
Neste capítulo foram apresentados os fundamentos para o entendimento dos métodos de otimização que solucionam problemas onde as variáveis associadas não são exclusivamente variáveis contínuas.

Em seqüência foi apresentada a teoria para a geração de novas restrições que permitem a obtenção da solução discreta de um problema a partir da solução obtida por relaxação. Através desta teoria, foi formalizado o algoritmo $MEHC$, bem como apresentadas e discutidas as limitações para a utilização deste algoritmo em casos de dimensão elevada.

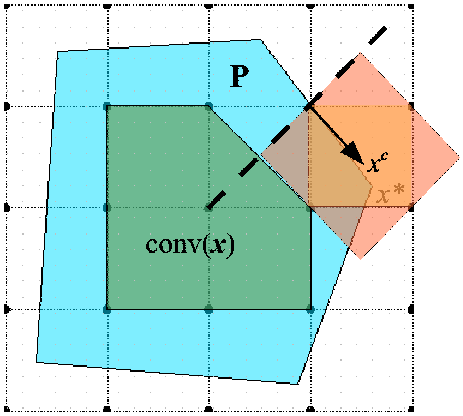
No capítulo seguinte serão detalhados os métodos baseados em *Elipsóide* para a solução de problemas MINP e NIP através do processo de enumeração.



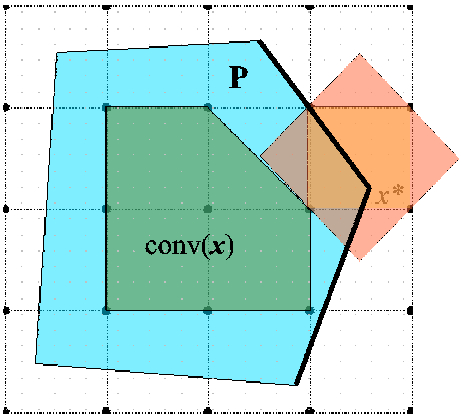
a) Solução $\mathbf{x}^*$ de um problema relaxado.



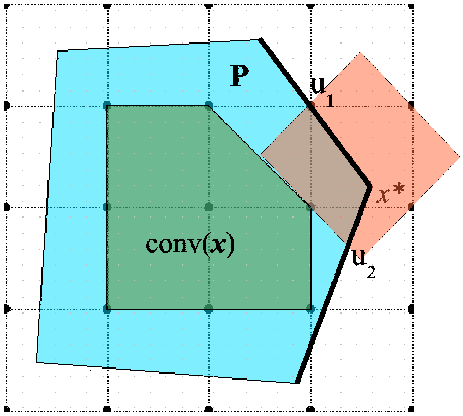
b) Hipercubo unitário que contém $\mathbf{x}^*$.



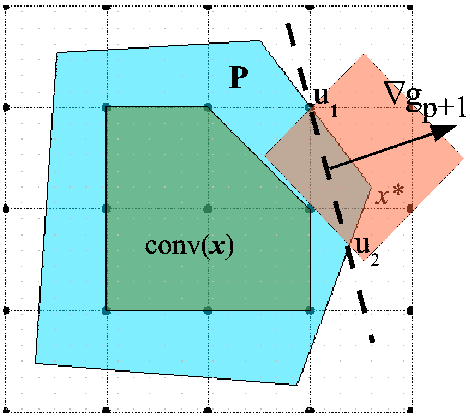
c) Hiperdiamente formado a partir do hipercubo unitário.



d) Conjunto de n restrições cujos valores são zero .



e) Pontos $\mathbf{u}_j$ definidos a partir das restrições e do hiperdiamante.



f) Nova restrição $\nabla g_{p+1}^T(\mathbf{x} - \mathbf{u}_j) \leq 0$.

Figura 9.4: Seqüência de criação de uma nova restrição por GDC para um algoritmo baseado em *Elipsóide*.

Algorithm 11 *Elipsoidal* com Hiperplanos de Corte

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^{z*}$ será procurada de acordo com o problema definido por(9.5). Um conjunto de restrições $g_i(\cdot) \forall\ i = 1, \ldots, p$ sendo que $g_j(\cdot) \forall\ 1 = 1, \ldots, \hat{p}$ e $n \leq \hat{p} \leq p$ são restrições lineares que definem um politopo.

Output: A melhor solução factível $\mathbf{x}^z$ encontrada. Para um problema infactível ou para $\mathbf{x}^{z*} \notin E_0$ ter-se-á $\mathbf{x}^z = \emptyset$.

function $MEHC(E_0)$
  $\hat{E} \leftarrow E_k$ ▷ Menor elipsóide que contém LSC.
  $\mathbf{x}^z \leftarrow \emptyset$ ▷ Zera o melhor ponto factível
  loop ▷ Laço de movimentação do LIC.
    $k \leftarrow 0$
    $E_k \leftarrow \hat{E}$
    loop ▷ Laço para o problema obtido por relaxação.
      if $g_i(\mathbf{x}_k) \leq 0\ \forall\ i = 1, \ldots, p$ **then**
        $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{x}_k)$
        $\{\mathbf{w}^z \mid \mathbf{w}^z(i) = \lceil \mathbf{x} \rceil\ \forall\ \bar{\nabla}_k(i) \geq 0\ e\ i = 1, \ldots, n \mid \mathbf{w}^z(j) = \lfloor \mathbf{x} \rfloor\ \forall\ \bar{\nabla}_k(j) < 0\ e\ j = 1, \ldots, n\}$ ▷ Heurística da vizinhança.
        if $\mathbf{w}^z \in \Omega$ e $f(\mathbf{w}^z) < f(\mathbf{x}^z)$ **then**
          $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{w}^z)$ ▷ Define o semi-espaço para cálculo do novo elipsóide.
          $\mathbf{x}^z \leftarrow \mathbf{w}^z$ ▷ Novo LSC.
          if $\bar{\nabla}_k = 0$ **then**
            return $\mathbf{x}^z$ ▷ Termine o algoritmo.
          end if
        else
          $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{x}_k)$ ▷ Define o semi-espaço para cálculo do novo elipsóide.
          $\mathbf{w}^z \leftarrow \mathbf{x}_k$
          if $\bar{\nabla}_k = 0$ **then**
            stop laço para o problema obtido por relaxação.
          end if
        end if
      else
        $\bar{\nabla}_k \leftarrow \bar{\nabla} g_i(\mathbf{x}_k)$ *para algum* $g_i(\mathbf{x}_k) > 0,\ \mid i = 1, \ldots, p$ ▷ Define o semi-espaço para cálculo do novo elipsóide.
        $\mathbf{w}^z \leftarrow \mathbf{x}_k$
      end if
      $E_{k+1} \leftarrow (E_k \cap H_-^{\bar{\nabla}_k, \mathbf{w}^z})$ ▷ Vide (2.14).
      if $\mathbf{x}^z \in E_{k+1}$ e $\mathbf{x}^z \neq \emptyset$ **then**
        $\hat{E} \leftarrow E_{k+1}$ ▷ Atualiza menor elipsóide que contém LSC.
      end if
      if vol $(E_{k+1}) \simeq 0$ ou $vol(E_{k+1}) \simeq vol(E_k)$ **then**
        stop laço para o problema obtido por relaxação.
      end if
      $k \leftarrow k + 1$
    end loop
    if $\mathbf{x}_k \in \Omega$ **then**
      return $\mathbf{x}_k$ ▷ Termine o algoritmo.
    end if
    $M\bar{\nabla} = \bar{\nabla} g_i(\mathbf{x}_k) \mid g_i(\mathbf{x}_k) = 0\ \forall\ g_i(\cdot)\ linear$
    if $rank(M\bar{\nabla}) \neq n$ **then**
      $M\bar{\nabla} = \bar{\nabla} f(\mathbf{x}_k)$ ▷ Não está na extremidade de um politopo.
      add restrições $ChvatalGomory_{Cut}(\mathbf{x}_k, M\bar{\nabla})$ ▷ Vide (9.4).
    else
      add restrições $ChvatalGomory_{Cut}(\mathbf{x}_k, M\bar{\nabla})$ ▷ Vide (9.4).
      add restrições $GDC_{Cut}(\mathbf{x}_k, M\bar{\nabla})$ ▷ Vide Fig.9.4.
    end if
    if *Não foi adicionada uma nova restrição* **then**
      return $\mathbf{x}_k$ ▷ Termine o algoritmo com falha.
    end if
    remove restrições $g_i(\cdot)\ \forall\ g_i(\cdot)\ linear \mid (H^{\bar{\nabla} g_i(\hat{\mathbf{w}}_i), \hat{\mathbf{w}}_i} \cap \hat{E}) = \emptyset \mid g_i(\hat{\mathbf{w}}_i) = 0$
    ▷ Exclui restrições inúteis.
  end loop
end function

Capítulo 10

Métodos Elipsoidais e Poliedro-Elipsoidais com Enumeração Aplicados a Problemas Discretos e Mistos

Neste capítulo é apresentada a teoria para a obtenção de métodos para a solução de problemas NIP e MINP através do processo de enumeração, implícita ou explícita, do conjunto de pontos discretos ou mistos discretos e contínuos existentes na região de interesse de busca.

Neste capítulo, inicialmente é definido um método baseado em enumeração explícita para a solução de problemas NIP. No decorrer do capítulo, é apresentada uma introdução ao método *Branch-and-Bound*, bem como a busca proposta por este método é aplicada para a implementação de uma nova variação de um método baseado em *Elipsóide* com enumeração para a solução de problemas NIP e MINP. Um método *Branch-and-Cut* também é definido e utilizado para criar duas variações de métodos baseados em *Elipsóide* para problemas NIP e MINP. Estes quatro algoritmos, juntamente com uma variação do método *Elipsoidal* para a solução de problemas com igualdades lineares, concluem a contribuição desta Tese para a solução de problemas mono-objetivo MINP e NIP.

Maiores detalhes sobre os temas tratados nas seções apresentadas neste capítulo podem ser conseguidos nas referências (Land & Doig 1960, Little, Murty, Sweeney & Kare 1963, Balas 1965, Dakin 1965, Crowder, Johnson & Padberg 1983, Salkin & Mathur n.d., Savelsbergh & Nemhauser 1993, Floudas n.d., Gu, Nemhauser & Savelsbergh 1995, Linderoth & Savelsbergh 1997, Wolsey n.d., Cordier, Marchand, Laundy & Wolsey 1999, Padberg 2005) e no trabalho (Moura, Smith & Takahashi 2007) apresentado na conferência MIP 2007

no Centre de Recherches Mathématiques da Université de Montréal.

10.1 Variação do Método *Elipsoidal* com Enumeração Explícita: *MEEE*

O problema (8.1) será aqui tratado e é representado abaixo para a comodidade do leitor:

$$min\ f(\mathbf{x}) \tag{10.1}$$

$$s.a.\ \mathbf{x} \in \Omega$$

$$\Omega \triangleq \{\mathbf{x}(1:z) \in \mathbb{I}_+^z\ e\ \mathbf{x}(z+1:z+d) \in \mathbb{R}^d \mid g_i(\mathbf{x}) \leq 0\ \forall\ i = 1,\ldots,p\}$$

onde, $z+d=n$, $f(\cdot):\mathbb{R}^n \longmapsto \mathbb{R}$ e $g(\cdot):\mathbb{R}^n \longmapsto \mathbb{R}^p$.

A forma mais simples de se criar uma variação do método *Elipsoidal* capaz de solucionar problemas discretos como os definidos por (10.1) para $z=n$, através da utilização do procedimento de enumeração, é enumerar explicitamente o conjunto das variáveis inteiras de um problema, conforme já apresentado na figura 8.4. Tal como descrito na seção 8.3, durante o processo de convergência de um algoritmo *Elipsoidal* na solução de um problema discreto, ou misto discreto e contínuo, deve-se garantir a existência de ao menos um ponto $\{\mathbf{x} \mid \mathbf{x}(1:z) \in \mathbb{I}^z\}$ dentro da metade do elipsóide que se pretende preservar, antes da geração do elipsóide da iteração seguinte.

Uma vez garantida a existência de uma solução dentro da metade do elipsóide que será preservada, o procedimento para a obtenção do novo centro e do novo elipsóide segue exatamente os mesmos passos definidos para estes cálculos na versão do algoritmo que soluciona problemas com variáveis puramente contínuas, vide (2.11) página 24. Dado que o elipsóide define um volume compacto, pode-se afirmar que o conjunto de soluções inteiras que podem ser encontradas no seu interior, quando o problema em questão é puramente discreto, é um conjunto finito. Por conseqüência, a busca de uma solução inteira na metade do elipsóide que se deseja preservar também será um processo finito, sendo também finita a convergência do método. Ressalta-se ainda que o processo de divisão do elipsóide permite a avaliação de apenas um subconjunto das possíveis soluções e que a enumeração se faz necessária apenas até que um ponto $\{\mathbf{x} \mid \mathbf{x}(1:z) \in \mathbb{I}^z\}$ seja encontrado.

Assim, para certas classes de problemas, a enumeração explícita ocorrerá em um subconjunto de soluções menor do que o conjunto total das soluções inteiras factíveis para o

problema. Claramente existem problemas onde será necessário avaliar todo o conjunto de soluções , como por exemplo quando o conjunto de soluções for unitário. Todavia, deve-se considerar que para alguns problemas um ponto poderá ser enumerado mais de uma vez durante o processo de convergência. O Lema 7 formaliza este raciocínio.

Lema 7 - *Convergência e Enumeração Finita para IP: Considere o problema definido por (10.1) onde $z = n$, um elipsóide não degenerado e de volume finito definido por $E_0 = \{\mathbf{x} \in \mathbb{R}^n \mid (\mathbf{x} - \mathbf{x}_0)^T Q_0^{-1} (\mathbf{x} - \mathbf{x}_0) \leq 1 \mid \Omega \subset E_0\}$ e um subconjunto das soluções factíveis $\hat{X} \triangleq \{E_0 \cap \Omega\}$. Um algoritmo baseado no método Elipsoidal, com critérios de parada* vol $(E_{k+1}) \simeq 0$ *ou $vol(E_{k+1}) \simeq vol(E_k)$ e que implementa o processo de enumeração explícita para garantir a escolha da metade do elipsóide a ser preservado, convergirá em um número finito de iterações e avaliará um número de pontos discretos igual ou inferior ao número de elementos do conjunto de soluções factíveis inteiras.*

Prova:

i) *Dado o problema ser puramente discreto temos que $\hat{X}$ define um conjunto com um número finito de soluções.*

ii) *Para demonstrar que a avaliação do número de pontos discretos é finita, suponha por absurdo que a convergência de um algoritmo baseado no método Elipsoidal enumere um número infinito de pontos durante o seu processo de convergência para um problema IP. Isto dar-se-ia por dois motivos. O primeiro ocorreria caso o algoritmo avaliasse em que em qualquer iteração durante o processo de convergência fosse avaliado um conjunto infinito de pontos discretos, o que contradiz o item i. O segundo ocorreria caso o algoritmo avaliasse um número finito de pontos discretos infinitas vezes. Esta hipótese ocorreria caso a convergência ocorresse num número infinito de iterações, o que contradiz os critérios de parada* vol $(E_{k+1}) \simeq 0$ *ou $vol(E_{k+1}) \simeq vol(E_k)$. Logo, por reductio ad absurdum a convergência ocorrerá com a enumeração de um número finito de pontos discretos.*

iii) *Para demonstrar que a avaliação do número de pontos discretos pode ser inferior ao número de pontos do conjunto de soluções $\hat{X}$, considere o conjuntos dos pontos já enumerados até a iteração k denominado $\hat{X}_k$, tal que $\hat{X}_k \subseteq \hat{X}$. Considerando-se que todo ponto $\mathbf{x}_k$ corresponderá ao LSC obtido por enumeração, os pontos do subconjunto $\varrho = \{\mathbf{x} \in (H_-^{\bar{\nabla} f(\mathbf{x}_{k-1}), \mathbf{x}_{k-1}} \cap \hat{X}) \mid \mathbf{x} \neq \mathbf{x}_{k-1}\}$ não foram enumerados na iteração $k-1$. Assim, durante a convergência do algoritmo, caso exista o subconjunto $\nu = \{\mathbf{x} \in \varrho \mid \mathbf{x} \notin ((H_-^{\bar{\nabla} f(\mathbf{x}_k), \mathbf{x}_k} \cap \hat{X}) \cup \hat{X}_{k-1}) \mid \mathbf{x} \notin E_{k+1}\}$ para alguma iteração*

k, os pontos do conjunto ν não serão avaliados na iteração k, não terão sido avaliados por qualquer outra iteração anterior a k e não serão avaliados por qualquer iteração posterior a k. Logo, o algoritmo terá avaliado um número de elementos menor que o número de elementos do conjunto de soluções factíveis inteiras.

O algoritmo 12 implementa uma variação do algoritmo *Elipsoidal* com enumeração explícita para a solução de problemas discretos denominada $MEEE$. A função $BuscaInt$ representa a rotina de enumeração explícita de um ponto discreto factível. Considera-se que esta rotina será interrompida ao primeiro ponto discreto factível encontrado. Para facilitar a representação do algoritmo, define-se que $f(\emptyset) = \infty_+$.

O algoritmo 12 constitui-se basicamente do algoritmo *Elipsoidal* com a inclusão da busca por um ponto factível discreto para garantir a preservação do semi-elipsóide correto. Três pontos devem ser destacados nesse algoritmo. Claramente, a mesma busca por um ponto factível discreto pode ser adicionada ao algoritmo $HISPE$ para criar uma versão do algoritmo $MEEE$ que considera valores já calculados para o cálculo do novo elipsóide E_{k+1}. Neste caso, poder-se-iam utilizar os pontos infactíveis testados na função $BuscaInt$ para o cálculo dos hiperplanos que cortam o elipsóide corrente.

O primeiro é definido pelo teste $((f(\mathbf{x}^z) - f(\mathbf{x}_k)) \leq \hat{\epsilon})$ ou $(\frac{\bar{\nabla}_k^T}{|\bar{\nabla}_k|}(\mathbf{x}^z - \mathbf{x}_k) \leq \hat{\epsilon})$ que implementa dois novos critérios de parada para o algoritmo. Este teste é executado apenas quando não existe um ponto factível no semi-elipsóide definido por $E_k \cap H_-^{\bar{\nabla}_k, \mathbf{x}_k}$. Por conseguinte, o ponto $\mathbf{x}_k$ define o atual LIC. Os novos critérios de parada se baseiam na diferença entre o valor da função-objetivo no LIC e no atual LSC, definido por $\mathbf{x}^z$, e na proximidade de $\mathbf{x}^z$ ao hiperplano definido por LIC. Para uma função-objetivo linear com coeficientes puramente discretos a diferença para o valor da função objetivo avaliada em LIC e $\mathbf{x}^z$ deve ser maior que um. Assim o valor do erro aceitável pare este caso deve ser definido como um, enquanto nos demais casos deve-se definir o valor do erro aceitável próximo de zero.

O segundo ponto também diz respeito a um critério de parada adicional. O critério incluído no teste $autovalores(Q_k) < \frac{1}{2}$ verifica se todos os autovalores de Q_k são menores do que meio, o que implica que o elipsóide atual E_k é menor que uma hiperesfera de raio unitário. Conseqüentemente, somente existe um único ponto discreto no seu interior.

O terceiro ponto é definido pelo parâmetro $H_+^{\bar{\nabla}(\mathbf{x}^z), \mathbf{x}^z}$ da função $BuscaInt$, o qual tenta acelerar o término do algoritmo buscando sempre uma solução melhor que o ponto $\mathbf{x}^z$, o qual define o LSC. Quando $\mathbf{x}^z$ já é a solução ótima e não se inclui este teste, o algoritmo realiza buscas consecutivas numa região onde não existe solução melhor que $\mathbf{x}^z$ até que o volume do elipsóide convirja para zero.

A figura 11.1 exibe uma seqüência de quatro momentos do processo de convergência

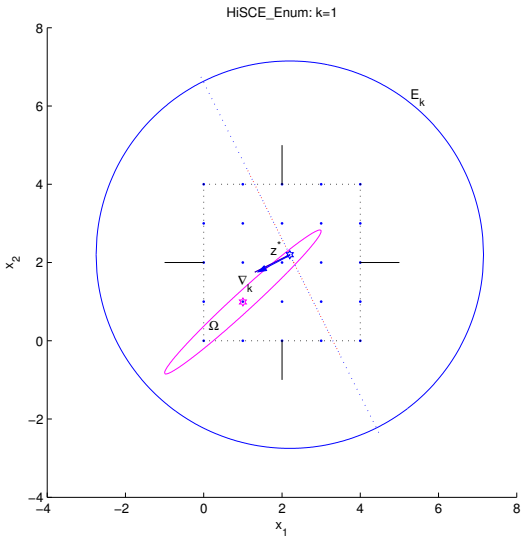
do algoritmo $MEEE$. A subfigura a apresenta o elipsóide inicial, a região factível Ω e o vetor $\bar{\nabla}_k$ tal que $(H_-^{\bar{\nabla}_k,\mathbf{x}_k} \cap \Omega \cap E_k) = \emptyset$. Conseqüentemente, para se obter o elipsóide E_{k+1} exibido na subfigura b foi necessária a enumeração de todos os pontos discretos na região $(H_-^{\bar{\nabla}_k,\mathbf{x}_k} \cap E_k)$, o que certamente não constituiu um processo eficiente de enumeração. Já as subfiguras c e d ilustram duas possibilidades de parada distintas para o método $MEEE$. A primeira, subfigura c, ocorre em função da ocorrência de todos os autovalores do elipsóide E_{k+1} serem menores que meio. A segunda , subfigura d, ocorre em função de não existir um ponto discreto factível entre os hiperplanos definidos pelos limites LIC e LSC diferente de $\mathbf{x}^z$.

Algorithm 12 *Elipsoidal* com Enumeração Explícita

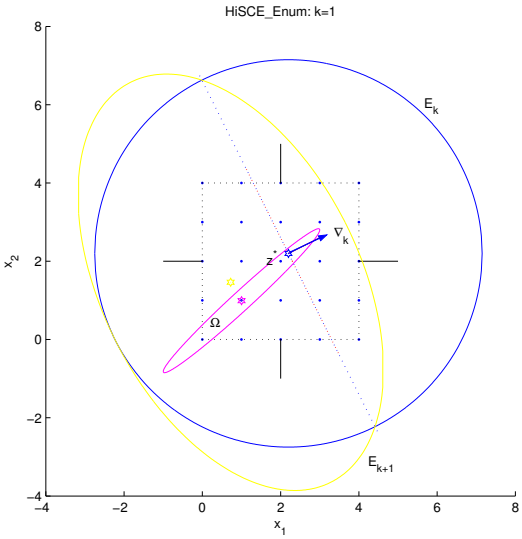
Input: Um elipsóide inicial E_0, onde a solução ótima inteira $\mathbf{x}^{z*}$ será procurada de acordo com o problema definido por (10.1). Um valor para precisão para o teste de limites $\hat{\epsilon} > 0$.

Output: A melhor solução factível inteira $\mathbf{x}^z$ encontrada. Para um problema infactível ou para $\mathbf{x}^{z*} \notin E_0$ ter-se-á $\mathbf{x}^z = \emptyset$.

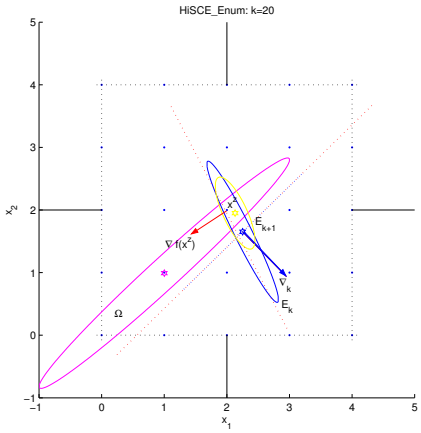
1: **function** $MEEE(E_0, \hat{\epsilon})$
2: $k \leftarrow 0$
3: $\mathbf{x}^z \leftarrow \emptyset$ ▷ Zere a melhor solução factível.
4: **loop** ▷ Laço de movimentação do LSC.
5: **if** $g_i(\mathbf{x}_k) \leq 0 \; \forall \; i = 1, \ldots, p$ **then**
6: $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{x}_k)$
7: **if** $f(\mathbf{x}_k) \geq f(\mathbf{x}^z)$ **then** ▷ Usa $f(\mathbf{x}^z)$ como restrição.
8: $\mathbf{w}^z \leftarrow \mathbf{x}_k$
9: **else**
10: $\mathbf{w}^z \leftarrow BuscaInt(E_k, H_-^{\bar{\nabla}_k, \mathbf{x}_k}, H_+^{\bar{\nabla} f(\mathbf{x}^z), \mathbf{x}^z})$ ▷ $\mathbf{w}^z = \{\mathbf{x} \mid \mathbf{x} \in (Q_k \cap H_-^{\bar{\nabla}_k, \mathbf{x}_k} \cap \Omega) \; e \; \mathbf{x} \notin H_+^{\bar{\nabla}(\mathbf{x}^z), \mathbf{x}^z}\}$.
11:
12: **if** $\mathbf{w}^z == \emptyset$ **then**
13: **if** $((f(\mathbf{x}^z) - f(\mathbf{x}_k)) \leq \hat{\epsilon})$ ou $(\frac{\bar{\nabla}_k^T}{|\bar{\nabla}_k|} * (\mathbf{x}^z - \mathbf{x}_k) \leq \hat{\epsilon})$ **then**
14: **return** $\mathbf{x}^z$ ▷ Termine o algoritmo.
15: **end if**
16: $\bar{\nabla}_k \leftarrow -\bar{\nabla}_k$ ▷ $\mathbf{x}_k$ define o LIC.
17: $\mathbf{w}^z \leftarrow BuscaInt(E_k, H_-^{\bar{\nabla}_k, \mathbf{x}_k}, H_+^{\bar{\nabla} f(\mathbf{x}^z), \mathbf{x}^z})$ ▷ Busca um ponto factível entre LSC e LIC.
18: **if** $\mathbf{w}^z == \emptyset$ **then**
19: **return** $\mathbf{x}^z$ ▷ Termine o algoritmo.
20: **end if**
21: **end if**
22: $\mathbf{x}^z \leftarrow \mathbf{w}^z$
23: $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{x}^z)$ ▷ Novo LSC.
24: **end if**
25: **else**
26: $\bar{\nabla}_k \leftarrow \bar{\nabla} g_i(\mathbf{x}_k)$ *para algum* $g_i(\mathbf{x}_k) > 0, \mid i = 1, \ldots, p$
27: $\mathbf{w}^z \leftarrow \mathbf{x}_k$
28: **end if**
29: $E_{k+1} \leftarrow (E_k \cap H_-^{\bar{\nabla}_k, \mathbf{w}^z})$ ▷ Veja (2.11).
30: **if** vol $(E_{k+1}) \simeq 0$ ou $vol(E_{k+1}) \simeq vol(E_k)$ ou $autovalores(Q_k) < \frac{1}{2}$ **then**
31: **return** $\mathbf{x}^z$ ▷ Termine o algoritmo.
32: **end if**
33: $k \leftarrow k + 1$
34: **end loop**
35: **end function**



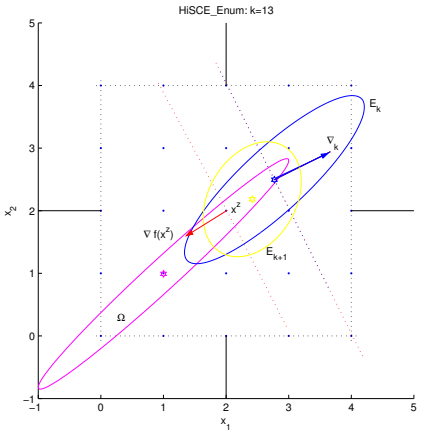
a) Elipsóide inicial E_k.



b) Elipsóide E_{k+1} após inversão da direção de $\bar{\nabla}_k$.



c) Parada por autovalores de $Q_k \leq \frac{1}{2}$.



d) Parada após inversão da direção de $\bar{\nabla}_k$.

Figura 10.1: Ilustração de uma seqüência de convergência do algoritmo $MEEE$.

10.2 Variações do Método *Elipsoidal* com Enumeração por *Branch-and-Bound*: *MEEBB*

Embora o processo de enumeração explícita garanta a existência de um algoritmo baseado em elipsóide capaz de convergir em um número finito de iterações, quando aplicado a um problema NIP, este processo não garante que o desempenho do algoritmo seja eficiente. À medida que a dimensão do problema aumenta, o conjunto de soluções inteiras cresce exponencialmente, o que acaba por inviabilizar muitas vezes a sua completa enumeração. Similarmente, o esforço para apenas encontrar um ponto discreto factível no interior da metade do elipsóide que se pretende preservar pode se tornar impraticável em problemas com região factível pequena no interior de elipsóides de grande volume. Este fato é principalmente ocasionado pelo número de pontos discretos infactíveis que serão avaliados por iteração até que se encontre um ponto factível. Por fim, a restrição de solução de problemas puramente discretos também acaba por limitar a aplicação prática do método de enumeração explícita.

Se os fatores supracitados forem melhor avaliados, perceber-se-á que as limitações da eficiência destes fatores são definidas pela forma como a enumeração foi implementada e não pela existência de um procedimento de enumeração. Conseqüentemente, a busca por formas mais eficientes de se enumerar o conjunto de soluções irá conduzir à obtenção de um algoritmo mais eficiente. Analogamente, a limitação da aplicação do algoritmo de enumeração explícita para a solução apenas de problemas NIP encontra-se na impossibilidade de se enumerar eficientemente soluções de problemas MINP sem a solução do problema por relaxação das variáveis. Este fato é decorrente da existência de variáveis contínuas que definem um número infinito de soluções factíveis.

10.2.1 Introdução ao Método *Branch-and-Bound*

A enumeração implícita de todas as soluções inteiras ou mistas inteiras e contínuas existentes em uma região de interesse de busca pode representar um esforço computacional muito além do que pode ser considerado exeqüível, como já descrito no capítulo 8. Todavia, a avaliação de critérios como a integralidade do problema, a exigência de valores não negativos para as variáveis ou qualquer outra restrição estrutural do problema muitas vezes pode indicar que determinados subconjuntos das variáveis inteiras do problema não podem produzir soluções factíveis. Por conseguinte estes subconjuntos não precisam ser explicitamente avaliados, sendo portanto considerados apenas implicitamente enumerados durante o processo de busca da solução ótima do problema. A utilização destes critérios

de enumeração explícita e implícita de todos os subconjuntos existentes em uma determinada região de interesse de busca define uma classe de algoritmos para a solução de problemas discretos e mistos discretos e contínuos, na qual o método mais conhecido é denominado por *Branch-and-Bound*.

O método *Branch-and-Bound* foi primeiramente apresentado no início da década de 1960 para a solução de problemas lineares discretos ou misto discretos e contínuos (Land & Doig 1960), sendo em seguida apresentada uma bem sucedida implementação computacional para a solução do problema do caixeiro viajante (Little et al. 1963). Já em 1965 foi apresentada uma variação do método para a solução de problemas binários lineares através de testes simples para a obtenção de limites duais e verificação da factibilidade do problema primal (Balas 1965). Neste mesmo ano surge o algoritmo baseado na ramificação de duas vias (Dakin 1965) o qual acabaria por se tornar a forma de codificação mais utilizada comercialmente para a solução de problemas lineares discretos (Wolsey n.d.). Atualmente o método *Branch-and-Bound* é um dos algoritmos mais bem sucedidos para a solução de problemas discretos ou mistos discretos e contínuos, não apenas pela sua eficiência para a solução destes problemas, mas também pela existência de inúmeras variações deste algoritmo. Cada uma destas variações adota uma estratégia distinta para tirar proveito de certas particularidades dos problemas que se destinam a resolver (Linderoth & Savelsbergh 1997).

O principal objetivo de um algoritmo geral *Branch-and-Bound* é realizar a enumeração de todo o conjunto de soluções inteiras ou mistas inteiras e contínuas sem realizar necessariamente a avaliação explícita das mesmas. O elemento chave para a realização deste processo de enumeração é o grupamento formado por um nó e os ramos que dele derivam. De um nó podem derivar dois ou mais ramos, de acordo com as características do problema, ou seja, se as variáveis não contínuas são binárias ou inteiras, e com a implementação do algoritmo. Um conjunto de nós e ramos formará uma árvore que descreverá todas as soluções possíveis para o problema. Um exemplo de representação em árvore para um problema com variáveis binárias é apresentado na figura 10.2.

O único nó existente no nível zero é chamado de nó raiz. No nível 1 existem dois nós que ramificam do nó raiz, um para o valor de $x_1 = 0$ e outro para o valor de $x_1 = 1$. O nó raiz é o nó pai dos dois nós do nível 1, sendo que cada um destes representa dois subproblemas candidatos. No nível 2 existem quatro nós baseados no valor das variáveis x_1 e x_2, onde pode-se observar que existem dois nós pais no nível 1. Repetindo-se o mesmo processo de ramificação obtém-se os oito problemas candidatos para o nível três.

A idéia básica de um algoritmo *Branch-and-Bound* consiste em primeiro resolver o problema contínuo com as variáveis relaxadas. Caso a solução deste problema relaxado

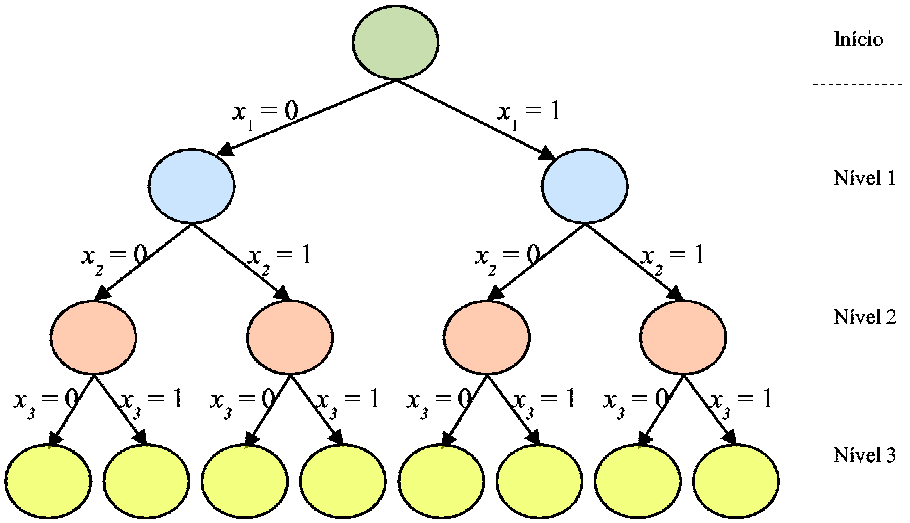


Figura 10.2: Representação em árvore para um problema binário.

não resulte na obtenção da solução inteira, ou mista inteira contínua, deve-se subdividir o problema inicial em diversos subproblemas com a intenção de que a solução destes problemas menores leve à obtenção da solução ótima. As ramificações do nó raiz que conduzem aos subproblemas representam as entradas que podem ser entendidas como uma lista de subproblemas candidatos. Dentre os subproblemas do nível um, um deve então ser selecionado e solucionado a partir da relaxação das variáveis não fixadas no nível anterior. Caso a solução relaxada deste subproblema possua todas as variáveis $\mathbf{x}(1:z) \in \mathbb{I}^z$, deve-se retornar ao nível anterior e selecionar outro subproblema candidato para a solução. Caso contrário, deve-se dividir o subproblema corrente em subproblemas menores. Um subproblema do nível dois deve então ser selecionado e solucionado por relaxação. Cada vez que uma solução que possua todas as variáveis $\mathbf{x}(1:z) \in \mathbb{I}^z$ é encontrada deve-se comparar esta solução como a melhor solução já encontrada em que todas as variáveis $\mathbf{x}(1:z) \in \mathbb{I}^z$. A melhor destas soluções deve ser armazenada, pois representa a solução candidata a ótima do problema. Estes procedimentos devem ser repetidos até que todos os subproblemas que podem ser gerados sejam examinados. Como pode-se concluir, a finitude deste procedimento depende necessariamente da existência de um conjunto finito de soluções para o problema original.

De modo a evitar a enumeração explícita de todas as soluções, pode-se adotar os seguintes critérios de interrupção do processo de ramificação. Este processo de interrupção é comumente conhecido pelo termo em inglês *Fathoming.*:

Critério 1: Como visto no Corolário 6 (página 158), caso a solução do subproblema com variáveis relaxadas seja infactível, tem-se que o problema discreto, ou misto

discreto e contínuo, também é infactível. Conseqüentemente as ramificações deste subproblema podem ser excluídas do processo de enumeração.

Critério 2: Como visto no Teorema 7 (página 157), caso a solução do subproblema com variáveis relaxadas de um determinado nó possua valor superior ou igual ao valor da melhor solução já encontrada, tem-se que todas as soluções de subproblemas ramificados do nó em questão sempre apresentarão soluções com valores superiores ou iguais ao da melhor solução já encontrada. Logo as ramificações deste subproblema podem ser excluídas do processo de enumeração.

Critério 3: Como visto no Teorema 7 (página 157), caso a solução de um subproblema com variáveis relaxadas seja também a solução do problema discreto, ou misto discreto e contínuo, tem-se que todos os subproblemas derivados do nó em questão terão soluções superiores ou iguais ao valor da solução com variáveis relaxadas. Por conseguinte, as ramificações deste subproblema podem ser excluídas do processo de enumeração.

A figura 10.3 apresenta um fluxograma representativo da implementação do algoritmo *Branch-and-Bound* proposto em (Land & Doig 1960). Deve-se observar que em cada ramo podem ser encontradas múltiplas ramificações. Isto ocorre devido ao fato de que a variável selecionada para ser fixada em um determinado nível deve assumir todos os valores discretos dentro de uma faixa definida pelos limites superiores e inferiores, e cada valor discreto representará um ramo para um subproblema candidato. Já a figura 10.4 apresenta um fluxograma representativo da implementação do algoritmo *Branch-and-Bound* proposto em (Dakin 1965). Neste algoritmo cada nó apresenta apenas duas ramificações uma vez que, ao invés de fixar o valor de uma variável em um determinado número discreto, em cada nó adiciona-se uma restrição de desigualdade correspondente ao arredondamento para baixo do valor da variável selecionada ($\mathbf{x}(i) \leq \lfloor w \rfloor$) e uma restrição de desigualdade correspondente ao arredondamento para cima do valor da variável selecionada ($\mathbf{x}(i) \geq \lceil w \rceil$).

10.2.2 Enumeração por *Branch-and-Bound*

Uma vez que o algoritmo *Branch-and-Bound* realiza a enumeração do conjunto de soluções mistas discretas e contínuas de forma mais eficiente, devido aos critérios de interrupção do processo de ramificação, pode-se utilizar este mesmo mecanismo de enumeração *Branch-and-Bound* para construir uma variação do algoritmo *Elipsoidal* capaz de solucionar problemas NIP ou MINP de forma mais eficiente.

A figura 10.5 exibe um fluxograma do processo de busca de um ponto factível pelo algoritmo *Branch-and-Bound* proposto por Dakin. Os objetos sombreados enfatizam as principais diferenças entre o algoritmo *Branch-and-Bound* e o algoritmo de enumeração. A busca inicia-se com o problema original $P_{i=0,j=0}$. Uma implementação similar pode ser obtida para o algoritmo de *Branch-and-Bound* proposto por Land e Doig. A cada nova restrição $\mathbf{x}(k) \leq \lfloor \mathbf{x}^l b \rfloor$ para $1 \leq k \leq z$ a variável i é incrementada e um novo problema $P_{i,j=0}$ é criado. A cada nova restrição $\mathbf{x} \geq \langle \mathbf{x} \rangle$ a variável j é definida como 1 e um novo problema $P_{i,j=1}$ é criado.

Comparando-se a figura 10.5 com a figura 10.4, a qual representa o fluxograma para a implementação do algoritmo *Branch-and-Bound* utilizado com base para a definição do algoritmo de enumeração, também percebe-se que todos os testes comparativos com a solução de limite superior foram retirados. Isto ocorre porque a idéia da enumeração é encontrar apenas uma solução factível para o problema.

O mecanismo principal que se encontra por trás do método de enumeração por *Branch-and-Bound* é enumerar o subconjunto dos pontos contidos em $L_{f \leq f(\mathbf{x}_k)}$ através de um processo de ramificação e exclusão de ramos pelos limites LIC e LSC. Este passo da enumeração garantirá que exista ao menos uma solução $\mathbf{x}(1:z) \in \mathbb{I}^z$ no semi-elipsóide que será utilizado para construir o novo elipsóide E_{k+1}. Um fluxograma geral para um método *Elipsoidal* com enumeração por *Branch-and-Bound* é exibido na figura 10.6. Os objetos sombreados enfatizam as principais diferenças entre o método *Elipsoidal* de Shor e o proposto método *Elipsoidal* com enumeração por *Branch-and-Bound*. A convergência global para o método proposto é garantida para as mesmas condições em que ambos os métodos *Elipsoidal* e *Branch-and-Bound* convergem.

Embora o método *Elipsoidal* seja adequado para a solução de problemas quasi-convexos e não necessariamente diferenciáveis, de modo geral, algoritmos derivados do método *Elipsoidal* não lidam com restrições de igualdade de forma eficiente. Restrições de igualdade usualmente implicam na degeneração do elipsóide ao longo de uma dimensão enquanto as demais não são contraídas o suficiente para permitir que o centro do elipsóide seja um ponto representativo da solução do problema. Certamente existem alternativas para abordar esta característica, tal como construir um algoritmo que avalie permanentemente a relação entre os valores dos autovalores do elipsóide de modo a impedir a degeneração prematura do mesmo.

Dado que a enumeração por *Branch-and-Bound* inclui etapas nas quais serão criadas restrições de desigualdade lineares, é necessária a implementação de uma variação do algoritmo *Elipsoidal* que solucionará o problema contínuo com variáveis relaxadas. Para o algoritmo de busca baseado no *Branch-and-Bound* proposto por Lang e Doig, em cada

nível de ramificação uma variável que deve ser inteira é fixada em um valor resultante da operação de arredondamento. Estes valores pré-fixados constituirão as restrições aqui denominadas de igualdades primárias do problema. Já para algoritmo de busca baseado no algoritmo *Branch-and-Bound* proposto por Dakin, embora não haja fixação explícita de uma variável em valor discreto, as restrições de igualdade primárias do problema aparecerão na forma de pares $\mathbf{x}(i) \geq \sigma$ and $\mathbf{x}(i) \leq \sigma$. Aqui, σ representa um valor discreto resultante do procedimento de arredondamento.

Outra restrição de igualdade, aqui denominada como restrição de igualdade secundária, pode ocorrer quando a fixação de uma variável em um valor específico impõe uma restrição indireta de um valor a outra variável. Um exemplo é a restrição $\mathbf{x}(a)+\mathbf{x}(b) \leq 0$, onde $\mathbf{x}(a)$ e $\mathbf{x}(b) \in \mathbb{R}_+$. Após a ocorrência da restrição de igualdade primária $x(a) = 0$, um restrição de igualdade secundária $\mathbf{x}(1) = 0$ acontece, uma vez que $(0 + \mathbf{x}(a)) \leq 0$ e $\mathbf{x}(b) \geq 0$. A existência de desigualdades primárias e secundárias deve ser verificada a cada acréscimo de desigualdades na implementação da busca baseada no algoritmo *Branch-and-Bound*, antes da execução do passo de solucionar o problema relaxado.

10.2.3 Algoritmos *MEDR*, *BB* e *MEEBB*

Para permitir o tratamento das restrições de igualdade primárias e secundárias, resultantes do processo de enumeração por *Branch-and-Bound*, foi criada uma variação do algoritmo *Elipsoidal* para a solução do problema obtido por relaxação que utilizará uma dimensão reduzida do problema, *MEDR*. Após a identificação das restrições de igualdade existentes, é definido um conjunto de variáveis $\gamma \subset \{1, \ldots, n\}$ cujos valores não estão fixos por restrições de igualdade. Este conjunto será utilizado como parâmetro de entrada do algoritmo *MEDR*. Durante os passos de otimização, o algoritmo *MEDR* calculará os vetores pertencentes ao conjunto subdiferencial e os projetará em um espaço de dimensão reduzida. Neste espaço de dimensão reduzida será definido e contraído o elipsóide até que a solução ótima projetada do problema relaxado seja encontrada. Os valores nos quais as variáveis não livres estão fixadas serão correspondentes aos valores que definem o ponto inicial $\mathbf{x}_0$, o qual é um parâmetro do algoritmo.

O algoritmo 13, mostrado a seguir, implementa a variação do método *MEDR* para o cálculo de um problema relaxado contínuo com dimensão reduzida. Como pode ser facilmente verificado, a mesma idéia de se reduzir a dimensão do problema pode ser implementada em um algoritmo *HISPE* para criar uma versão do algoritmo *MEDR* que considera valores já calculados para o cálculo do novo elipsóide E_{k+1}.

O algoritmo 14 representa a codificação do método *Branch-and-Bound* apresentada na figura 10.5 e baseado na proposição de Dakin. Um algoritmo similar pode ser desenvolvido

com base na codificação do método *Branch-and-Bound* proposto por Land e Doig. Ambos os algoritmos podem ser utilizados na busca de um ponto discreto factível durante o processo de enumeração. A estrutura de regras para a identificação das restrições de igualdade secundárias é definida pela matriz $\mathbf{S} \in \mathbb{R}^{nc,4}$, em que nc representa o número de regras existentes. Para cada regra i as colunas da matriz $\mathbf{S}$ representam:

$\mathbf{S}(i,1)$ **:** índice primário da variável de $\mathbf{x}$;

$\mathbf{S}(i,2)$ **:** valor da variável primária $\mathbf{x}(\mathbf{S}(i,1))$;

$\mathbf{S}(i,3)$ **:** índice secundário da variável de $\mathbf{x}$;

$\mathbf{S}(i,4)$ **:** valor da variável secundária $\mathbf{x}(\mathbf{S}(i,3))$;

A regra da restrição de igualdade secundária é definida por (10.2), bem como é definido o significado da variável booleana $\mathbf{S}^i$.

$$\begin{aligned} \mathbf{S}^i &\triangleq \mathbf{x}(\mathbf{S}(i,1)) == \mathbf{S}(i,2) \\ regra\ i &\triangleq\ se\ \mathbf{S}^i\ defina\ \mathbf{x}(\mathbf{S}(i,3)) \leftarrow \mathbf{S}(i,4) \end{aligned} \tag{10.2}$$

Para o algoritmo 14, define-se o problema $P_{i,j}$ como constituído das restrições $g(\cdot)$, o conjunto de variáveis livres γ, a atual solução $\mathbf{x}^{lb}$ obtida por relaxação, e o índice selecionado k da variável $\mathbf{x}$ utilizado para criar a nova restrição. Adicionar um problema à lista, significa guardar as variáveis que constituem este problema. Excluí-lo da lista, significa apagar as variáveis. Para facilitar a representação do algoritmo, define-se que $f(\emptyset) = \infty_+$. Como este algoritmo pode ser utilizado tanto como enumerador de um ponto factível, quanto como um algoritmo *Branch-and-Bound* convencional, a variável booleana *PareBusca* foi incluída como parâmetro. Esta variável deverá ser definida como *True* no caso em que a execução deva ser interrompida após encontrado o primeiro ponto factível.

Por fim, o algoritmo 15 apresenta a proposição da variação do método *Elipsoidal* com a enumeração de um ponto discreto ou misto contínuo e discreto pelo algoritmo *Branch-and-Bound*. Este algoritmo corresponde ao fluxograma geral apresentado na figura 10.6. Deve-se salientar que a função *BuscaBB* executa uma função *BB* correspondente ao algoritmo *Branch-and-Bound*. Em função da codificação utilizada BB^{LD}, baseado na proposição de Land e Doig, ou BB^{Da}, baseado na proposição de Dakin, teremos duas variações distintas para o algoritmo proposto $MEEBB$.

Novamente, uma variação do algoritmo $HiSCPE$ pode ser desenvolvida sob os mesmos critérios de busca de um ponto discreto ou misto discreto e contínuo. Assim ter-se-á uma

versão do algoritmo $MEEBB$ que considera valores já calculados para o cálculo do novo elipsóide E_{k+1}.

Uma comparação entre os algoritmos algoritmo 15 e algoritmo 12 demonstra as semelhanças existentes. Já a principal diferença está no fato do algoritmo $MEEBB$ ser capaz de solucionar problemas MINP, em função do processo de busca utilizado. Todavia esta diferença é preponderante para permitir a solução de problemas NIP.

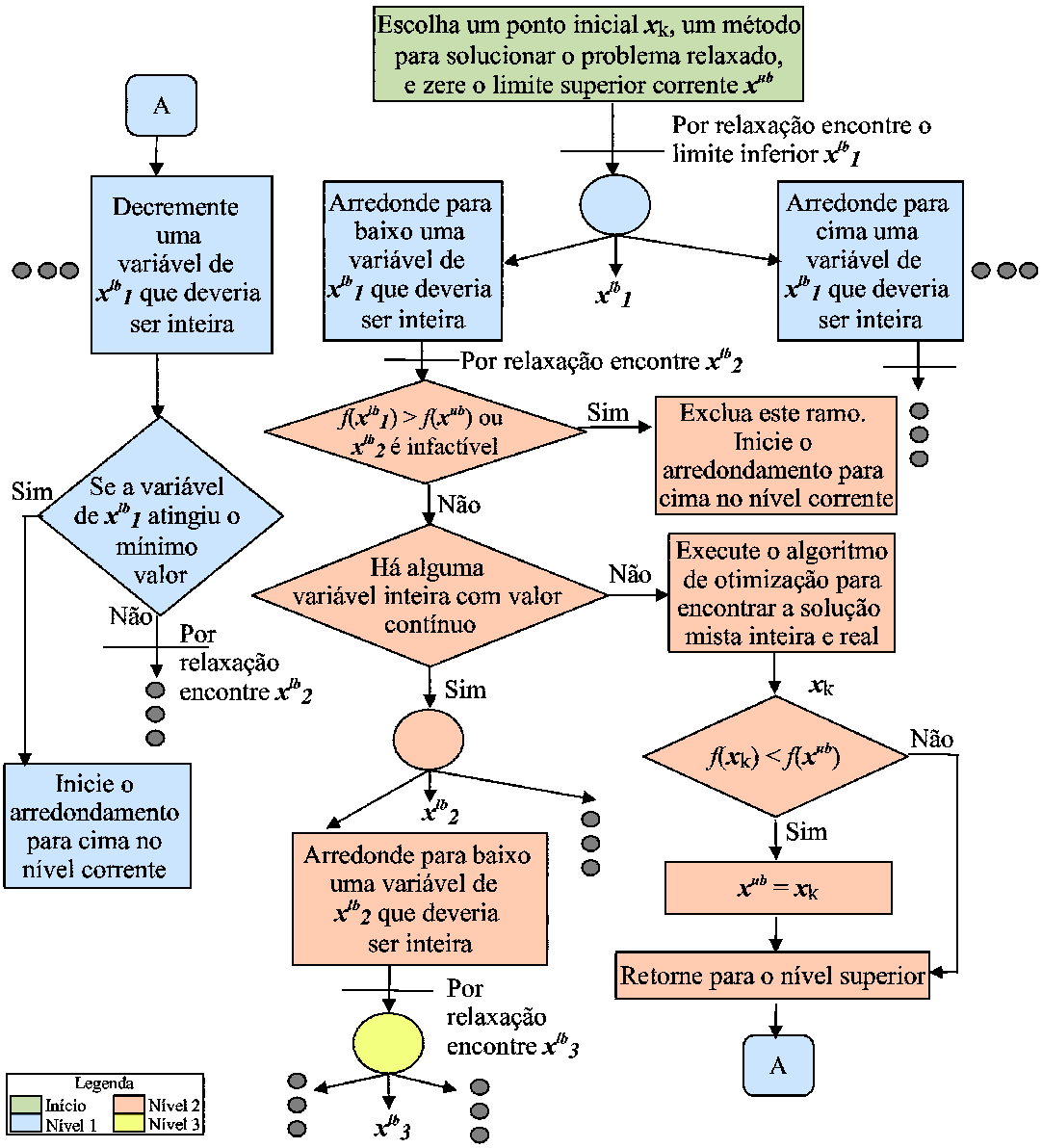


Figura 10.3: Fluxograma representativo do algoritmo *Branch-and-Bound* proposto por Land e Doig.

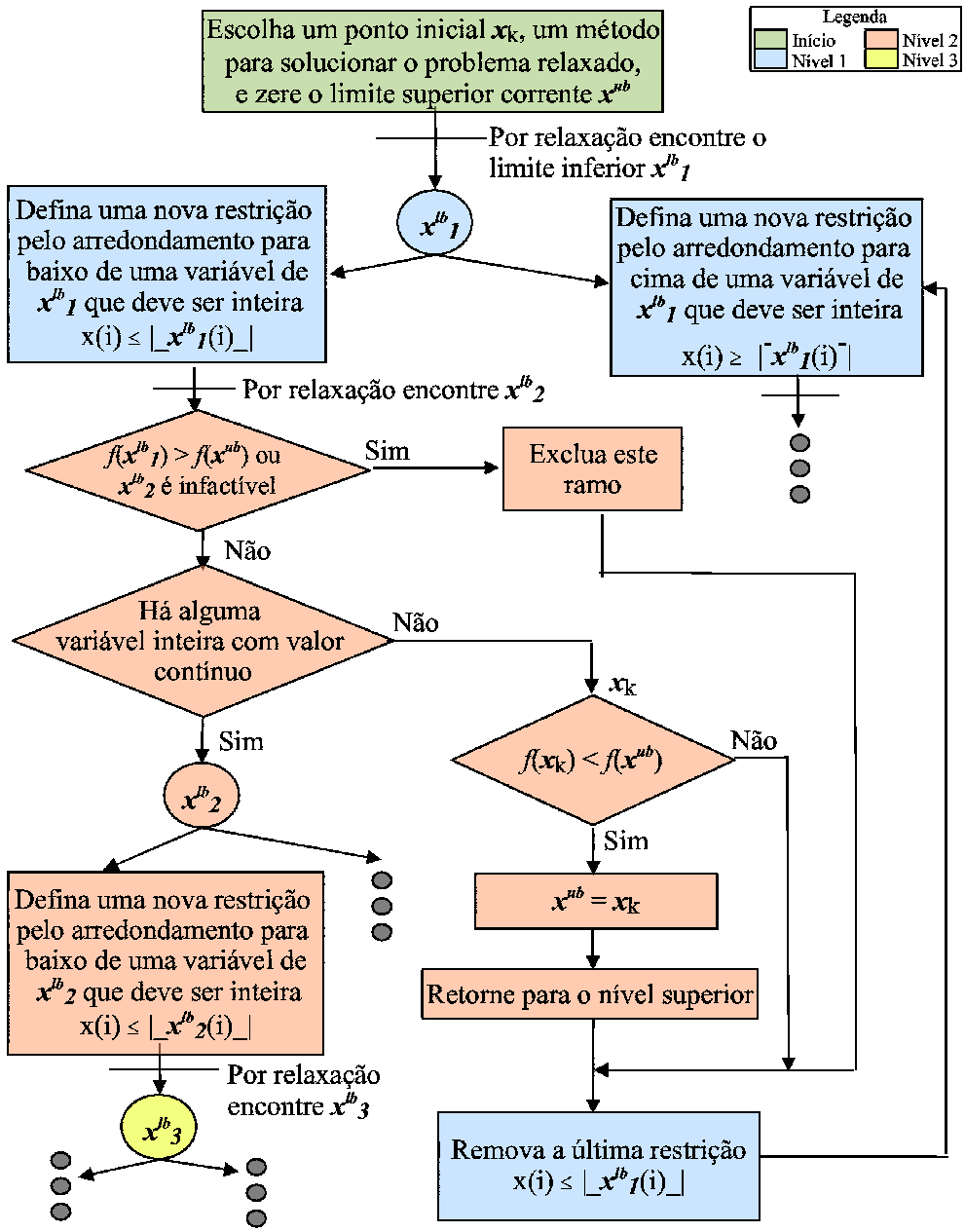


Figura 10.4: Fluxograma representativo do algoritmo *Branch-and-Bound* proposto por Dankin.

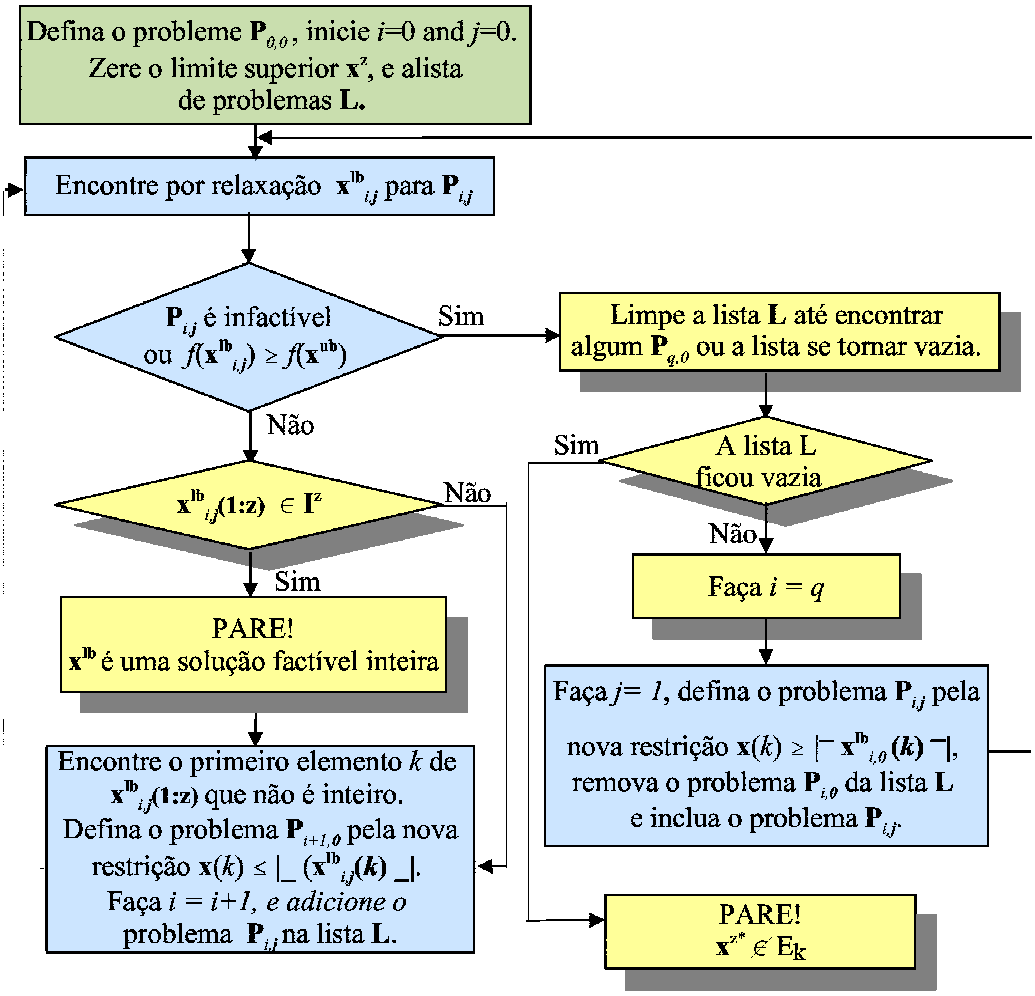


Figura 10.5: Fluxograma do algoritmo de enumeração por *Branch-and-Bound* para a proposição de Dakin.

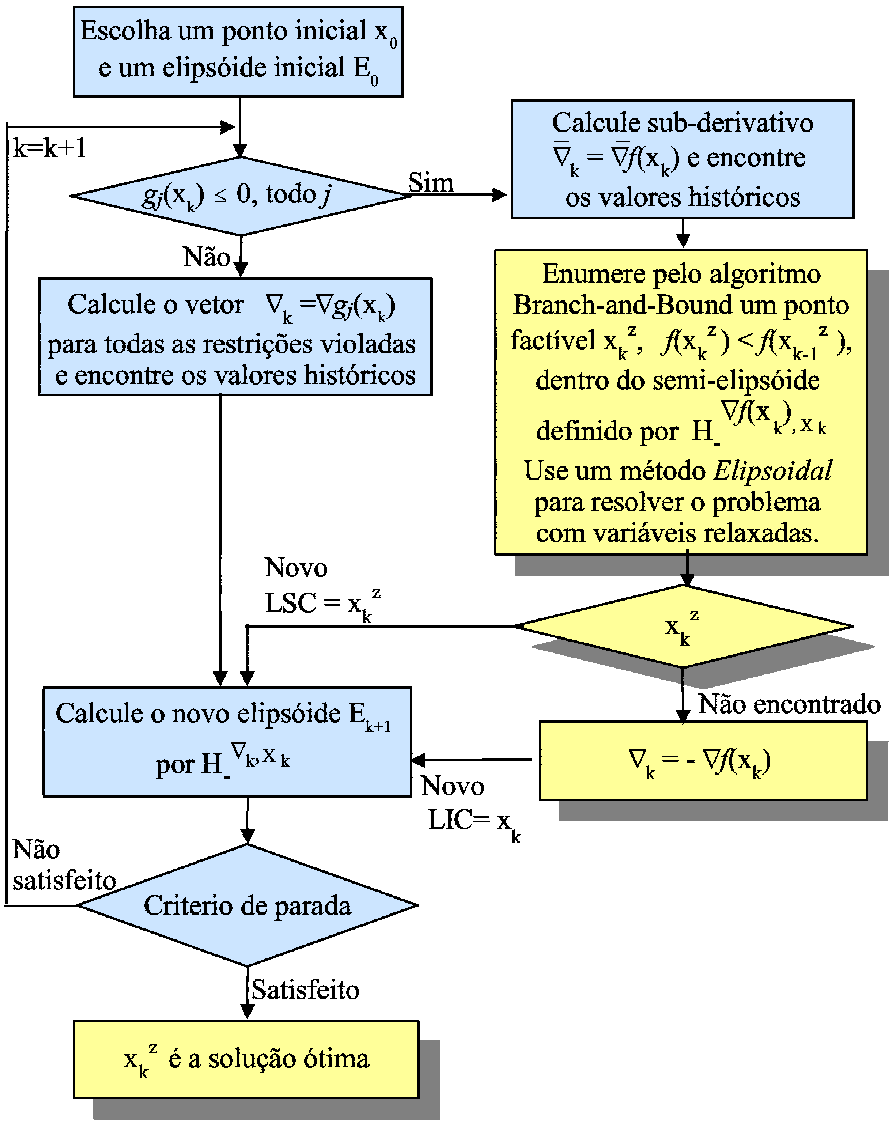


Figura 10.6: Fluxograma geral para um método *Elipsoidal* com enumeração por *Branch-and-Bound*.

Algorithm 13 *Elipsoidal* com Dimensão Reduzida

Input: Um elipsóide inicial E_0, onde a solução contínua ótima $\mathbf{x}^*$ será procurada de acordo com o problema definido por (10.1). O conjunto de variáveis livres e γ com tamanho q.

Output: A melhor solução factível $\mathbf{x}^f$ encontrada. Para um problema infactível ou para $\mathbf{x}^* \notin E_0$ ter-se-á $\mathbf{x}^f = \emptyset$.

1: **function** $MEDR(E_0, \gamma)$
2: $k \leftarrow 0$
3: $\mathbf{x}^f \leftarrow \emptyset$ ▷ Zera a melhor solução factível.
4: $\hat{\mathbf{x}}_k \leftarrow \mathbf{x}_k(\gamma)$ ▷ $\hat{\mathbf{x}}_k \in \mathbb{R}^q$
5: $\hat{Q}_k \leftarrow Q_k(\gamma, \gamma)$ ▷ $\hat{Q}_k \in \mathbb{R}^{q \times q}$
6: **loop** ▷ Laço principal do algoritmo.
7: **if** $g_i(\mathbf{x}_k) \leq 0 \; \forall \; i = 1, \ldots, p$ **then**
8: $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{x}_k)$
9: $\mathbf{x}^f \leftarrow \mathbf{x}_k$
10: **if** $\bar{\nabla}_k = 0$ **then**
11: **return** $\mathbf{x}^f$ ▷ Termine o algoritmo.
12: **end if**
13: **else**
14: $\bar{\nabla}_k \leftarrow \bar{\nabla} g_i(\mathbf{x}_k)$ *para algum* $g_i(\mathbf{x}_k) > 0, \mid i = 1, \ldots, p$
15: **end if**
16: $\hat{\bar{\nabla}}_k \leftarrow \bar{\nabla}_k(\gamma)$ ▷ $\hat{\bar{\nabla}}_k \in \mathbb{R}^q$
17: $\hat{E}_{k+1} \leftarrow (\hat{E}_k \cap H_-^{\hat{\bar{\nabla}}_k, \hat{\mathbf{x}}_k})$ ▷ Veja (2.11).
18: $\mathbf{x}_{k+1} \leftarrow \{w_1, \ldots, w_n\} \mid \mathbf{w}(\gamma) = \hat{\mathbf{x}}_{k+1} \mid w_i = \mathbf{x}_0(i) \; \forall \; i \notin \gamma$
19: **if** $vol(\hat{E}_{k+1}) \simeq 0$ ou $vol(\hat{E}_{k+1}) \simeq vol(E_k)$ **then**
20: **if** $g_i(x_{k+1}) \leq 0 \; \forall \; i = 1, \ldots, p$ **then**
21: $\mathbf{x}^f \leftarrow \mathbf{x}_{k+1}$
22: **end if**
23: **return** $\mathbf{x}^f$ ▷ Termine o algoritmo.
24: **end if**
25: $k \leftarrow k + 1$
26: **end loop**
27: **end function**

Algorithm 14 *Branch-and-Bound* sobre proposição de Dakin

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^{z*}$ será procurada de acordo com o problema definido por (10.1). A estrutura de restrições de igualdade secundárias $\mathbf{S}$. A variável *PareBusca*.

Output: A melhor solução factível $\mathbf{x}^z$ encontrada. Para um problema infactível ter-se-á $\mathbf{x}^z = \emptyset$.

1: **function** $BB^{Da}(E_0, \mathbf{S}, PareBusca)$
2: $i \leftarrow 0, j \leftarrow 0$
3: $\mathbf{x^z} \leftarrow \emptyset, \quad \mathbf{x}^{lb}_{i,j} \leftarrow \emptyset$ ▷ Zera o melhor ponto factível e os pontos dos LIC.
4: $\gamma \leftarrow \{1, \ldots, n\}, \quad \bar{\gamma} \leftarrow \emptyset$ ▷ Zera o conjunto de variáveis livres e forçadas.
5: $lP \leftarrow \emptyset$ ▷ Lista de problemas.
6: **loop** ▷ Laço principal do branch-and-bound.
7: **release** todas as regras de restrições de igualdade secundárias
8: **if** $\exists$ *algum* k *e* $\sigma \mid \mathbf{x}^{lb}_{i,j}(k) \leq \sigma$ *e* $\mathbf{x}^{lb}_{i,j}(k) \geq \sigma$ **then**
9: $\bar{\gamma} \leftarrow$ *todos os* $k, \quad \gamma \leftarrow \{l \mid l \in \{1, \ldots, n\} \mid l \notin \bar{\gamma}\}$
10: **end if**
11: **apply** $regra^l \; \forall \; \mathbf{S}^l == True$ ▷ Restrições de igualdade secundárias.
12: $\bar{\gamma} \leftarrow (\bar{\gamma} \cup l) \mid \forall \; \mathbf{S}^l == True, \quad \gamma \leftarrow \{l \mid l \in \{1, \ldots, n\} \mid l \notin \bar{\gamma}\}$
13: $\tilde{\mathbf{x}}_0(\bar{\gamma}) \leftarrow \mathbf{x}^{lb}_{i,j}(\bar{\gamma})$ ▷ Copie os valores fixos para o ponto que é o centro do elipsóide.
14: $\mathbf{x}^{lb}_{i,j} \leftarrow MEDR(E_0, \gamma)$ ▷ Resolução do problema relaxado algoritmo 13.
15: **if** $\mathbf{x}^{lb}_{i,j} = \emptyset$ ou $f(\mathbf{x}^{lb}_{i,j}) > f(\mathbf{x^z})$ **then**
16: **go to** linha 32
17: **end if**
18: **if** $\mathbf{x}^{lb}_{i,j}(1:z) \in \mathbb{I}^z_+$ **then** ▷ Ponto factível.
19: **if** $f(\mathbf{x}^{lb}_{i,j}) < f(\mathbf{x^z})$ **then** ▷ Melhor solução até o momento.
20: $\mathbf{x^z} \leftarrow \mathbf{x}^{lb}_{i,j}$ ▷ Atualiza LSC.
21: **if** $PareBusca == True$ **then**
22: **return** $\mathbf{x^z}$ ▷ Termine o algoritmo.
23: **end if**
24: **end if**
25: **go to** linha 32
26: **end if**
27: $k \leftarrow qualquer(\{l \mid l \in \{1, \ldots, z\} \mid \mathbf{x}^{lb}_{i,j}(l) \notin \mathbb{I}_+\})$
28: **add** nova restrição $\mathbf{x}(k) \leq \lfloor \mathbf{x}^{lb}_{i,j}(k) \rfloor$
29: $i \leftarrow i + 1, \quad j \leftarrow 0$
30: **add** problema $P_{i,j}$ na lista lP
31: **go to** linha 07
32: **loop while** $j = 1$
33: **restore** da lista lP o último problema e **remove** este problema da lista lP
34: **end loop**
35: **if** $lP = \emptyset$ **then**
36: **return** $\mathbf{x^z}$ ▷ Termine o algoritmo.
37: **end if**
38: **remove** última restrição $\mathbf{x}(k) \leq \lfloor \mathbf{x}^{lb}_{i,j}(k) \rfloor$
39: **add** nova restrição $\mathbf{x}(k) \geq \lceil \mathbf{x}^{lb}_{i,j}(k) \rceil$
40: **remove** problema $P_{i,0}$ da lista lP
41: $j \leftarrow 1$
42: **add** problema $P_{i,j}$ na lista lP
43: **end loop**
44: **end function**

Algorithm 15 *Elipsoidal* com Enumeração por Branch-and-Bound

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^{z*}$ será procurada de acordo com o problema definido por (10.1). A estrutura de restrições de igualdade secundárias $\mathbf{S}$. Um valor para precisão para o teste de limites $\hat{\epsilon} > 0$.

Output: A melhor solução factível $\mathbf{x}^z$ encontrada. Para um problema infactível ou para $\mathbf{x}^{z*} \notin E_0$ ter-se-á $\mathbf{x}^z = \emptyset$.

function $MEEBB(E_0, \mathbf{S}, \hat{\epsilon})$
  $k \leftarrow 0$
  $\mathbf{x}^z \leftarrow \emptyset$ ▷ Zera o melhor ponto factível
  loop ▷ Laço de movimentação do LSC.
    if $g_i(\mathbf{x}_k) \leq 0 \; \forall \; i = 1, \ldots, p$ **then**
      $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{x}_k)$
      if $\mathbf{x}_k \notin \Omega$ e $f(\mathbf{x}_k) < f(\mathbf{x}^z)$ **then**
        $\mathbf{w}^z \leftarrow BuscaBB(E_k, \bar{\nabla}_k, \mathbf{x}^z, \mathbf{S})$ ▷ Procura um ponto factível.
        if $\mathbf{w}^z = \emptyset$ **then**
          if $((f(\mathbf{x}^z) - f(\mathbf{x}_k)) \leq \hat{\epsilon})$ ou $(\frac{\bar{\nabla}_k^T}{|\bar{\nabla}_k|} * (\mathbf{x}^z - \mathbf{x}_k) \leq \hat{\epsilon})$ **then**
            return $\mathbf{x}^z$ ▷ Termine o algoritmo.
          end if
          $\bar{\nabla}_k \leftarrow -\bar{\nabla}_k$ ▷ $\mathbf{x}_k$ é o novo LIC
          $\mathbf{w}^z \leftarrow BuscaBB(E_k, \bar{\nabla}_k, \mathbf{x}^z, \mathbf{S})$ ▷ Procura um ponto factível.
          if $\mathbf{w}^z = \emptyset$ **then**
            return $\mathbf{x}^z$ ▷ Termine o algoritmo.
          end if
        end if
      else
        $\mathbf{w}^z \leftarrow \mathbf{x}_k$
      end if
      $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{w}^z)$
      $\mathbf{x}^z \leftarrow \mathbf{w}^z$ ▷ Melhor solução até o momento.
    else
      $\bar{\nabla}_k \leftarrow \bar{\nabla} g_i(\mathbf{x}_k)$ *para algum* $g_i(\mathbf{x}_k) > 0, \mid i = 1, \ldots, p$
      $\mathbf{w}^z \leftarrow \mathbf{x}_k$
    end if
    $E_{k+1} \leftarrow (E_k \cap H_-^{\bar{\nabla}_k, \mathbf{w}^z})$ ▷ Vide (2.14).
    if vol $(E_{k+1}) \simeq 0$ ou $vol(E_{k+1}) \simeq vol(E_k)$ **then**
      return $\mathbf{x}^z$ ▷ Termine o algoritmo.
    end if
    $k \leftarrow k + 1$
  end loop
end function

function $BuscaBB(E_k, \bar{\nabla}_k, \mathbf{x}^z, \mathbf{S})$ ▷ Esta função corresponde à busca de uma melhor solução que $\mathbf{x}^z$, por *Branch-and-Bound* no semi-elipsóide definido por $E_k \cap H_-^{\bar{\nabla}_k, \mathbf{x}_k}$.
  add restrição $(\bar{\nabla}_k)^T(\mathbf{x} - \mathbf{x}_k) \leq 0$
  if $\mathbf{x}^z \neq \emptyset$ **then**
    add restrição $(\bar{\nabla} f(\mathbf{x}^z))^T(\mathbf{x} - \mathbf{x}^z) \leq 0$
  end if
  $[\mathbf{x^z}, E_k] \leftarrow BC(E_k, \mathbf{S}, True)$ ▷ Algoritmo 16
  release restrições adicionadas.
  return o primeiro ponto factível $\mathbf{x}^z$.
end function

10.3 Variações do Método *Elipsoidal* com *Branch-and-Cut*

10.3.1 Introdução ao *Branch-and-Cut*

A necessidade de se solucionar problemas muito difíceis foi a centelha necessária para se aplicar em conjunto diferentes idéias inovadoras durante o processo de solução dos problemas (Wolsey n.d.). Um bom exemplo desta associação está na integração de técnicas de plano de corte aos algoritmos de *Branch-and-Bound*. Para estes problemas difíceis, tal como já discutido na seção 1.2.2, um incremento no esforço de cálculo do algoritmo de otimização (ECAO) não significa necessariamente um incremento no esforço computacional total para a determinação da solução ótima (ECT). Em diversos casos, é exatamente o aumento da complexidade do algoritmo de otimização que permitirá a redução do ECT.

O primeiro código mais genérico que adotou a utilização de planos de corte efetivos para serem utilizados no nó raiz de um algoritmo *Branch-and-Bound* para a solução de problemas BIP de larga escala pode ser encontrado em (Crowder et al. 1983). Esta abordagem pode ser denominada mais precisamente de um método *Cut-and-Branch*. Já a idéia de *Branch-and-Cut*, a qual determina a geração de planos de corte durante o processo de ramificação, pode ser identificada em (Savelsbergh & Nemhauser 1993) e posteriormente em (Gu et al. 1995). Um número cada vez maior de algoritmos que combinam técnicas de geração de hiperplanos de corte a *Branch-and-Bound* podem ser encontrados em publicações recentes (Wolsey n.d., Cordier et al. 1999, Padberg 2005).

Embora a geração de planos de corte durante o processo de ramificação de uma algoritmo *Branch-and-Bound* possa parecer ser apenas um diferencial mínimo, na prática esta geração representa uma mudança de filosofia. O que se pretende é, não apenas avaliar os nodos de forma rápida, mas sim gerar o trabalho que for necessário para garantir a aproximação dos limites LIC e LSC existentes a cada processo de ramificação.

10.3.2 Fundamentos de *Branch-and-Cut* Aplicados ao Método *Elipsoidal*

Uma idéia análoga à *Branch-and-Cut* pode ser construída quando um método baseado em *Elipsóide* é utilizado como ferramenta de solução do problema obtido por relaxação em um algoritmo *Branch-and-Bound*. Primeiramente deve-se observar que um método *Elipsoidal* precisa de um elipsóide inicial para restringir a região de busca. Então, para o método de *Branch-and-Cut* aqui proposto, a idéia primordial será utilizar os limites encontrados durante o processo de ramificação da enumeração para cortar (comprimir) o

elipsóide que será utilizado durante o próximo processo de busca da solução relaxada. Por fim, tal como pretendido por um algoritmo *Branch-and-Cut*, este processo comprimirá os limites LIC e LSC um contra o outro.

Um corte eficiente (i.e., do inglês *strong cut*) é aqui definido como o semi-espaço que contém uma solução inteira, ou mista inteira e contínua, e que intercepta o corrente elipsóide de modo que o novo elipsóide com volume menor possa ser criado a partir desta intersecção. O ponto preponderante do método *Branch-and-Cut* é como definir cortes que poderão ser utilizados na compressão do elipsóide corrente. O Teorema 8 e o Lema 8 formalizam os cortes que podem ser utilizados juntamente ao algoritmo *Branch-and-Bound* como cortes eficientes.

Teorema 8 - *Critério de Corte para o Elipsóide: Considere o problema descrito por (10.1) e o elipsóide* $\{\check{E} \subset \mathbb{R}^n \mid \mathbf{x}^{z*} \in \check{E}\}$ *, definido pelo centro* $\check{\mathbf{x}}$ *e pela matriz* $\check{Q}^{-1}$*. Defina a função* $\rho(\xi, \mathbf{x}) = \frac{\xi^T(\check{\mathbf{x}}-\mathbf{x})}{\sqrt{\xi^T\check{Q}^{-1}\xi}}$ *na qual* ξ *e* $\mathbf{x} \in \mathbb{R}^n$ *. Suponha o ponto* $\{\mathbf{w} \mid \mathbf{w}(1:z) \in \mathbb{I}_+^z \mid \mathbf{w}(z+1:n) \in \mathbb{R}^{n-z}\}$*, então*

i) $\{H_+^{\bar{\nabla}f(\mathbf{w}),\mathbf{w}} \mid z = 0 \mid \mathbf{w} = \mathbf{x}^*\}$ *é um corte eficiente para o elipsóide* $\check{E}$ *se* $(-\frac{1}{n} < \rho(-\bar{\nabla}f(\mathbf{w}), \mathbf{w}) \leq 1)$*.*

ii) $\{H_-^{\bar{\nabla}g_i(\mathbf{w}),\mathbf{w}} \mid \exists$ *algum* $g_i(\mathbf{w}) > 0 \; \forall \; i = 1, \ldots, p\}$ *é um corte eficiente para o elipsóide* $\check{E}$ *se* $(-\frac{1}{n} < \rho(\bar{\nabla}g_i(\mathbf{w}), \mathbf{w}) \leq 1)$*.*

iii) $\{H_-^{\bar{\nabla}f(\mathbf{w}),\mathbf{w}} \mid \mathbf{w} \in \Omega\}$ *é um corte eficiente para o elipsóide* $\check{E}$ *se* $(-\frac{1}{n} < \rho(\bar{\nabla}f(\mathbf{w}), \mathbf{w}) \leq 1)$*.*

Prova:

i) *O corte por LIC global,* $H_+^{\bar{\nabla}f(\mathbf{x}^*),\mathbf{x}^*}$*, é garantido pelo Teorema 7 (página 157). Desde que* $(\bar{\nabla}f(\mathbf{x}^*))^T(\mathbf{x}^{z*} - \mathbf{x}^*) < 1$*,* $\mathbf{x}^{z*} \in (H_+^{\bar{\nabla}f(\mathbf{x}^*),\mathbf{x}^*} \cap \check{E})$*. A função* $\rho(-\bar{\nabla}f(\mathbf{w}), \mathbf{w})$ *mede a distância,* $\triangle$*, do ponto mais próximo ao hiperplano* $H^{\bar{\nabla}f(\mathbf{w}),\mathbf{x}}$ *até o centro do elipsóide* $\check{\mathbf{x}}$ *pela métrica do elipsóide. Por fim,* $\triangle \in (-\frac{1}{n}, 1]$ *porque* $\triangle > 1$ *implica que* $\mathbf{x}^{z*} \notin \Omega$*, o que é impossível, e* $\triangle \leq -\frac{1}{n}$ *resulta em um novo elipsóide com volume igual ao volume do elipsóide* $\check{E}$*, o que contradiz a definição de um corte eficiente.*

ii) *Os cortes por pontos infactíveis,* $H_-^{\bar{\nabla}g_i(\mathbf{w}),\mathbf{w}}$*, são garantidos por* $\Omega \subset H_-^{\bar{\nabla}g_i(\mathbf{w}),\mathbf{w}}$ *para todos os* $g_i(\mathbf{w}) > 0$*. Então* $\mathbf{x}^{z*} \in (H_-^{\bar{\nabla}g_i(\mathbf{w}),\mathbf{w}} \cap \check{E})$*. O intervalo* $\triangle \in (-\frac{1}{n}, 1]$ *é definido de forma análoga ao item i.*

iii) *Os cortes das soluções factíveis,* $H_{-}^{\bar{\nabla}f(\mathbf{w}),\check{\mathbf{x}}}$*, são garantidos por* $(\bar{\nabla}f(\mathbf{w}))^T(\mathbf{x}^{z*}-\mathbf{w}) \leq 0$ *para todo* $\mathbf{w} \in \Omega$*. Por conseqüência* $\mathbf{x}^{z*} \in (H_{-}^{\bar{\nabla}f(\mathbf{w}),\check{\mathbf{x}}} \cap \check{E})$*. O intervalo* $\triangle \in (-\frac{1}{n}, 1]$ *é definido de forma análoga ao item i e completa a demonstração do teorema.*

Lema 8 - *Critério de Corte por Curva de Nível: Considere o problema descrito por (10.1) e o elipsóide* $\{\check{E} \subset \mathbb{R}^n \mid \mathbf{x}^{z*} \in \check{E}\}$ *, definido pelo centro* $\check{\mathbf{x}}$ *e pela matriz* $\check{Q}^{-1}$*. Defina a função* $\rho(\xi, \mathbf{x}) = \frac{\xi^T(\check{\mathbf{x}}-\mathbf{x})}{\sqrt{\xi^T\hat{Q}^{-1}\xi}}$ *na qual* ξ *e* $\mathbf{x} \in \mathbb{R}^n$*. Suponha um ponto* $\{\mathbf{w} \in \Omega\}$ *e um ponto* $\{\mathbf{v} \in \mathbb{R}^n \mid f(\mathbf{v}) > f(\mathbf{w})\}$*. O semi-espaço* $H_{-}^{\bar{\nabla}f(\mathbf{v}),\mathbf{v}}$ *é um corte eficiente para o elipsóide* $\check{E}$ *se* $(-\frac{1}{n} < \rho(\nabla)f(\mathbf{v}), \mathbf{v}) \leq 1$*.*

Prova:

Diretamente do Teorema 8, uma vez que $f(\mathbf{x}) - f(\mathbf{w}) \leq 0$ *pode ser considerada um restrição para o problema definido por (10.1) para todo* $\{\mathbf{w} \in \Omega\}$*.*

Concomitantemente ao processo de geração de cortes para o elipsóide, ao ser considerada a redução do volume que ocorrerá durante o processo de ramificação do algoritmo *Branch-and-Cut*, é possível definir um novo critério de interrupção deste processo de ramificação. Este critério possibilitará a exclusão de um ramo no qual nenhum hiperplano definido por vetores paralelos aos eixos das variáveis livres, definidas por γ, intercepta o elipsóide corrente. O Teorema 9 formaliza este critério de interrupção do processo de ramificação.

Teorema 9 - *Critério de Interrupção do Processo de Ramificação Aplicado a um Elipsóide: Considere o problema descrito por (10.1) e o elipsóide* $\{\check{E} \subset \mathbb{R}^n \mid \mathbf{x}^{z*} \in \check{E}\}$ *, definido pelo centro* $\check{\mathbf{x}}$ *e pela matriz* $\check{Q}^{-1}$*. Suponha que o ponto* $\mathbf{x}^{lb} \in \mathbb{R}^n$ *é a solução de um problema obtido por relaxação para a região de busca definida pelo elipsóide* $\check{E}$ *e que* $\{\mathbf{x}^{lb}(k) \in \mathbb{R} \mid \mathbf{x}^{lb}(k) \notin \mathbb{I} \mid k \in \{1, \ldots, z\}$*. Considere a nova restrição definida por uma das equações a seguir:*

i) $g_{p+1} \triangleq \{\mathbf{x}(k) \leq \sigma \mid \sigma = \lfloor \mathbf{x}^{lb}(k) \rfloor\}$.

ii) $g_{p+1} \triangleq \{\mathbf{x}(k) \geq \sigma \mid \sigma = \lceil \mathbf{x}^{lb}(k) \rceil\}$.

iii) $g_{p+1} \triangleq \{\mathbf{x}(k) = \sigma \mid \sigma = \lfloor \mathbf{x}^{lb}(k) \rfloor\}$.

iv) $g_{p+1} \triangleq \{\mathbf{x}(k) = \sigma \mid \sigma = \lceil \mathbf{x}^{lb}(k) \rceil\}$.

Também considere o ponto $\{\mathbf{w} \mid \mathbf{w}(i) = \mathbf{x}^{lb}(i) \; \forall \; (\{i \neq k \; e \; i = 1, \ldots, n\} \mid \mathbf{w}(k) = \sigma\}$ *e o vetor* $\{\xi \mid \xi(i) = 0 \; \forall \; (\{i \neq k \; e \; i = 1, \ldots, n\} \mid \xi(k) = (\mathbf{x}^{lb}(k) - \sigma)\}$*. O novo problema definido por (10.1) adicionado da restrição* g_{p+1} *não possui solução factível* $\mathbf{x}^{z*}$ *se*

$|\frac{\xi^T(\mathbf{w}-\check{\mathbf{x}})}{\sqrt{\xi^T\check{Q}^{-1}\xi}}| > 1$.

Prova:

i) *Como* $\mathbf{x}^{\mathbf{z}*} \in \check{E}$ *então* $\in (\check{E} \cap H_{-}^{\xi,\mathbf{w}}) \neq \emptyset$. *Dado que* $\mathbf{x}^{lb} \notin H_{-}^{\xi,\mathbf{w}}$ *não há um caso em que* $\check{E} \subset H_{-}^{\xi,\mathbf{w}}$. *Por conseguinte* $(\check{E} \cap H^{\xi,\mathbf{w}}) \neq \emptyset$.

ii) *Se existe um hiperplano que intercepta o elipsóide* $\check{E}$, *a distância do ponto mais próximo que pertence ao hiperplano até o centro do elipsóide* $\check{\mathbf{x}}$ *deve pertencer ao intervalo* $[-1, 1]$, *quando a distância é medida pela norma do elipsóide* $\check{E}$. *O valor* 1 *representa o raio unitário, considerando-se que um elipsóide é medido como uma hiperesfera unitária na sua própria norma.*

iii) *Uma vez que* $\frac{\xi^T(\mathbf{w}-\check{\mathbf{x}})}{\sqrt{\xi^T\check{Q}^{-1}\xi}}$ *representa a distância entre o hiperplano* $H^{\xi,\mathbf{w}}$ *na norma do elipsóide* $\check{E}$, *se* $|\frac{\xi^T(\mathbf{w}-\check{\mathbf{x}})}{\sqrt{\xi^T\check{Q}^{-1}\xi}}| > 1$ *então* $(\check{E} \cap H^{\xi,\mathbf{w}}) = \emptyset$ *e* $\mathbf{x}^{z*}$ *não pode ser um resultado para o problema.*

Um fluxograma geral para o algoritmo *Branch-and-Cut* baseado na proposição de *Branch-and-Bound* de Dakin é apresentado na figura 10.7. Uma implementação análoga pode ser construída para a proposição de *Branch-and-Bound* de Lang and Doig. A convergência global é novamente assegurada para as mesmas condições em que os métodos *Branch-and-Bound* e *Elipsoidal* convergem. As principais diferenças entre o fluxograma da figura 10.7 para o fluxograma de *Branch-and-Bound* apresentado na figura 10.5 estão indicadas pelas caixas sombreadas. A estrutura de restrições de igualdade secundárias deve ser definida manualmente pela análise das restrições do problema.

10.3.3 Método *Elipsoidal* com Enumeração por *Branch-and-Cut*: *MEEBC*

Um fluxograma geral para um método *Elipsoidal* com enumeração por *Branch-and-Cut* corresponde quase que exatamente ao fluxograma apresentado para a enumeração por *Branch-and-Bound* exibido na figura 10.6. A diferença fica no fato de que o elipsóide comprimido durante a função de busca *Branch-and-Cut* se sobreporá ao elipsóide corrente E_k. Tal como ocorre para a enumeração baseada *Branch-and-Bound*, a convergência global para o método proposto é garantida para as mesmas condições em que ambos os métodos *Elipsoidal* e *Branch-and-Bound* convergem.

O algoritmo 16 representa a codificação do método *Branch-and-Cut* baseado na proposição de Dakin apresentado na figura 10.7. Este foi desenvolvido a partir do algoritmo 14

acrescida a redução do elipsóide pelos cortes definidos no Teorema 8, no Lema 8 e na nova regra de interrupção do processo de ramificação definida no Teorema 9. Um algoritmo similar pode ser codificado para a proposição de *Branch-and-Bound* de Land e Doig. As mesmas definições descritas para o algoritmo 14, referentes à estrutura de restrições de igualdade secundárias $\mathbf{S}$ e a um problema $P_{i,j}$, são válidas. Como este algoritmo pode ser utilizado tanto como enumerador de um ponto factível, quanto como algoritmo *Branch-and-Cut* que encontra a solução ótima, a variável booleana $ParеBusca$ foi incluída como parâmetro. Esta variável deverá ser definida como $True$ no caso em que a execução deve ser interrompida após encontrado o primeiro ponto factível.

Este algoritmo irá acelerar o processo de busca por um ponto discreto, ou misto contínuo e discreto, na medida em que o elipsóide que representa a região de interesse de busca do problema obtido por relaxação tiver seu volume efetivamente reduzido. Além das idéias discutidas nas seções anteriores, foi incluída no algoritmo uma heurística de busca de uma solução na vizinhança do centro do elipsóide corrente. Esta busca consiste em arredondar todas as variáveis que devem ser inteiras do centro do elipsóide corrente na direção oposta à direção do vetor pertencente ao conjunto subdiferencial avaliado neste mesmo ponto. Esta parece ser uma boa heurística para um problema quando o centro do elipsóide corrente possui todas as restrições $g(\cdot) \leq 0$ e o elipsóide corrente ainda possui um volume de tamanho considerável.

10.3.4 Método *Branch-and-Cut Elipsoidal*: $MBCE$

Como regra geral, os algoritmos baseados em *Elipsóide* dependem da convergência do elipsóide corrente para um elipsóide cujo volume tenda a zero. Embora tenham sido criados dois novos critérios de parada para os algoritmos *Elipsoidais* com enumeração, quando da definição de um novo LIC, os algoritmos até aqui propostos para a solução de problemas NIP e MINP continuam fortemente influenciados por esta regra.

Uma alternativa para este fato, é adotar o algoritmo *Branch-and-Cut* não apenas para procurar um ponto factível, mas para concluir o processo de busca da solução ótima. O método *Branch-and-Cut Elipsoidal*, $MBCE$, aqui proposto parte desta proposição incluindo a adição de um laço inicial de um algoritmo *Elipsoidal* e da heurística da vizinhança de modo a procurar reduzir o elipsóide que será utilizado como elipsóide inicial no algoritmo *Branch-and-Cut*.

Um fluxograma geral para um método $MBCE$ corresponde ao fluxograma apresentado na figura 10.8. Os objetos sombreados enfatizam as principais diferenças entre o método *Elipsoidal* de Shor e o proposto método *Branch-and-Cut Elipsoidal*. Tal como ocorre para os demais métodos, a convergência global para o método proposto é garantida para as

mesmas condições em que ambos os métodos *Elipsoidal* e *Branch-and-Bound* convergem.

O algoritmo 18 representa a codificação do método $MBCE$ tal como apresentado pelo fluxograma da figura 10.8. A busca pela solução ótima inicia-se com o método *Elipsoidal* e continua até que um ponto em que todas as restrições $g(\cdot) \leq 0$ seja encontrado, em cuja vizinhança não exista um ponto factível. Então, o elipsóide corrente será utilizado como o elipsóide inicial que define a região de interesse de busca do algoritmo *Branch-and-Cut*. Durante o processo de ramificação o elipsóide será cortado (comprimido) pelos limites LIC e LSC à medida que estes são encontrados. Este processo irá reduzir a região de interesse de busca na qual a solução dos problemas obtidos por relaxação deve ser encontrada. Como ganho adicional, o elipsóide reduzido também definirá um novo critério de interrupção do processo de ramificação. O algoritmo *Branch-and-Cut* utilizado corresponde ao algoritmo 18 ou uma implementação equivalente baseada na codificação proposta por Land e Doig. As mesmas definições descritas para os algoritmos anteriormente descritos, referentes à estrutura de restrições de igualdade secundárias $\mathbf{S}$ e a um problema $P_{i,j}$, são válidas.

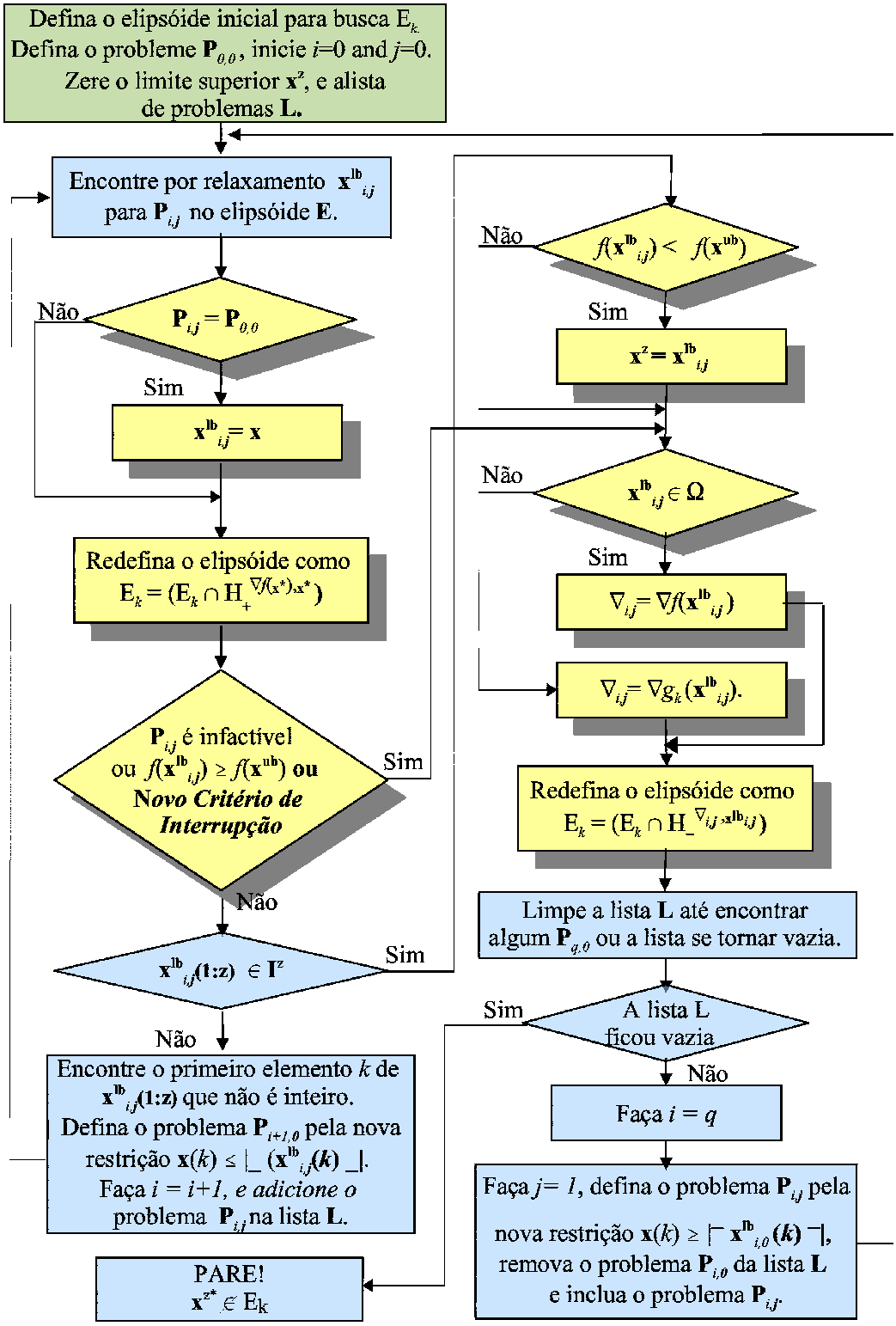


Figura 10.7: Fluxograma de um método *Branch-and-Cut* sobre a codificação de Dakin com solução do problema obtido por relaxação via *Elipsóide*.

Algorithm 16 *Branch-and-Cut* sobre proposição de Dakin

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^{z*}$ será procurada de acordo com o problema definido por (10.1). A estrutura de restrições de igualdade secundárias $\mathbf{S}$. A variável *PareBusca*.

Output: A melhor solução factível $\mathbf{x}^z$ encontrada. Para um problema infactível ter-se-á $\mathbf{x}^z = \emptyset$. O elipsóide de volume reduzido E_0.

1: **function** $BC^{Da}(E_k, \mathbf{S}, PareBusca)$
2: $i \leftarrow 0,\ j \leftarrow 0$
3: $\mathbf{x^z} \leftarrow \emptyset, \quad \mathbf{x}^{lb}_{i,j} \leftarrow \emptyset$ ▷ Zera o melhor ponto factível e os pontos dos LIC.
4: $\gamma \leftarrow \{1,\ldots,n\}, \quad \bar{\gamma} \leftarrow \emptyset$ ▷ Zera o conjunto de variáveis livres e forçadas.
5: $lP \leftarrow \emptyset$ ▷ Lista de problemas.
6: **loop** ▷ Laço principal do branch-and-bound.
7: **release** todas as regras de restrições de igualdade secundárias
8: **if** $\exists$ *algum* k *e* $\sigma \mid \mathbf{x}^{lb}_{i,j}(k) \leq \sigma$ *e* $\mathbf{x}^{lb}_{i,j}(k) \geq \sigma$ **then**
9: $\bar{\gamma} \leftarrow$ *todos os* $k, \quad \gamma \leftarrow \{l \mid l \in \{1,\ldots,n\} \mid l \notin \bar{\gamma}\}$
10: **end if**
11: **apply** $regra^l\ \forall\ \mathbf{S}^l == True$ ▷ Restrições de igualdade secundárias.
12: $\bar{\gamma} \leftarrow (\bar{\gamma} \cup l) \mid \forall\ \mathbf{S}^l == True, \quad \gamma \leftarrow \{l \mid l \in \{1,\ldots,n\} \mid l \notin \bar{\gamma}\}$
13: $\tilde{E}_0 \leftarrow E_k$
14: $\tilde{\mathbf{x}}_0(\bar{\gamma}) \leftarrow \mathbf{x}^{lb}_{i,j}(\bar{\gamma})$ ▷ Copie os valores fixos para o ponto que é o centro do elipsóide.
15: $\mathbf{x}^{lb}_{i,j} \leftarrow ERD(\tilde{E}_0, \gamma)$ ▷ Resolução do problema relaxado algoritmo 13.
16: **if** $i = 0$ and $j = 0$ **then**
17: $E_k \leftarrow (E_k \cap H_+^{\bar{\nabla} f(\mathbf{x}^{lb}_{i,j}), \mathbf{x}^{lb}_{i,j}})$ ▷ Atualize o elipsóide pelo LIC global.
18: **end if**
19: **if** $\mathbf{x}^{lb}_{i,j} = \emptyset$ ou $f(\mathbf{x}^{lb}_{i,j}) > f(\mathbf{x^z})$ ou *Critério de Interrupção de Ramificação* é aplicável **then**
20: **go to** linha 45
21: **end if**
22: **if** $\mathbf{x}^{lb}_{i,j}(1:z) \in \mathbb{I}^z_+$ **then** ▷ Ponto factível.
23: **if** $f(\mathbf{x}^{lb}_{i,j}) < f(\mathbf{x^z})$ **then** ▷ Melhor solução até o momento.
24: $\mathbf{x^z} \leftarrow \mathbf{x}^{lb}_{i,j}$ ▷ Atualiza LSC.
25: **if** $PareBusca == True$ **then**
26: **return** $\mathbf{x^z}$ e E_k ▷ Termine o algoritmo.
27: **end if**
28: **end if**
29: **go to** linha 36
30: **end if**
31: $k \leftarrow qualquer(\{l \mid l \in \{1,\ldots,z\} \mid \mathbf{x}^{lb}_{i,j}(l) \notin \mathbb{I}_+\})$
32: **add** nova restrição $\mathbf{x}(k) \leq \lfloor \mathbf{x}^{lb}_{i,j}(k) \rfloor$
33: $i \leftarrow i + 1, \quad j \leftarrow 0$
34: **add** problema $P_{i,j}$ na lista lP
35: **go to** linha 07
36: **if** $\mathbf{x}^{lb}_{i,j} \in \Omega$ **then**
37: $E_k \leftarrow (E_k \cap H_-^{\bar{\nabla} f(\mathbf{x}^{lb}_{i,j}), \mathbf{x}^{lb}_{i,j}})$ ▷ Atualize o elipsóide por ponto factível.
38: **else**
39: **loop** para toda l constraints $\mid l = 1,\ldots,p$
40: **if** $g_l(\mathbf{x}^{lb}_{i,j}) > 0$ **then**
41: $E_k \leftarrow (E_k \cap H_-^{\bar{\nabla} g_l(\mathbf{x}^{lb}_{i,j}), \mathbf{x}^{lb}_{i,j}})$ ▷ Atualize o elipsóide por um ponto infactível.
42: **end if**
43: **end loop**
44: **end if**
45: **loop** while $j = 1$
46: **restore** da lista lP o último problema e **remove** este problema da lP
47: **end loop**
48: **if** $lP = \emptyset$ **then**
49: **return** $\mathbf{x^z}$ e E_k ▷ Termine o algoritmo.
50: **end if**
51: **remove** última restrição $\mathbf{x}(k) \leq \lfloor \mathbf{x}^{lb}_{i,j}(k) \rfloor$
52: **add** nova restrição $\mathbf{x}(k) \geq \lceil \mathbf{x}^{lb}_{i,j}(k) \rceil$
53: **remove** problema $P_{i,0}$ da lista lP
54: $j \leftarrow 1$
55: **add** problema $P_{i,j}$ na lista lP
56: **end loop**
57: **end function**

Algorithm 17 *Elipsoidal* com Enumeração por Branch-and-Cut

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^{z*}$ será procurada de acordo com o problema definido por (10.1). A estrutura de restrições de igualdade secundárias $\mathbf{S}$. Um valor para precisão para o teste de limites $\hat{\epsilon} > 0$.

Output: A melhor solução factível $\mathbf{x}^z$ encontrada. Para um problema infactível ou para $\mathbf{x}^{z*} \notin E_0$ ter-se-á $\mathbf{x}^z = \emptyset$.

function $MEEBC(E_0, \mathbf{S}, \hat{\epsilon})$
  $k \leftarrow 0$
  $\mathbf{x}^z \leftarrow \emptyset$ ▷ Zera o melhor ponto factível
  loop ▷ Laço de movimentação do LSC.
    if $g_i(\mathbf{x}_k) \leq 0 \; \forall \; i = 1, \ldots, p$ **then**
      $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{x}_k)$
      $\{\mathbf{w}^z \mid \mathbf{w}^z(i) = \lceil \mathbf{x} \rceil \; \forall \; \bar{\nabla}_k(i) \geq 0 \; e \; i = 1, \ldots, z \mid \mathbf{w}^z(j) = \lfloor \mathbf{x} \rfloor \; \forall \; \bar{\nabla}_k(j) < 0 \; e \; j = 1, \ldots, z \mid \mathbf{w}^z(l) = \mathbf{x}(l) \; \forall \; l = z+1, \ldots, n\}$ ▷ Heurística da vizinhança.
      if $\mathbf{w}^z \notin \Omega$ e $f(\mathbf{w}^z) < f(\mathbf{x}^z)$ **then**
        $[\mathbf{w}^z, \mathbf{E}] \leftarrow BuscaBC(E_k, \bar{\nabla}_k, \mathbf{x}^z, \mathbf{S})$ ▷ Procura um ponto factível.
        if $\mathbf{w}^z = \emptyset$ **then**
          if $((f(\mathbf{x}^z) - f(\mathbf{x}_k)) \leq \hat{\epsilon})$ ou $(\frac{\bar{\nabla}_k^T}{|\bar{\nabla}_k|} * (\mathbf{x}^z - \mathbf{x}_k) \leq \hat{\epsilon})$ **then**
            return $\mathbf{x}^z$ ▷ Termine o algoritmo.
          end if
          $\bar{\nabla}_k \leftarrow -\bar{\nabla}_k$ ▷ $\mathbf{x}_k$ é o novo LIC
          $[\mathbf{w}^z, \mathbf{E}] \leftarrow BuscaBC(E_k, \bar{\nabla}_k, \mathbf{x}^z, \mathbf{S})$ ▷ Procura um ponto factível.
          if $\mathbf{w}^z = \emptyset$ **then**
            return $\mathbf{x}^z$ ▷ Termine o algoritmo.
          end if
        end if
      end if
      $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{w}^z)$
      $\mathbf{x}^z \leftarrow \mathbf{w}^z$ ▷ Melhor solução até o momento.
    else
      $\bar{\nabla}_k \leftarrow \bar{\nabla} g_i(\mathbf{x}_k)$ *para algum* $g_i(\mathbf{x}_k) > 0, \mid i = 1, \ldots, p$
      $\mathbf{w}^z \leftarrow \mathbf{x}_k$
    end if
    $E_{k+1} \leftarrow (E_k \cap H_-^{\bar{\nabla}_k, \mathbf{w}^z})$ ▷ Vide (2.14).
    if vol $(E_{k+1}) \simeq 0$ ou $vol(E_{k+1}) \simeq vol(E_k)$ **then**
      return $\mathbf{x}^z$ ▷ Termine o algoritmo.
    end if
    $k \leftarrow k + 1$
  end loop
end function

function $BuscaBC(E_k, \bar{\nabla}_k, \mathbf{x}^z, \mathbf{S})$ ▷ Esta função corresponde à busca de uma melhor solução que $\mathbf{x}^z$, por *Branch-and-Cut* no semi-elipsóide definido por $E_k \cap H_-^{\bar{\nabla}_k, \mathbf{x}_k}$.
  add restrição $(\bar{\nabla}_k)^T(\mathbf{x} - \mathbf{x}_k) \leq 0$
  if $\mathbf{x}^z \neq \emptyset$ **then**
    add restrição $(\bar{\nabla} f(\mathbf{x}^z))^T(\mathbf{x} - \mathbf{x}^z) \leq 0$
  end if
  $[\mathbf{x^z}, E_k] \leftarrow BC(E_k, \mathbf{S}, True)$ ▷ Algoritmo 16
  release restrições adicionadas.
  return o primeiro ponto factível $\mathbf{x}^z$ e o elipsóide E_k.
end function

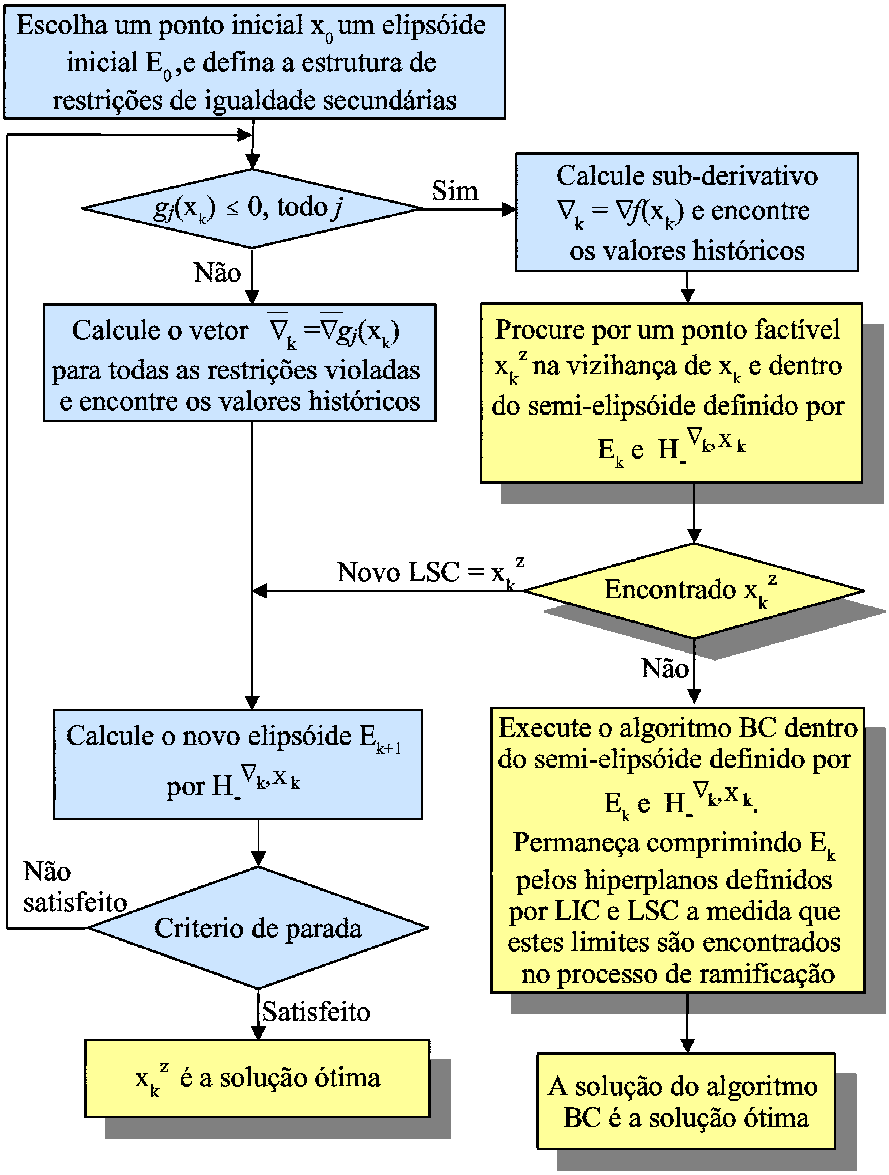


Figura 10.8: Fluxograma representativo do algoritmo *Branch-and-Cut Elipsoidal.*

Algorithm 18 *Branch-and-Cut Elipsoidal*

Input: Um elipsóide inicial E_0, onde a solução ótima $\mathbf{x}^{z*}$ será procurada de acordo com o problema definido por (10.1). A estrutura de restrições de igualdade secundárias $\mathbf{S}$.

Output: A melhor solução factível $\mathbf{x}^z$ encontrada. Para um problema infactível ou para $\mathbf{x}^{z*} \notin E_0$ ter-se-á $\mathbf{x}^z = \emptyset$.

function $MBCE(E_0, \mathbf{S})$
  $k \leftarrow 0$
  $\mathbf{x}^z \leftarrow \emptyset$ ▷ Zera o melhor ponto factível
  loop ▷ Laço de movimentação do LSC.
    if $g_i(\mathbf{x}_k) \leq 0 \; \forall \; i = 1, \ldots, p$ **then**
      $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{x}_k)$
      $\{\mathbf{w}^z \mid \mathbf{w}^z(i) = \lceil \mathbf{x} \rceil \; \forall \; \bar{\nabla}_k(i) \geq 0 \; e \; i = 1, \ldots, z \mid \mathbf{w}^z(j) = \lfloor \mathbf{x} \rfloor \; \forall \; \bar{\nabla}_k(j) < 0 \; e \; j = 1, \ldots, z \mid \mathbf{w}^z(l) = \mathbf{x}(l) \; \forall \; l = z+1, \ldots, n\}$ ▷ Heurística da vizinhança.
      if $\mathbf{w}^z \notin \Omega$ e $f(\mathbf{w}^z) < f(\mathbf{x}^z)$ **then**
        if $\mathbf{x}^z \neq \emptyset$ **then**
          add constraint $(\bar{\nabla} f(\mathbf{x}^z))^T (\mathbf{x} - \mathbf{x}^z) \leq 0$
        end if
        $[\mathbf{w}^z, \mathbf{E}] \leftarrow BC(E_k, \mathbf{S}, False)$ ▷ Algoritmo 16
        if $f(\mathbf{w}^z) < f(\mathbf{x}^z)$ **then**
          $\mathbf{x}^z \leftarrow \mathbf{w}^z$
        end if
        release restrições adicionadas.
        return $\mathbf{x}^z$ ▷ Termine o algoritmo.
      end if
      $\mathbf{x}^z \leftarrow \mathbf{w}^z$ ▷ Melhor solução até o momento.
      $\bar{\nabla}_k \leftarrow \bar{\nabla} f(\mathbf{w}^z)$ ▷ Define o semi-espaço para cálculo do novo elipsóide.
    else
      $\bar{\nabla}_k \leftarrow \bar{\nabla} g_i(\mathbf{x}_k)$ *para algum* $g_i(\mathbf{x}_k) > 0, \mid i = 1, \ldots, p$ ▷ Define o semi-espaço para cálculo do novo elipsóide.
      $\mathbf{w}^z \leftarrow \mathbf{x}_k$
    end if
    $E_{k+1} \leftarrow (E_k \cap H_-^{\bar{\nabla}_k, \mathbf{w}^z})$ ▷ Vide (2.14).
    if vol $(E_{k+1}) \simeq 0$ ou $vol(E_{k+1}) \simeq vol(E_k)$ **then**
      return $\mathbf{x}^z$ ▷ Termine o algoritmo.
    end if
    $k \leftarrow k + 1$
  end loop
end function

10.4 Conclusões do Capítulo

Neste capítulo foram apresentados os fundamentos para métodos baseados em *Elipsóide* com a característica de possuírem convergência global quando aplicados a problemas NIP e MINP através do processo de enumeração, quer seja implícita ou explícita.

Em seqüência foram propostas quatro variações do algoritmo *Elipsoidal*, sendo a primeira baseada em enumeração explícita, a segunda baseada em enumeração por *Branch-and-Bound* e as duas últimas baseadas num método proposto de *Branch-and-Cut*. Também foi proposta uma variação do método *Elipsoidal* para a solução de problemas com igualdades lineares, a ser utilizado nos algoritmos de *Branch-and-Bound* e *Branch-and-Cut*.

No capítulo seguinte serão detalhados os resultados obtidos com as variações propostas do algoritmo *Elipsoidal* para a solução de problemas NIP e MINP.

Capítulo 11

Resultados Computacionais para Problemas Discretos e Mistos

Neste capítulo são exibidos os problemas de simulação NIP e MINP e os resultados obtidos pelas variações do algoritmo *Elipsoidal* desenvolvidos a partir das definições, lemas e teoremas apresentados nos capítulos 8, 9 e 10.

Para referência da avaliação do desempenho dos algoritmos propostos, os resultados obtidos serão exibidos conjuntamente com os correspondentes resultados para o algoritmo *Branch-and-Bound*. A escolha do algoritmo *Branch-and-Bound* se deve ao fato deste algoritmo ser considerado, pela literatura, um algoritmo eficaz para a solução de problemas MINP. Para a solução dos subproblemas obtidos por relaxação das variáveis inteiras, o algoritmo *Branch-and-Bound* utilizará um algoritmo baseado em *Elipsóide*. Esta escolha se justifica pelo fato dos algoritmos baseados em *Elipsóide* serem considerados, pela literatura, algoritmos eficazes para a solução de problemas QCND. Nas tabelas e figuras com os resultados, a identificação $ALGORITMO^{LD}$ representa um algoritmo com a implementação do *Branch-and-Bound* a partir da proposição de Land e Doig, enquanto a identificação $ALGORITMO^{Da}$ representa a proposição de Dakin.

Quanto à codificação do método *Elipsoidal* utilizado como base do processo de enumeração e como algoritmo para a solução dos problemas obtidos por relaxação, considerou-se sempre a utilização do algoritmo $MPESC$, tal como apresentado na seção 6.1.2 para o cálculo do novo elipsóide. Ressalta-se que o algoritmo *Branch-and-Bound* de referência utilizou o mesmo algoritmo $MPESC$ para a solução do problema relaxado. Isto ocorreu com a finalidade de se distinguir os ganhos obtidos pelas proposições aqui avaliadas do ganho resultante da utilização dos valores históricos já apresentados na seção 7. Provavelmente, ao se utilizar o algoritmo $HISPE$ com a configuração $\tau \approx 3n$ obter-se-á resultados ainda melhores do que os aqui apresentados. Assim, espera-se com os resultados aqui

apresentados validar os métodos propostos, mesmo quando implementados com algoritmos baseados em *Elipsóide* sem a utilização de valores históricos para reduzir a região de interesse de busca. Também foi adotado o algoritmo de fatorização de *Cholesky*[1], vide seção 2.3.4, para melhorar a estabilidade numérica dos algoritmos propostos, evitando os erros que conduzem à perda da positividade de Q_k antes de uma convergência satisfatória.

Como detalhe da implementação, deve-se registrar que todos os algoritmos foram codificados utilizando-se a linguagem *MatLab*®, em sua versão 6.5, assim como todos os testes computacionais foram executados em um computador *AMD Turion*$^{64\,X2}$ sobre Microsoft *Windows*® *XP*.

Como critérios de parada para os laços dos algoritmos *Elipsoidais* foram utilizados a estabilização para o conjunto $\vartheta \triangleq \{\mathbf{x}_k, \ldots, \mathbf{x}_{k-4} \mid \mathbf{x} \in \mathbb{R}^n\}$ tal que a função erro definida como $\varepsilon(\cdot) = \{max(\mathbf{x}_i - \mathbf{x}_j) \mid \forall\, i \in \vartheta \mid \forall\, j \in \vartheta\}$ apresentasse valor inferior a um parâmetro $\epsilon = 10^{-6}$ fornecido, bem como a perda de positividade de Q_k.

Em todos os testes propostos, os limites de $\mathbf{x}$ foram definidos por um hipercubo $\mathbf{C} \subset \mathbb{R}^n_+$ de lado $lado^{\mathbf{C}}$ igual a 10, sendo que o elipsóide E_0 foi inicializado com a menor hiperesfera que continha o hipercubo, centralizado em ponto $\mathbf{x}_0$ gerado aleatoriamente de modo que $(E_0 \cap \Omega) \neq \emptyset$. Não foi definido um número máximo de iterações, de modo que o algoritmo sempre parasse por um dos critérios definidos.

De modo a gerar uma análise estatística preliminar dos resultados, cada problema definido por uma dimensão e para um número de variáveis inteiras específico foi testado com um conjunto de 10 parâmetros distintos gerados aleatoriamente, sendo que os resultados apresentados a seguir indicarão, salvo quando explicitamente informado, o valor médio dos resultados obtidos em cada teste.

Nas tabelas de resultados temos as seguintes informações apresentadas como parâmetros de avaliação de desempenho dos métodos:

Acessos $f(\mathbf{x})$ **:=** número de vezes em que a função objetivo foi avaliada. Esta informação é usada como medida do esforço computacional para avaliação do modelo matemático do problema.

Cálculos Q_{k+1} **:=** número de vezes em que o novo elipsóide E_{k+1} foi calculado. Esta informação é usada como medida do esforço computacional do algoritmo durante o processo de convergência para a solução ótima. Para os algoritmos *Branch-and-Bound*, este parâmetro representa a soma do número de elipsóides calculados pelo algoritmo baseado em *Elipsóide* durante a solução dos subproblemas obtidos por

[1]Nos testes realizados para problemas NIP e MINP observou-se um desempenho equivalente para a fatorização de *Cholesky* e a fatorização UDU.

relaxação. Para os algoritmos propostos, este parâmetro representa a soma do número de elipsóides calculados na solução dos subproblemas obtidos por relaxação e do número de elipsóides calculados durante as iterações inerentes aos algoritmos.

Tempo Total := tempo total em segundos para se encontrar a solução ótima $\mathbf{x}^{z*}$. Esta informação é usada como medida da soma do esforço computacional para avaliação do modelo matemático do problema e do esforço computacional do algoritmo durante o processo de convergência para a solução ótima. Para validar este parâmetro, a solução ótima encontrada por todos os métodos foi comparada. Considerando-se *Maxf* o valor máximo de $f(\cdot)$ avaliado no ponto ótimo encontrado pelos métodos sob comparação e *Minf* o valor mínimo de $f(\cdot)$ para os mesmos pontos ótimos encontrados, a fração $\frac{Minf}{Maxf}$ foi sempre menos que 10^{-5} em todos os problemas testados.

$100 * (Tempo)/(Tempo\ BB)$:= percentual do tempo total obtido com o método proposto sob teste com base no tempo total obtido com o algoritmo de referência *Branch-and-Bound*, considerando-se a mesma codificação, quer seja pela proposição de Dakin ou pela proposição de Land and Doig.

11.1 Testes de Convergência para o Algoritmo $MEHC$

Em função das limitações já descritas na seção 9.3 os resultados que aqui serão apresentados para o algoritmo $MEHC$ pretendem apenas demonstrar a capacidade deste algoritmo convergir para problemas NIP, com função-objetivo linear. Não serão, portanto, estabelecidas comparações de desempenho em relação aos demais algoritmos, quer sejam previamente existentes quer sejam propostos nesta Tese.

O problema discreto escalar com função-objetivo linear e restrições convexas é aqui denominado PIELC e a forma geral deste problema é representada pela formulação abaixo. Todas as variáveis são restritas pelo hipercubo $\mathbf{C} \subset \mathbb{R}^n_+$ com lados, neste caso em particular, igual a 4 e canto inferior no ponto $\{0, \ldots, 0\} \in \mathbb{R}^n$.

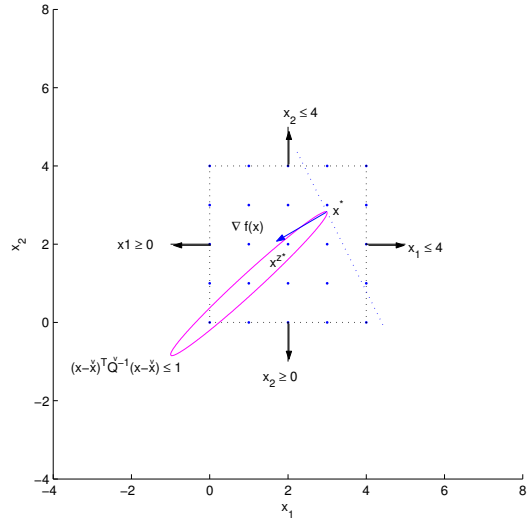
$$min\ f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} \tag{11.1}$$

$$s.t.\ \Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x} \in \mathbf{I}^n_+\}$$

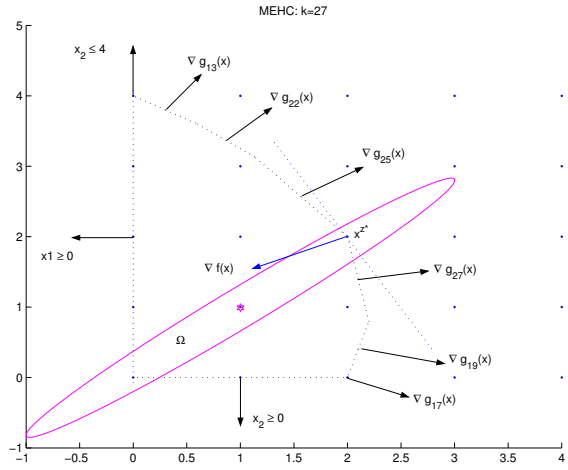
$$g(\mathbf{x}) = \begin{cases} g_1(\mathbf{x}) = (\mathbf{x} - \breve{\mathbf{x}})^T(\breve{Q})^{-1}(\mathbf{x} - \breve{\mathbf{x}}) \\ g_{1+i}(\mathbf{x}) = -\mathbf{x}(i) \mid i = 1, \ldots, n \\ g_{n+1+i}(\mathbf{x}) = \mathbf{x}(i) - lado^{\mathbf{C}} \mid i = 1, \ldots, n \end{cases}$$

onde $\{\mathbf{c} \in \mathbf{R}^n \mid |\mathbf{c}_j| \leq 2 \, \forall \, j\}$, $\breve{\mathbf{x}} \in \mathbf{C}$ e $\breve{Q} \in \mathbb{R}^{n \times n}$ é uma matriz simétrica definida positiva. O centro $\breve{\mathbf{x}}$ do elipsóide e o vetor $\mathbf{C}$ são parâmetros gerados aleatoriamente. A matriz $\breve{Q}$ foi construída com autovalores aleatórios de modo a garantir que
$(\breve{E} \cap \mathbf{C} \cap \mathbb{I}_+^n) \neq \emptyset$, sendo $\breve{E}$ o elipsóide definido pelo centro $\breve{\mathbf{x}}$ e pela matriz $\breve{Q}$.

A figura 11.1 exibe uma seqüência com os momentos inicial e final do processo de convergência do algoritmo *MEHC* para o problema definido por (11.1). A subfigura 11.1.*a* apresenta as restrições iniciais, de um problema testado, formadas pelo elipsóide $\breve{E}$ e o hipercubo $\mathbf{C}$. O ponto $\mathbf{x}^*$ que representa a solução do problema obtido por relaxação e o gradiente da função objeto também são exibidos. Já a subfigura 11.1.*b* apresenta o politopo resultante da adição dos diversos cortes calculados no processo de convergência e que coincide com a casca convexa do problema no ponto da solução ótima $\mathbf{x}^{z*}$. Claramente se observa nesta subfigura a tendência a formar uma curvatura convexa definida pelas restrições adicionadas, tal como discutido na seção 9.3.



a) Restrições inicialmente definidas para um problema PIELC.



b) Restrições finais para a solução de um problema PIELC.

Figura 11.1: Ilustração de uma seqüência de convergência do algoritmo *MEHC*.

Deve-se registrar ainda que no processo de teste realizado, o algoritmo *MEHC* foi capaz apenas de solucionar problemas PIELC para dimensões onde $n < 5$. Este comportamento está de acordo com as limitações comentadas na seção 9.3. Independentemente destas limitações, diversos problemas IP propostos em (Salkin & Mathur n.d., Wolsey n.d.) e cuja dimensão era reduzida, também foram solucionados com sucesso pelo algoritmo *MEHC*.

11.2 Problema MNIP Escalar Quasi-Convexo

Duas classes de problemas MNIP escalares quasi-convexos foram utilizadas para avaliar o desempenho dos algoritmos baseados em enumeração propostos. A primeira classe é composta por problemas QCND de diversas dimensões e combinações do número de variáveis inteiras nos quais os parâmetros foram gerados de forma aleatória. A segunda é composta por problemas mistos com variáveis binárias de engenharia química. A definição destes problemas e os resultados alcançados são exibidos nas próximas seções.

11.2.1 Problema Misto Contínuo e Discreto com Parâmetros Aleatórios

O problema misto discreto e contínuo, escalar e quasi-convexo é aqui denominado PMEQC e representado por (11.2). Todas as variáveis são restritas ao hipercubo $\mathbf{C} \subset \mathbb{R}^n_+$ e canto inferior esquerdo no ponto $\{0,\ldots,0\} \in \mathbb{R}^n$. Nenhuma regra de restrição de igualdade secundária foi necessária para os problemas PMEQC.

$$min\ f(\mathbf{x}) = max(\{-(1+e^{(\lambda_j(\mathbf{x}_j-\tilde{\mathbf{x}}_j))^2})\}) \mid j=1,\ldots,n \tag{11.2}$$

$$s.a.\ \Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x}(1:z) \in \mathbf{I}^z_+\ and\ \mathbf{x}(z+1:z+d) \in \mathbf{R}^d \mid n = d+z\}$$

$$g(\mathbf{x}) = \begin{cases} g_i(\mathbf{x}) = (\mathbf{x}-\breve{\mathbf{x}}_i)^T(\breve{Q}_i)^{-1}(\mathbf{x}-\breve{\mathbf{x}}_i) \mid i=1,\ldots,n+1 \\ g_{i+n+1}(\mathbf{x}) = -\mathbf{x}(i) \mid i=1,\ldots,n \\ g_{i+2n+1}(\mathbf{x}) = \mathbf{x}(i) - lado^{\mathbf{C}} \mid i=1,\ldots,n \end{cases}$$

na qual, $\tilde{\mathbf{x}}$ e $\breve{\mathbf{x}}_i \in \mathbf{C}$, $0 < \lambda_j < 10^{-1}$, e $\breve{Q}_i \in \mathbb{R}^{n\times n}$ é uma matriz simétrica definida positiva. O centro dos elipsóides $\breve{\mathbf{x}}_i$ e o ponto $\tilde{\mathbf{x}}$ são gerados de forma aleatória. As matrizes $\breve{Q}_i$ são construídas com autovalores aleatórios de modo a garantir que $(\breve{E}_1 \cap \ldots \breve{E}_{n+1} \cap \mathbb{I}^z_+) \neq \emptyset$, onde $\breve{E}_i$ representa o elipsóide definido pelo centro $\breve{\mathbf{x}}_i$ e pela matriz $\breve{Q}_i$.

O elipsóide inicial E_0 foi definido como a hiperesfera que contém $\mathbf{C}$. A precisão utilizada para determinar que um valor é considerado como inteiro $|\mathbf{x}(i) - \langle\mathbf{x}(i)\rangle|$ foi definida como 10^{-6}.

Para criar as tabelas e gráficos de resultados, os problemas PMEQC foram gerados para dimensões do intervalo $n = [3, 10]$. Para cada problema de uma determinada dimensão foram testados até 5 diferentes valores do número de variáveis inteiras, iniciando-se com o valor de $z = 1$ e terminando com $z = n$, o que determina um problema puramente discreto.

Inicialmente, os problemas $PMEQC$ foram utilizados para se estabelecer uma com-

paração de desempenho dos métodos *Elipsoidais* aqui propostos, $MEEE$, $MEEBB$, $MEEBC$ e $MBCE$, em relação ao algoritmo de *Branch-and-Bound*. A partir desta comparação, foram geradas as tabelas 11.1, 11.2 e 11.3 que exibem os resultados obtidos. Em função da limitação do algoritmo $MEEE$ de solucionar somente problemas NIP, os resultados para este algoritmo não são apresentados apenas os problemas com variáveis mistas inteiras e contínuas.

Algoritmos	$z : n$	Acessos $f(\mathbf{x})$	Cálculos Q_{k+1}	Tempo Total	$100 * (Tempo)/$ (Tempo BB)
$MEEBB^{LD}$	1 : 3	1204	5870	1.99	320.43
$MEEBC^{LD}$	1 : 3	3130	8029	2.86	460.16
$MBCE^{LD}$	1 : 3	604	529	0.43	68.88
BB^{LD}	1 : 3	796	1293	0.62	100.00
$MEEBB^{Da}$	1 : 3	296	985	2.02	283.94
$MEEBC^{Da}$	1 : 3	619	590	1.68	236.71
$MBCE^{Da}$	1 : 3	778	867	0.61	85.51
BB^{Da}	1 : 3	1076	1011	0.71	100.00
$MEEBB^{LD}$	2 : 3	1972	6718	3.23	387.59
$MEEBC^{LD}$	2 : 3	5343	10143	4.61	552.35
$MBCE^{LD}$	2 : 3	646	813	0.51	61.39
BB^{LD}	2 : 3	861	1500	0.83	100.00
$MEEBB^{Da}$	2 : 3	597	1821	4.62	335.66
$MEEBC^{Da}$	2 : 3	1240	1348	4.47	325.10
$MBCE^{Da}$	2 : 3	1080	1836	0.91	66.37
BB^{Da}	2 : 3	1843	2218	1.38	100.00
$MEEE$	3 : 3	753	865	2.50	287.07
$MEEBB^{LD}$	3 : 3	2194	7494	3.55	407.50
$MEEBC^{LD}$	3 : 3	7028	12635	5.87	672.74
$MBCE^{LD}$	3 : 3	754	849	0.55	63.24
BB^{LD}	3 : 3	914	1523	0.87	100.00
$MEEBB^{Da}$	3 : 3	1300	3428	6.18	285.17
$MEEBC^{Da}$	3 : 3	1821	2011	6.68	308.15
$MBCE^{Da}$	3 : 3	1601	2726	1.40	64.42
BB^{Da}	3 : 3	2674	3903	2.17	100.00

Tabela 11.1: Resultados comparativos dos métodos $MEEE$, $MEEBB$, $MEEBC$, $MBCE$ e BB para um problema PMEQC, dimensão 3.

Algoritmos	$z : n$	Acessos $f(\mathbf{x})$	Cálculos Q_{k+1}	Tempo Total	$100 * (Tempo)/$ (Tempo BB)
$MEEBB^{LD}$	1 : 4	1469	8977	2.95	149.89
$MEEBC^{LD}$	1 : 4	2784	11735	4.93	250.21
$MBCE^{LD}$	1 : 4	1090	760	0.94	47.89
BB^{LD}	1 : 4	2785	2578	1.97	100.00
$MEEBB^{Da}$	1 : 4	263	1392	2.56	93.92
$MEEBC^{Da}$	1 : 4	851	916	2.47	90.81
$MBCE^{Da}$	1 : 4	1552	1306	1.31	48.05
BB^{Da}	1 : 4	3933	2869	2.72	100.00
$MEEBB^{LD}$	2 : 4	3791	17856	6.05	187.59
$MEEBC^{LD}$	2 : 4	4734	14710	6.88	213.32
$MBCE^{LD}$	2 : 4	1754	2997	1.77	54.92
BB^{LD}	2 : 4	3717	5814	3.22	100.00
$MEEBB^{Da}$	2 : 4	714	2721	7.24	133.11
$MEEBC^{Da}$	2 : 4	1684	1912	4.87	89.52
$MBCE^{Da}$	2 : 4	3101	3058	2.49	45.84
BB^{Da}	2 : 4	7453	6279	5.44	100.00
$MEEBB^{LD}$	3 : 4	4027	15934	6.58	149.91
$MEEBC^{LD}$	3 : 4	4359	12842	6.57	149.74
$MBCE^{LD}$	3 : 4	2472	5099	2.59	59.10
BB^{LD}	3 : 4	4367	7344	4.39	100.00
$MEEBB^{Da}$	3 : 4	1273	3994	12.15	138.42
$MEEBC^{Da}$	3 : 4	2536	2559	6.74	76.81
$MBCE^{Da}$	3 : 4	5782	7016	4.82	54.94
BB^{Da}	3 : 4	11287	11460	8.78	100.00
$MEEE$	4 : 4	1673	1261	32.04	515.26
$MEEBB^{LD}$	4 : 4	5862	19158	9.25	148.75
$MEEBC^{LD}$	4 : 4	5746	18406	8.39	134.85
$MBCE^{LD}$	4 : 4	4819	8908	4.86	78.17
BB^{LD}	4 : 4	6031	9735	6.22	100.00
$MEEBB^{Da}$	4 : 4	2712	7294	19.83	128.33
$MEEBC^{Da}$	4 : 4	1767	1804	9.04	58.52
$MBCE^{Da}$	4 : 4	11522	17167	10.25	66.35
BB^{Da}	4 : 4	18375	24973	15.45	100.00

Tabela 11.2: Resultados comparativos dos métodos $MEEE$, $MEEBB$, $MEEBC$, $MBCE$ e BB para um problema PMEQC, dimensão 4.

Algoritmos	$z:n$	Acessos $f(\mathbf{x})$	Cálculos Q_{k+1}	Tempo Total	$100*(Tempo)/$ (Tempo BB)
$MEEBB^{LD}$	1 : 5	4536	16441	7.23	261.20
$MEEBC^{LD}$	1 : 5	4116	15895	6.72	242.91
$MBCE^{LD}$	1 : 5	1909	2131	1.81	65.33
BB^{LD}	1 : 5	3095	4139	2.77	100.00
$MEEBB^{Da}$	1 : 5	668	2001	4.94	141.97
$MEEBC^{Da}$	1 : 5	668	2041	4.84	139.09
$MBCE^{Da}$	1 : 5	2855	2138	2.27	65.12
BB^{Da}	1 : 5	4891	3203	3.48	100.00
$MEEBB^{LD}$	2 : 5	7526	29699	12.32	246.92
$MEEBC^{LD}$	2 : 5	7723	28424	11.94	239.31
$MBCE^{LD}$	2 : 5	2798	6417	3.37	67.55
BB^{LD}	2 : 5	3549	9743	4.99	100.00
$MEEBB^{Da}$	2 : 5	1649	4762	10.73	193.60
$MEEBC^{Da}$	2 : 5	1442	4418	10.50	189.45
$MBCE^{Da}$	2 : 5	4882	4570	4.06	73.24
BB^{Da}	2 : 5	6925	6039	5.54	100.00
$MEEBB^{LD}$	3 : 5	11098	43095	17.79	253.91
$MEEBC^{LD}$	3 : 5	13655	48952	20.68	295.07
$MBCE^{LD}$	3 : 5	3156	11417	4.77	68.02
BB^{LD}	3 : 5	4100	15833	7.01	100.00
$MEEBB^{Da}$	3 : 5	2563	6823	21.07	242.44
$MEEBC^{Da}$	3 : 5	2527	6764	22.74	261.74
$MBCE^{Da}$	3 : 5	6829	7884	6.10	70.19
BB^{Da}	3 : 5	10264	10081	8.69	100.00
$MEEBB^{LD}$	4 : 5	13436	52392	23.94	251.23
$MEEBC^{LD}$	4 : 5	13689	53204	24.23	254.30
$MBCE^{LD}$	4 : 5	4033	14477	6.38	66.96
BB^{LD}	4 : 5	4730	20154	9.53	100.00
$MEEBB^{Da}$	4 : 5	5101	12227	40.32	293.27
$MEEBC^{Da}$	4 : 5	6654	11347	39.33	286.04
$MBCE^{Da}$	4 : 5	10261	14291	9.83	71.48
BB^{Da}	4 : 5	15673	16939	13.75	100.00
$MEEE$	5 : 5	26951	1725	866.87	8898.13
$MEEBB^{LD}$	5 : 5	16798	64222	30.56	313.68
$MEEBC^{LD}$	5 : 5	17041	64562	30.66	314.69
$MBCE^{LD}$	5 : 5	4290	14592	6.65	68.31
BB^{LD}	5 : 5	4835	20620	9.74	100.00
$MEEBB^{Da}$	5 : 5	7015	16237	49.05	258.92
$MEEBC^{Da}$	5 : 5	6947	11339	41.55	219.34
$MBCE^{Da}$	5 : 5	11449	16283	11.03	58.19
BB^{Da}	5 : 5	21376	24853	18.95	100.00

Tabela 11.3: Resultados comparativos dos métodos $MEEE$, $MEEBB$, $MEEBC$, $MBCE$ e BB para um problema PMEQC, dimensão 5.

As tabelas 11.1, 11.2 e 11.3 mostram que os métodos baseados em enumeração não apresentaram resultados superiores aos alcançados com o algoritmo *Branch-and-Bound*. Como era esperado, vide seção 10.1, o método $MEEE$ apresentou um grande aumento no tempo total de convergência à medida que a dimensão foi aumentada. Já os métodos $MEEBB$ e $MEEBC$ apresentaram entre si resultados similares para os problemas, porém quase sempre não apresentaram resultados superiores aos alcançados com o algoritmo *Branch-and-Bound*, tanto para a codificação de *Branch-and-Bound* proposta por Land e Doig quanto para a proposição de Dakin. Este fato se deve primordialmente pelo lento processo de compressão do elipsóide na etapa final do algoritmo, até que um critério de parada seja alcançado. Entretanto, a principal importância destes métodos não diz respeito ao seu desempenho de tempo de convergência. Ela se deve à nova porta que estes métodos abrem para a implementação de um algoritmo que solucione problemas vetoriais mistos discretos e contínuos com funções quasi-convexas, de forma análoga aos métodos de otimização vetorial para variáveis contínuas apresentados em (Moura & Takahashi 2007). Como observação fundamental, as tabelas 11.1, 11.2 e 11.3 mostram o bom desempenho do algoritmo $MBCE$ comparado a todos os demais métodos.

Algoritmos	$z : n$	Acessos $f(\mathbf{x})$	Cálculos Q_{k+1}	Tempo Total	$100 * (Tempo)/$ (Tempo BB)
$MBCE^{LD}$	1 : 8	5592	6050	7.22	68.59
BB^{LD}	1 : 8	11219	9455	10.52	100.00
$MBCE^{Da}$	1 : 8	9381	3926	8.20	57.57
BB^{Da}	1 : 8	19244	6657	14.25	100.00
$MBCE^{LD}$	3 : 8	15054	29849	21.71	83.09
BB^{LD}	3 : 8	21080	31490	26.13	100.00
$MBCE^{Da}$	3 : 8	37020	18876	31.56	63.61
BB^{Da}	3 : 8	59908	28058	49.61	100.00
$MBCE^{LD}$	5 : 8	38339	109688	70.49	91.49
BB^{LD}	5 : 8	40849	118895	77.05	100.00
$MBCE^{Da}$	5 : 8	108770	76871	105.40	91.20
BB^{Da}	5 : 8	120961	82925	115.57	100.00
$MBCE^{LD}$	7 : 8	135551	450781	260.91	84.87
BB^{LD}	7 : 8	153107	428537	307.44	100.00
$MBCE^{Da}$	7 : 8	381299	415929	426.35	82.72
BB^{Da}	7 : 8	491778	457528	515.40	100.00
$MBCE^{LD}$	8 : 8	148877	459648	280.97	82.41
BB^{LD}	8 : 8	165385	487695	340.93	100.00
$MBCE^{Da}$	8 : 8	494146	544825	537.29	72.48
BB^{Da}	8 : 8	727982	753416	741.31	100.00

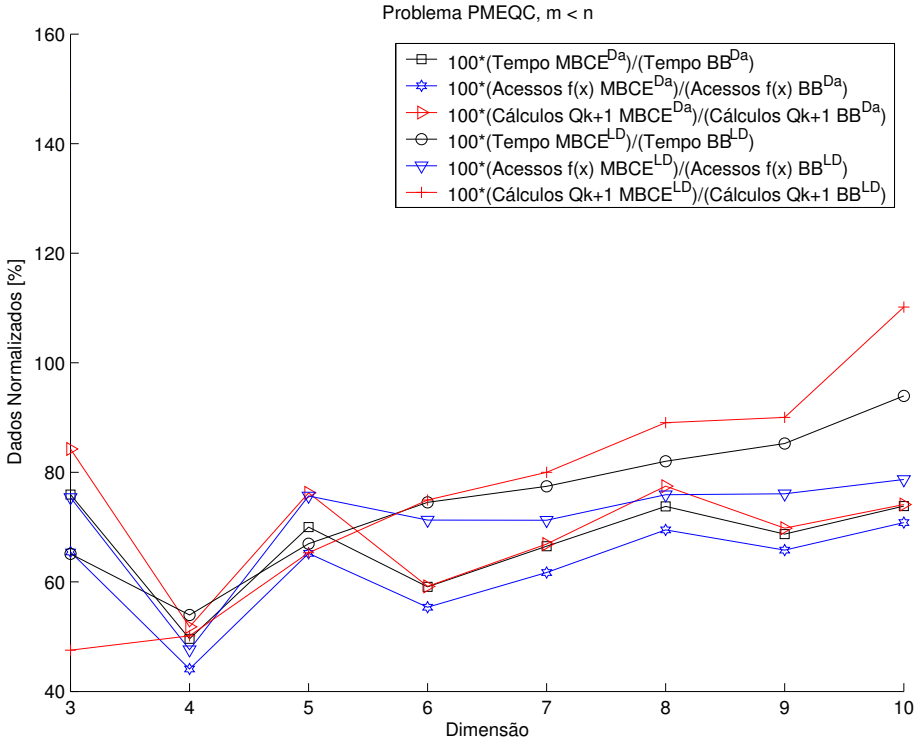
Tabela 11.4: Resultados comparativos dos métodos $MBCE$ e BB para um problema PMEQC, dimensão 8.

Para melhor analisar o desempenho do algoritmo $MBCE$ em relação ao algoritmo *Branch-and-Bound*, novos testes foram realizados. As tabelas 11.4 e 11.5 confirmam os bons resultados obtidos com o algoritmo $MBCE$ em comparação ao algoritmo *Branch-and-Bound*.

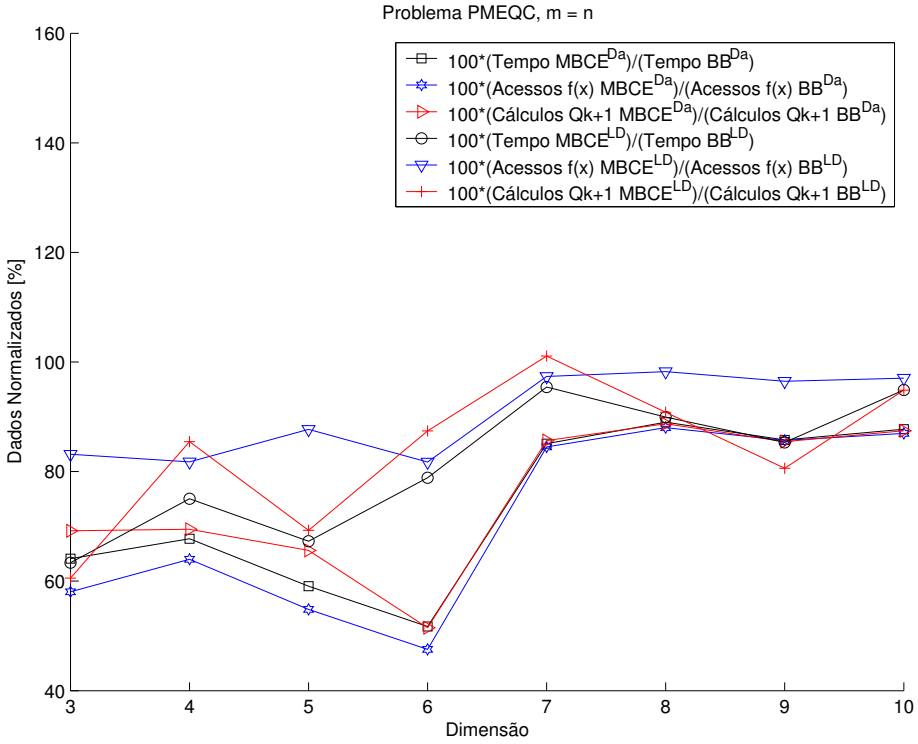
Algoritmos	$z:n$	Acessos $f(\mathbf{x})$	Cálculos Q_{k+1}	Tempo Total	$100*(Tempo)/$ (Tempo BB)
$MBCE^{LD}$	1 : 9	8381	7347	10.56	69.63
BB^{LD}	1 : 9	16055	11931	15.17	100.00
$MBCE^{Da}$	1 : 9	13391	4640	11.98	56.27
BB^{Da}	1 : 9	28109	8370	21.28	100.00
$MBCE^{LD}$	3 : 9	20503	39943	31.32	90.70
BB^{LD}	3 : 9	29815	36125	34.53	100.00
$MBCE^{Da}$	3 : 9	44355	17935	37.54	52.70
BB^{Da}	3 : 9	85021	34074	71.23	100.00
$MBCE^{LD}$	5 : 9	48870	119267	84.12	91.57
BB^{LD}	5 : 9	55388	118173	91.86	100.00
$MBCE^{Da}$	5 : 9	128885	69519	118.53	76.62
BB^{Da}	5 : 9	170528	87235	154.71	100.00
$MBCE^{LD}$	7 : 9	90458	378174	219.29	89.12
BB^{LD}	7 : 9	95121	434602	246.05	100.00
$MBCE^{Da}$	7 : 9	335601	275663	352.95	89.32
BB^{Da}	7 : 9	382187	301498	395.15	100.00
$MBCE^{LD}$	9 : 9	112481	469341	279.95	88.53
BB^{LD}	9 : 9	114402	559928	316.22	100.00
$MBCE^{Da}$	9 : 9	537762	533791	565.32	86.99
BB^{Da}	9 : 9	623761	588149	649.90	100.00

Tabela 11.5: Resultados comparativos dos métodos $MBCE$ e BB para um problema PMEQC, dimensão 9.

Os resultados obtidos para todos os casos testados para o método $MBCE$ são mostrados nas figuras 11.2, 11.3, e 11.4. Na figura 11.2 um resultado comparativo dos parâmetros *Acessos* $f(\mathbf{x})$, *Cálculos* Q_{k+1} e *Tempo Total* é apresentado como valor percentual, onde os valores obtidos com o algoritmo *Branch-and-Bound* são utilizados como valores base para este cálculo. A razão para o uso dos valores em percentuais não se deve apenas ao fato de todos os valores serem apresentados com referência ao mesmo eixo do gráfico. A razão também se deve à necessidade de se calcular um único valor representativo para cada dimensão. As linhas de valores do gráfico são geradas dividindo-se os resultados do algoritmo $MBCE$ pelos resultados do algoritmo *Branch-and-Bound* para a mesma codificação, Dakin ou Land and Doig.



a) Problema misto discreto e real QCND.



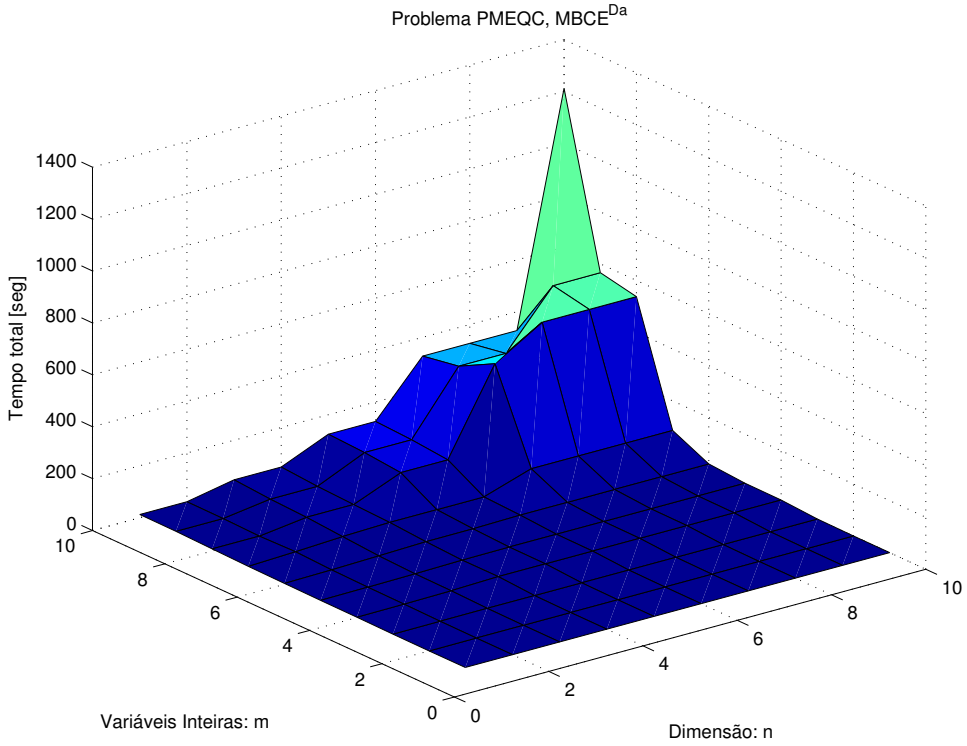
b) Problema discreto QCND

Figura 11.2: Resultados percentuais de Tempo, Acessos $f(\mathbf{x})$ e Cálculos Q_{k+1}: $MBCE^{LD}$ x BB^{LD} x $MBCE^{Da}$ x BB^{DL} para problemas PMEQC.

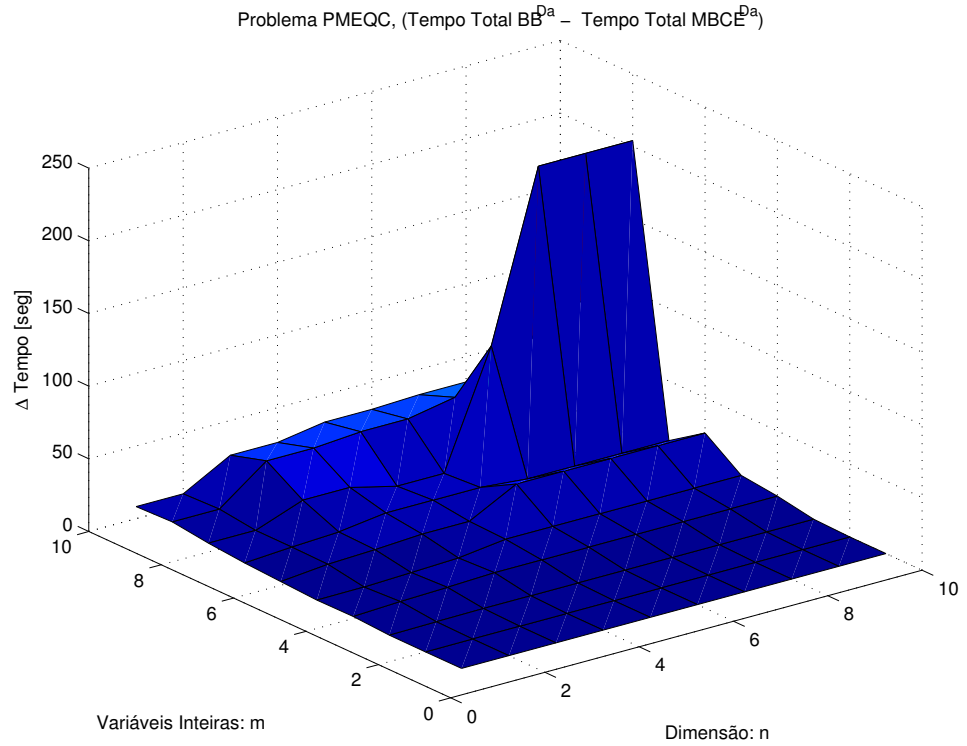
Os valores percentuais de *Acessos* $f(\mathbf{x})$ e *Tempo Total* permaneceram quase sempre abaixo de 90%, o que representa um considerável ganho obtido pelo algoritmo $MECB$ proposto. Todavia observa-se uma tendência de redução dos ganhos à medida que a dimensão aumenta, o que pode significar que este ganho poderá desaparecer para problemas de grande dimensão. Uma vez que o método *Elipsoidal* perde eficiência concomitantemente ao aumento da dimensão do problema, este comportamento é de certa forma esperado. O Teorema 8 (página 206) define uma distância $\triangle$ que deve estar compreendida no intervalo $(-\frac{1}{n}, 1]$ para que seja criado um corte eficiente. Como este intervalo se contrai quando do aumento da dimensão do problema o número de cortes capazes de reduzir o elipsóide corrente diminui simultaneamente ao aumento da dimensão. Conseqüentemente, este fato pode reduzir a eficiência do método de *Branch-and-Cut* aqui proposto para dimensões elevadas. O parâmetro *Cálculos* Q_{k+1} demonstra que o tempo total do método *Branch-and-Cut* foi inferior ao total do método *Branch-and-Bound* mesmo quando o algoritmo $MBCE$ despendeu mais esforço computacional por calcular um maior número de elipsóides.

A figura 11.3 e a figura 11.4 ilustram o crescimento exponencial do esforço computacional para resolver os problemas PMEQC com o aumento da dimensão e do número de variáveis inteiras, nas subfiguras 11.3.a e 11.4.a. Como pode ser observado, o tempo total para convergência aumenta de forma exponencial em função da dimensão do problema independentemente do número de variáveis inteiras z existentes. Em função do tempo total para convergência aumentar exponencialmente e do valor percentual do tempo total tender para 100%, a diferença em segundos dos resultados do tempo total de convergência, ($\Delta Tempo = Tempo\ Total\ BB$ - $Tempo\ Total\ MBEC$), é mostrada nas subfiguras 11.3.b e 11.4.b. Este resultado é particularmente interessante, uma vez que os gráficos mostram a mesma tendência de comportamento exponencial para $\Delta Tempo$. Conseqüentemente, isto permite a interpretação de que o método de *Branch-and-Cut* aqui proposto pode ser considerado eficiente quando comparado com o método de *Branch-and-Bound*.

Já a figura 11.5 mostra a contração do volume do elipsóide corrente utilizado para o cálculo dos problemas obtidos por relaxação das variáveis, durante o processo de *Branch-and-Cut* para os algoritmos $MBCE^{Da}$ e $MBCE^{LD}$. Considerou-se como base para o volume do elipsóide corrente o determinante da matriz Q_k. Para um problema PMEQC específico, no qual a dimensão é $n = 4$ e o número de variáveis inteiras $z = 3$, o algoritmo *Branch-and-Cut* baseado na proposição de Dakin produziu uma contração mais eficiente do volume do elipsóide corrente. Este fato também contribuiu para reduzir o número de problemas obtidos por relaxação das variáveis que foram avaliados. Isto pode ser afirmado uma vez que a linha do gráfico para o algoritmo $MBCE^{Da}$ é a menos extensa.

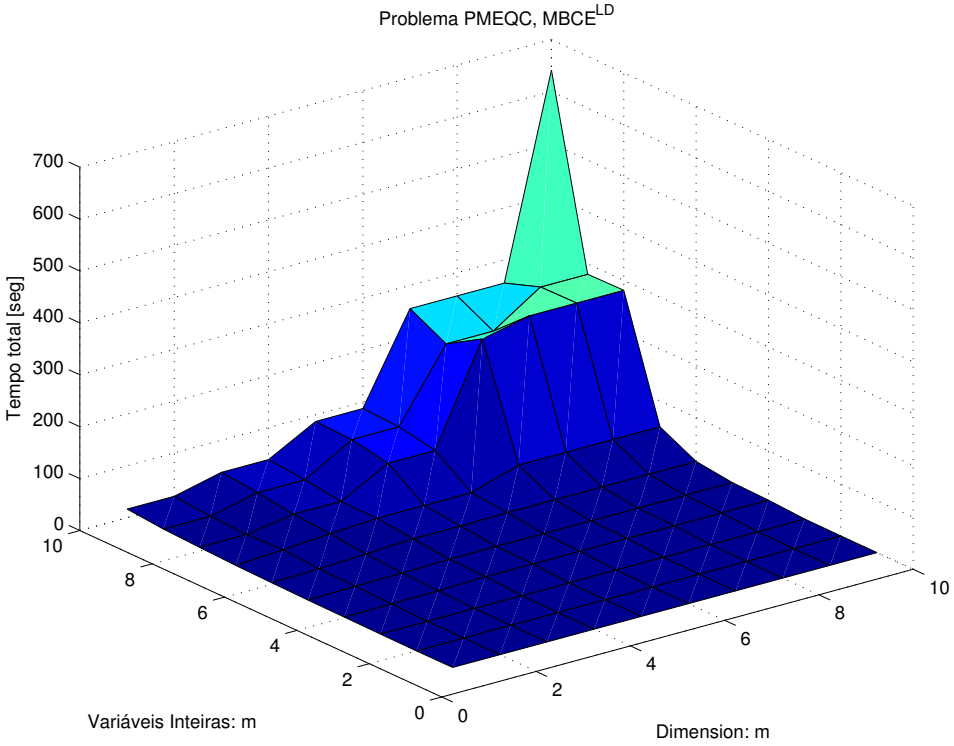


a) Tempo total para $MBCE^{Da}$.

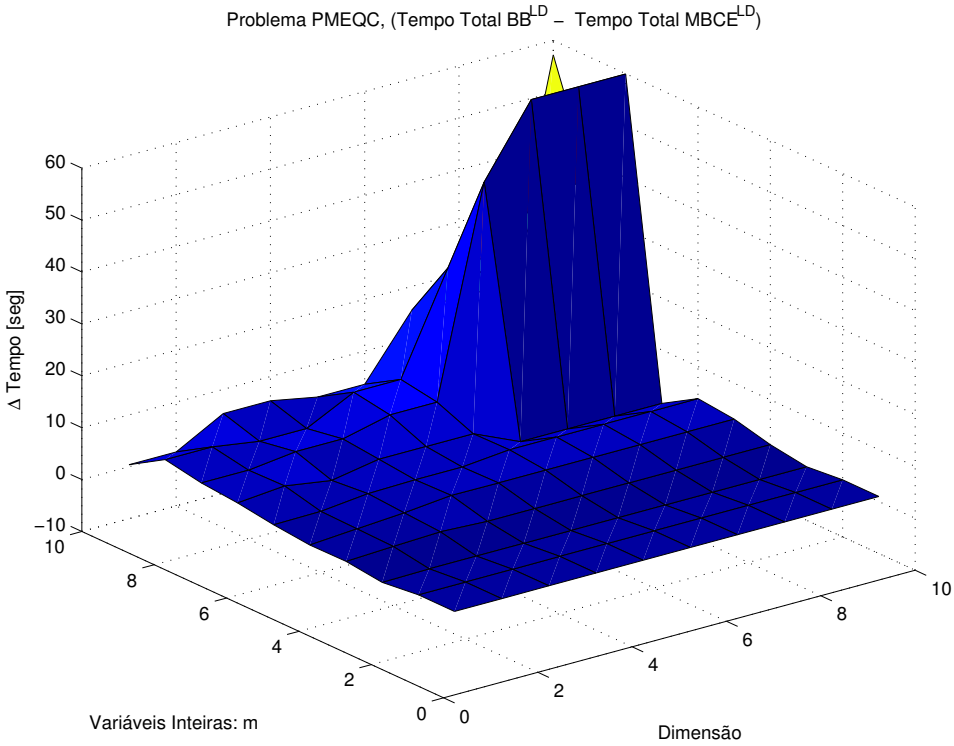


b) Δ Tempo total para $MBCE^{Da}$ e BB^{Da}.

Figura 11.3: Resultados dos problemas PMEQC para BB^{Da} e $MBCE^{Da}$: número de variáveis inteiras z x dimensão n.



a) Tempo total para $MBCE^{LD}$.



b) Δ tempo total para $MBCE^{LD}$ e BB^{LD}.

Figura 11.4: Resultados dos problemas PMEQC para BB^{LD} e $MBCE^{LD}$: número de variáveis inteiras z x dimensão n.

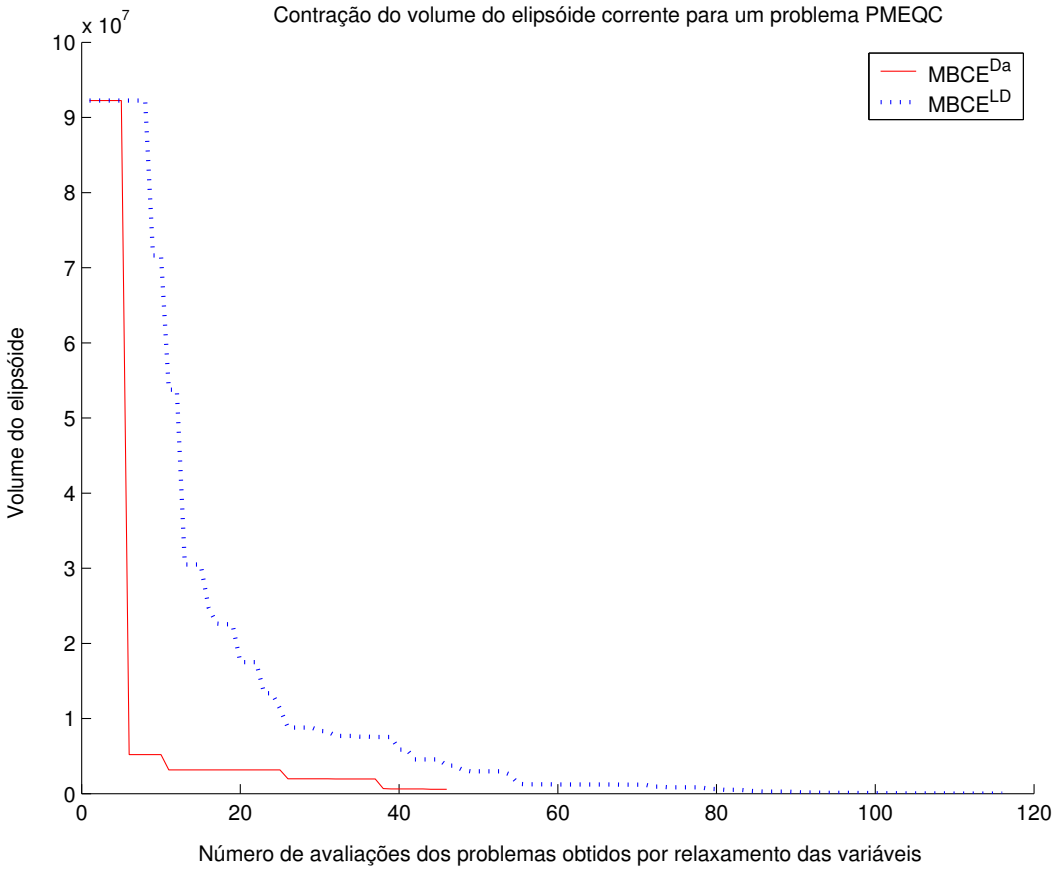


Figura 11.5: Contração do volume do elipsóide para os algoritmos $MBCE^{Da}$ e $MBCE^{LD}$, dimensão $n = 4$ e $z = 3$.

11.2.2 Problema Binário para Processo Químico de Sintetização

Dois problemas referentes ao processo químico de sintetização (Duran & Grossmann 1986) são representados por (11.3) e (11.4). Estes problemas representam a classe de problemas binários escalares quasi-convexos, PBEQC, apesar do fato dos problemas serem realmente pseudo-convexos. O objetivo da utilização destes problemas como teste se deve não apenas ao fato de se tratarem de casos reais, mas também por serem problemas binários, os quais teoricamente constituem a classe de problemas mais difíceis de serem solucionados pelos métodos propostos com ganhos de desempenho.

11.2.2.1 Processo de Sintetização 1

O primeiro problema associado a um processo químico de sintetização é apresentado a seguir:

$$\begin{aligned} min\ f(\mathbf{x}) \ &= \ 5\mathbf{x}(1) + 6\mathbf{x}(2) + 8\mathbf{x}(3) + 10\mathbf{x}(4) - 18ln(\mathbf{x}(5)+1) + \\ &\quad -7\mathbf{x}(6) - 19.2ln(\mathbf{x}(4) - \mathbf{x}(5) + 1) + 10 \\ s.a.\ \Omega &= \{g(\mathbf{x}) \leq 0 \mid \mathbf{x}(1:3) \in \{0,1\}^3\ e\ \mathbf{x}(4:6) \in \mathbf{R}^3_+\} \end{aligned} \tag{11.3}$$

$$g(\mathbf{x}) \triangleq \begin{cases} g_1(\mathbf{x}) := -0.8ln(\mathbf{x}(5)+1) - 0.96ln(\mathbf{x}(4) - \mathbf{x}(5) + 1) + 0.8\mathbf{x}(6) \leq 0 \\ g_2(\mathbf{x}) := -ln(\mathbf{x}(5)+1) - 1.2 * ln(\mathbf{x}(4) - \mathbf{x}(5) + 1) + \mathbf{x}(6) + 2 * \mathbf{x}(3) - 2 \leq 0 \\ g_3(\mathbf{x}) := \mathbf{x}(5) - \mathbf{x}(4) \leq 0 \\ g_4(\mathbf{x}) := \mathbf{x}(5) - 2\mathbf{x}(1) \leq 0 \\ g_5(\mathbf{x}) := \mathbf{x}(4) - \mathbf{x}(5) - 2\mathbf{x}(2) \leq 0 \\ g_6(\mathbf{x}) := \mathbf{x}(1) + \mathbf{x}(2) \leq 1 \\ g_{6+i} := -\mathbf{x}(i) \leq 0 \mid i = 1, \ldots, 6 \\ g_{12+i} := \mathbf{x}(i) - 1 \leq 0 \mid i = 1, 2, 3 \end{cases}$$

O elipsóide inicial E_0 foi definido de modo a garantir que $E_0 \supset \Omega$. Deve-se registrar que o hipercubo $\mathbf{C}$ neste caso é um hiperparalelepípedo de lado $lado^{\mathbf{C}}$ igual a 10 para as variáveis contínuas e $lado^{\mathbf{C}}$ igual a 1 para as variáveis inteiras. A solução ótima é previamente conhecida como $\mathbf{x}^{z*} = [0\ 1\ 0\ 1.30097\ 0\ 1]^T$. A precisão utilizada para determinar que um valor é considerado como inteiro $|\mathbf{x}(i) - \langle \mathbf{x}(i) \rangle|$ foi definida como 10^{-6}.

A tabela 11.6 exibe os resultados para o problema de sintetização 1 alcançados pelos algoritmos BB^{Da}, BB^{LD}, $MBCE^{Da}$, and $MBCE^{Ld}$. Em acréscimo aos parâmetros já utilizados para a medição de desempenho, um novo parâmetro $\Delta f/f(\mathbf{x}^{z*})$ foi incluído.

Este parâmetro representa o valor médio do erro entre a diferença da solução ótima conhecida $\mathbf{x}^{z*}$ e da solução obtida pela execução de cada algoritmo dividida pelo valor da solução ótima conhecida $\mathbf{x}^{z*}$.

Algoritmos	$z:n$	Acessos $f(\mathbf{x})$	Cálculos Q_{k+1}	Tempo Total	$100*(Tempo)/$ (Tempo BB)	$\Delta f/$ $f(\mathbf{x}^{z*})$
$MBCE^{LD}$	3 : 6	17218	5217	4.63	96.92	7.8e-006
BB^{LD}	3 : 6	17776	5686	4.78	100.00	7.7e-006
$MBCE^{Da}$	3 : 6	17373	5379	4.72	97.42	8.1e-006
BB^{Da}	3 : 6	17707	5699	4.85	100.00	7.7e-006

Tabela 11.6: Resultados do problema de sintetizacao 1.

Os resultados para este teste indicam uma pequena diferença em favor dos métodos propostos, para um mesmo patamar de erro $\Delta f/f(\mathbf{x}^{z*})$ alcançado. A principal razão para a pequena melhora está no fato de existirem poucos cortes eficientes gerados durante o processo de convergência. Uma vez que o problema é binário, não há muitas oportunidades para encontrar diferentes pontos associados ao LSC e soluções infactíveis de subproblemas, os quais poderiam ser utilizados para gerar cortes eficientes. Este resultado é uma indicação preliminar de que o algoritmo $MBCE$ perde eficiência para problemas binários, entretanto o método permanece como uma alternativa válida para a solução de problemas PBEQC.

11.2.2.2 Processo de Sintetização 2

O segundo problema associado a um processo químico de sintetização é apresentado a seguir:

$$\begin{aligned} min\ f(\mathbf{x}) \ &= \ 5\mathbf{x}(1) + 8\mathbf{x}(2) + 6\mathbf{x}(3) + 10\mathbf{x}(4) + 6\mathbf{x}(5) + \\ &-10\mathbf{x}(6) - 15\mathbf{x}(7) + 15\mathbf{x}(8) + 5\mathbf{x}(9) - 15\mathbf{x}(10) - 20\mathbf{x}(11) \\ &s.a.\ \Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x}(1:5) \in \{0,1\}^5\ e\ \mathbf{x}(6:11) \in \mathbf{R}_+^6\} \end{aligned} \tag{11.4}$$

$$\Omega \triangleq \begin{cases} -ln(\mathbf{x}(8)+\mathbf{x}(9)+1) \leq 0 \\ -\mathbf{x}(6)-\mathbf{x}(7)-\mathbf{x}(10)+\mathbf{x}(9)+2\mathbf{x}(11) \leq 0 \\ -\mathbf{x}(6)-\mathbf{x}(7)-0.75\mathbf{x}(10)+\mathbf{x}(8)+2\mathbf{x}(11) \leq 0 \\ \mathbf{x}(10)-\mathbf{x}(11) \leq 0 \\ 2\mathbf{x}(10)-\mathbf{x}(8)-2\mathbf{x}(11) \leq 0 \\ -0.5\mathbf{x}(8)+\mathbf{x}(9) \leq 0 \\ 0.2\mathbf{x}(8)-\mathbf{x}(9) \leq 0 \\ e^{\mathbf{x}(6)}-10\mathbf{x}(1) \leq 0 \\ e^{\mathbf{x}(7)/1.2}-10\mathbf{x}(2) \leq 0 \\ 1.25\mathbf{x}(10)-10\mathbf{x}(8) \leq 0 \\ -2\mathbf{x}(8)+2\mathbf{x}(11)-10\mathbf{x}(5) \leq 0 \\ \mathbf{x}(1)+\mathbf{x}(2)-1=0 \\ \mathbf{x}(4)+\mathbf{x}(5)-1 \leq 0 \\ 0 \leq \mathbf{x}(6) \leq 2 \\ 0 \leq \mathbf{x}(7) \leq 2 \\ 0 \leq \mathbf{x}(8) \\ 0 \leq \mathbf{x}(9) \\ 0 \leq \mathbf{x}(10) \leq 2 \\ 0 \leq \mathbf{x}(11) \leq 3 \end{cases}$$

Similarmente ao realizado para o problema 1, o elipsóide inicial E_0 foi definido de modo a garantir que $E_0 \supset \Omega$. Novamente deve-se registrar que o hipercubo $\mathbf{C}$ neste caso é um hiperparalelepípedo de lado $lado^{\mathbf{C}}$ igual a 10 para as variáveis contínuas e $lado^{\mathbf{C}}$ igual a 1 para as variáveis inteiras. A solução ótima é previamente conhecida como $\mathbf{x}^{z*} = [0\ 1\ 1\ 1\ 0\ 0\ 2\ 0.6520\ 0.3260\ 1.0784\ 1.0784]^T$. A precisão utilizada para determinar que um valor é considerado como inteiro $|\mathbf{x}(i) - \langle \mathbf{x}(i) \rangle|$ foi definida como 10^{-5}.

A tabela 11.7 exibe os resultados para o problema de sintetização 2 alcançados pelos algoritmos BB^{Da}, BB^{LD}, $MBCE^{Da}$, and $MBCE^{Ld}$. Ressalta-se que o novo novo parâmetro de avaliação de desempenho $\Delta f/f(\mathbf{x}^{z*})$ também foi incluído.

Novamente os resultados para este teste indicam uma pequena diferença em favor dos métodos propostos, para um mesmo patamar de erro $\Delta f/f(\mathbf{x}^{z*})$ alcançado. Todavia, o método permanece como uma alternativa válida para a solução de problemas PBEQC.

Algoritmos	$z : n$	Acessos $f(\mathbf{x})$	Cálculos Q_{k+1}	Tempo Total	$100 * (Tempo)/$ (Tempo BB)	$\Delta f/$ $f(\mathbf{x}^{z*})$
$MBCE^{LD}$	5 : 10	158026	31573	38.29	99.07	3.2e-006
BB^{LD}	5 : 10	159178	32967	38.65	100.00	3e-006
$MBCE^{Da}$	5 : 10	154205	34070	39.04	99.50	1.9e-005
BB^{Da}	5 : 10	153847	35260	39.24	100.00	1.9e-005

Tabela 11.7: Resultados do problema de sintetizacao 2.

11.3 Problema MNIP Escalar Convexo

Em função dos resultados obtidos para os problemas quasi-convexos, os métodos propostos foram também avaliados quanto ao seu desempenho para a solução de problemas convexos. O problema misto discreto e contínuo, escalar e convexo é aqui denominado PMEC e representado por (11.5). Todas as variáveis são restritas ao hipercubo $\mathbf{C} \subset \mathbb{R}^n_+$ e canto inferior esquerdo no ponto $\{0, \ldots, 0\} \in \mathbb{R}^n$. Nenhuma regra de restrição de igualdade secundária foi necessária para os problemas PMEQC.

$$min\ f(\mathbf{x}) = max(\{(\mathbf{x} - \tilde{\mathbf{x}}_j)^T (\tilde{Q}_j)^{-1} (\mathbf{x} - \tilde{\mathbf{x}}_j)\}) \mid j = 1, \ldots, n \tag{11.5}$$

$$s.a.\ \Omega = \{g(\mathbf{x}) \leq 0 \mid \mathbf{x}(1:z) \in \mathbf{I}^z_+ \ e\ \mathbf{x}(z+1:z+d) \in \mathbf{R}^d \mid n = d + z\}$$

$$g(\mathbf{x}) = \begin{cases} g_i(\mathbf{x}) = (\mathbf{x} - \breve{\mathbf{x}}_i)^T (\breve{Q}_i)^{-1} (\mathbf{x} - \breve{\mathbf{x}}_i) \mid i = 1, \ldots, n+1 \\ g_{i+n+1}(\mathbf{x}) = -\mathbf{x}(i) \mid i = 1, \ldots, n \\ g_{i+2n+1}(\mathbf{x}) = \mathbf{x}(i) - lado^{\mathbf{C}} \mid i = 1, \ldots, n \end{cases}$$

na qual, $\tilde{\mathbf{x}}_j$ e $\breve{\mathbf{x}}_i \in \mathbf{C}$, $\breve{Q}_j$ e $\breve{Q}_i \in \mathbb{R}^{n \times n}$ são matrizes simétricas definidas positivas. Os centros $\breve{\mathbf{x}}_i$ e $\tilde{\mathbf{x}}_j$ dos elipsóides são gerados de forma aleatória. As matrizes $\tilde{Q}_j$ são construídas com autovalores aleatórios. As matrizes $\breve{Q}_i$ são construídas com autovalores aleatórios de modo a garantir que $(\breve{E}_1 \cap \ldots \breve{E}_{n+1} \cap \mathbb{I}^z_+) \neq \emptyset$, onde $\breve{E}_i$ representa o elipsóide definido pelo centro $\breve{\mathbf{x}}_i$ e pela matriz $\breve{Q}_i$.

Similar ao definido para os problemas quasi-convexos, o elipsóide inicial E_0 foi definido como a hiperesfera que contém $\mathbf{C}$. A precisão utilizada para determinar que um valor é considerado como inteiro $|\mathbf{x}(i) - \langle \mathbf{x}(i) \rangle|$ foi definida como 10^{-6}.

Para criar as tabelas e gráficos de resultados, os problemas PMEC foram gerados para dimensões do intervalo $n = [3, 11]$. Para cada problema de uma determinada dimensão foram testados até 5 diferentes valores do número de variáveis inteiras, iniciando-se com o valor de $z = 1$ e terminando com $z = n$, o que determina um problema puramente discreto.

Algoritmos	$z:n$	Acessos $f(\mathbf{x})$	Cálculos Q_{k+1}	Tempo Total	$100*(Tempo)/$ (Tempo BB)
$MBCE^{LD}$	1 : 5	1475	3852	2.00	56.48
BB^{LD}	1 : 5	3456	5942	3.54	100.00
$MBCE^{Da}$	1 : 5	1943	3169	2.15	59.99
BB^{Da}	1 : 5	3797	4810	3.58	100.00
$MBCE^{LD}$	2 : 5	2155	8308	3.54	53.77
BB^{LD}	2 : 5	6458	10996	6.59	100.00
$MBCE^{Da}$	2 : 5	3355	5819	3.66	43.02
BB^{Da}	2 : 5	9050	11328	8.51	100.00
$MBCE^{LD}$	3 : 5	3368	12295	5.16	54.10
BB^{LD}	3 : 5	9684	16227	9.53	100.00
$MBCE^{Da}$	3 : 5	6608	10250	6.67	41.48
BB^{Da}	3 : 5	18046	20891	16.09	100.00
$MBCE^{LD}$	4 : 5	5478	17613	8.26	60.16
BB^{LD}	4 : 5	14194	21295	13.73	100.00
$MBCE^{Da}$	4 : 5	14174	19944	13.22	47.84
BB^{Da}	4 : 5	32848	36826	27.63	100.00
$MBCE^{LD}$	5 : 5	5607	18977	8.65	58.17
BB^{LD}	5 : 5	15409	23188	14.87	100.00
$MBCE^{Da}$	5 : 5	18149	25014	16.34	43.22
BB^{Da}	5 : 5	47349	49981	37.82	100.00

Tabela 11.8: Resultados comparativos dos métodos $MBCE$ e BB para um problema PMEC, dimensão 5.

As tabelas 11.8, 11.9 e 11.10 exibem os resultados obtidos pelo método $MBCE$ em comparação ao algoritmo *Branch-and-Bound*, tanto para a codificação de *Branch-and-Bound* proposta por Land e Doig quanto para a proposição de Dakin. Tal como ocorrido para os problemas quasi-convexos, observa-se claramente que o algoritmo $MBCE$ apresentou sempre os melhores resultados.

Algoritmos	$z : n$	Acessos $f(\mathbf{x})$	Cálculos Q_{k+1}	Tempo Total	$100 * (Tempo)/$ (Tempo BB)
$MBCE^{LD}$	1 : 7	2346	9660	5.52	71.36
BB^{LD}	1 : 7	5713	10678	7.73	100.00
$MBCE^{Da}$	1 : 7	3491	5136	4.56	59.16
BB^{Da}	1 : 7	7273	7458	7.71	100.00
$MBCE^{LD}$	2 : 7	4213	19062	10.10	62.45
BB^{LD}	2 : 7	12239	22657	16.18	100.00
$MBCE^{Da}$	2 : 7	6782	10977	9.03	47.40
BB^{Da}	2 : 7	18490	18461	19.05	100.00
$MBCE^{LD}$	3 : 7	6344	29403	15.13	59.51
BB^{LD}	3 : 7	19474	35841	25.43	100.00
$MBCE^{Da}$	3 : 7	12037	18361	15.19	42.27
BB^{Da}	3 : 7	36232	34843	35.93	100.00
$MBCE^{LD}$	4 : 7	12904	47599	24.99	66.03
BB^{LD}	4 : 7	30469	54364	37.85	100.00
$MBCE^{Da}$	4 : 7	23803	32954	27.52	45.97
BB^{Da}	4 : 7	63102	58450	59.87	100.00
$MBCE^{LD}$	7 : 7	47774	109072	61.21	60.66
BB^{LD}	7 : 7	87255	133663	100.91	100.00
$MBCE^{Da}$	7 : 7	114361	120779	97.47	48.37
BB^{Da}	7 : 7	246861	205611	201.53	100.00

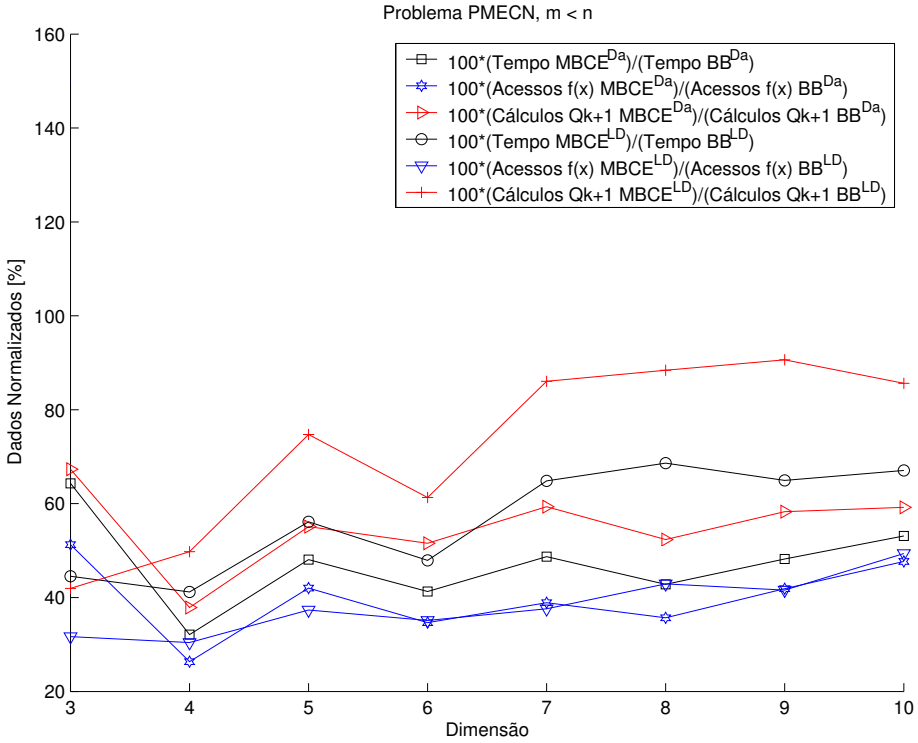
Tabela 11.9: Resultados comparativos dos métodos $MBCE$ e BB para um problema PMEC, dimensão 7.

Algoritmos	$z:n$	Acessos $f(\mathbf{x})$	Cálculos Q_{k+1}	Tempo Total	$100*(Tempo)/$ (Tempo BB)
$MBCE^{LD}$	1 : 9	3481	14985	8.89	65.82
BB^{LD}	1 : 9	8218	16047	13.50	100.00
$MBCE^{Da}$	1 : 9	5026	7588	7.35	60.87
BB^{Da}	1 : 9	9638	10860	12.08	100.00
$MBCE^{LD}$	3 : 9	10554	56647	30.72	64.59
BB^{LD}	3 : 9	27035	62343	47.56	100.00
$MBCE^{Da}$	3 : 9	22249	30306	28.99	43.42
BB^{Da}	3 : 9	59768	57184	66.77	100.00
$MBCE^{LD}$	5 : 9	19194	96441	52.10	63.59
BB^{LD}	5 : 9	45172	112757	81.93	100.00
$MBCE^{Da}$	5 : 9	49862	57355	58.05	39.37
BB^{Da}	5 : 9	139440	119992	147.46	100.00
$MBCE^{LD}$	7 : 9	64071	284864	150.11	65.74
BB^{LD}	7 : 9	151077	307268	228.35	100.00
$MBCE^{Da}$	7 : 9	157071	193837	186.20	49.19
BB^{Da}	7 : 9	370560	309877	378.51	100.00
$MBCE^{LD}$	9 : 9	64544	298585	158.12	62.10
BB^{LD}	9 : 9	165758	335799	254.61	100.00
$MBCE^{Da}$	9 : 9	184945	233147	215.62	34.33
BB^{Da}	9 : 9	670569	511411	628.15	100.00

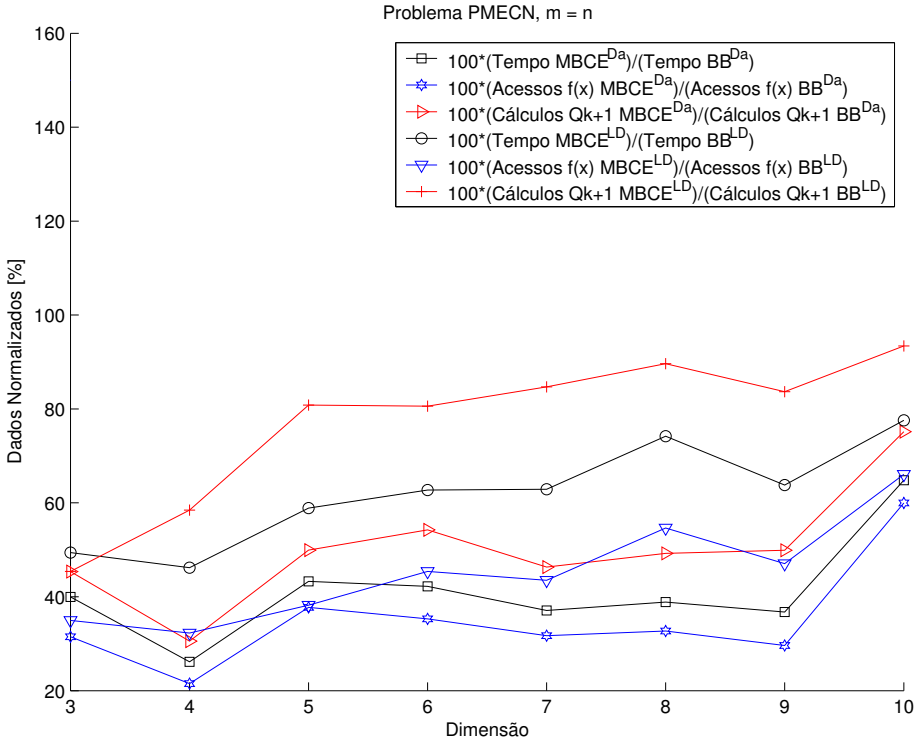
Tabela 11.10: Resultados comparativos dos métodos $MBCE$ e BB para um problema PMEC, dimensão 9.

Os resultados obtidos para todos os casos testados para o método $MBCE$ são exibidos nas figuras 11.6, 11.7, e 11.8, de forma análoga ao apresentado para os problemas quasi-convexos.

Os valores percentuais de *Acessos* $f(\mathbf{x})$ e *Tempo Total* foram superiores aos apresentados na figura 11.2 para o caso de problemas PMEQC e permaneceram quase sempre abaixo de 75%. Todavia a mesma tendência de redução dos ganhos à medida que a dimensão aumenta é observada. Esta tendência mantém aberto o questionamento do desempenho do método $MBCE$ para problemas de dimensão elevada.

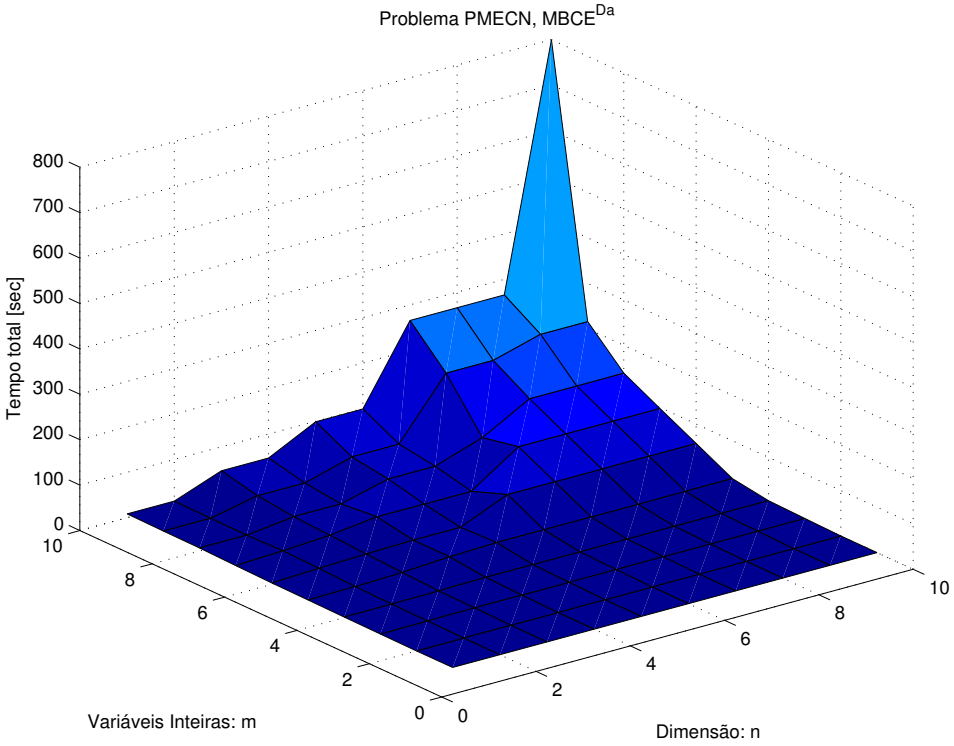


a) Problema misto discreto e contínuo convexo.

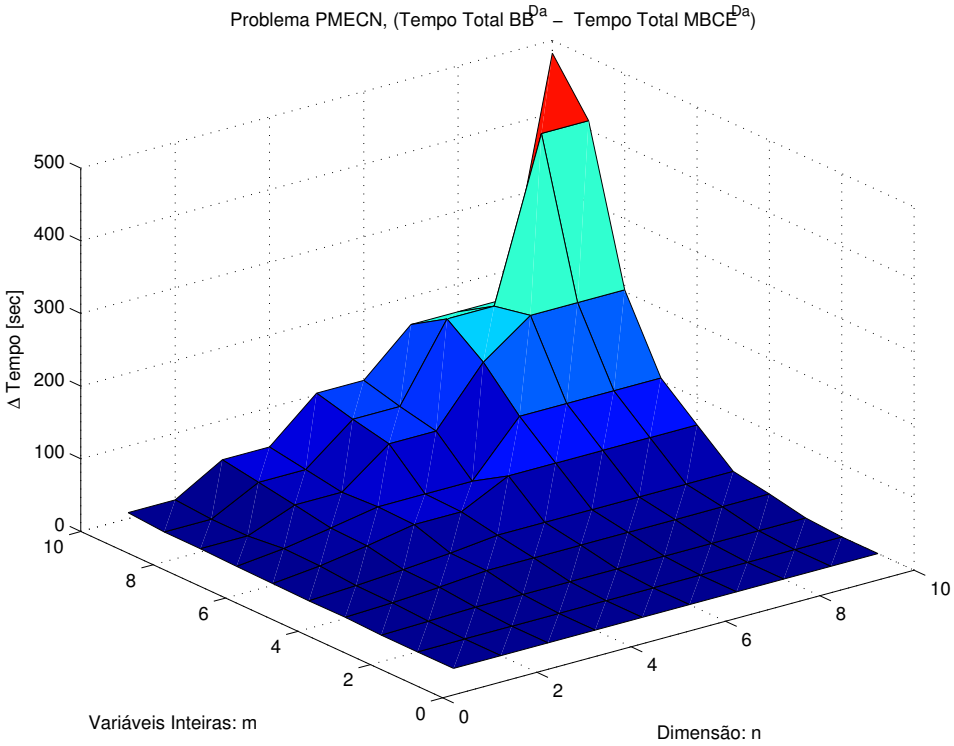


b) Problema discreto convexo.

Figura 11.6: Resultados percentuais de Tempo, Acessos $f(\mathbf{x})$ e Cálculos Q_{k+1}: $MBCE^{LD}$ x BB^{LD} x $MBCE^{Da}$ x BB^{DL} para problemas PMEC.

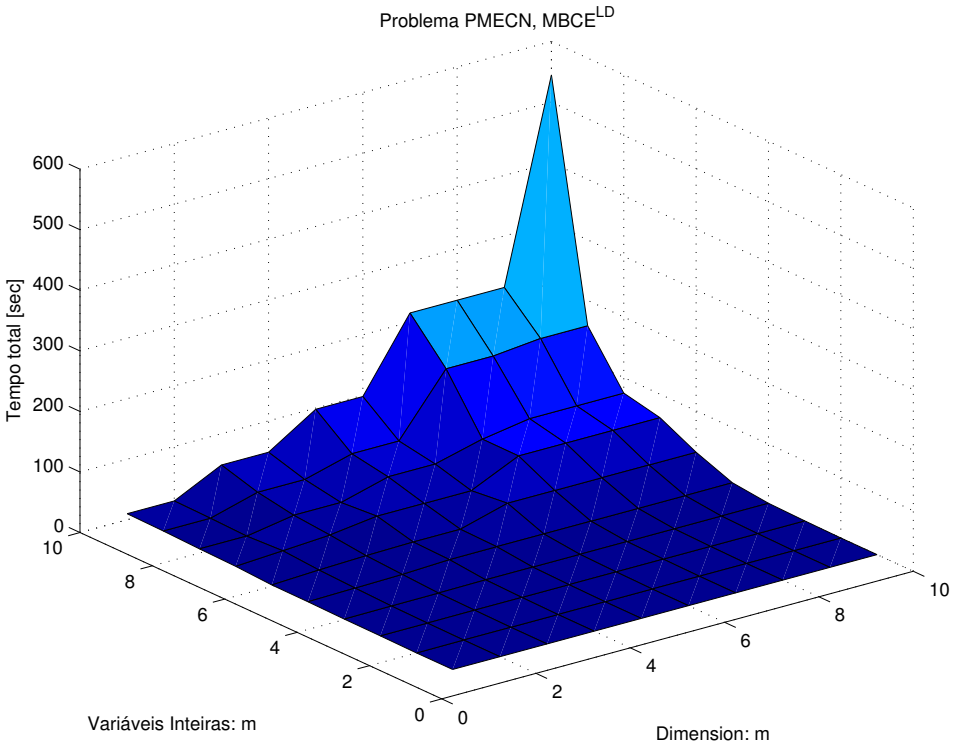


a) Tempo total para $MBCE^{Da}$.

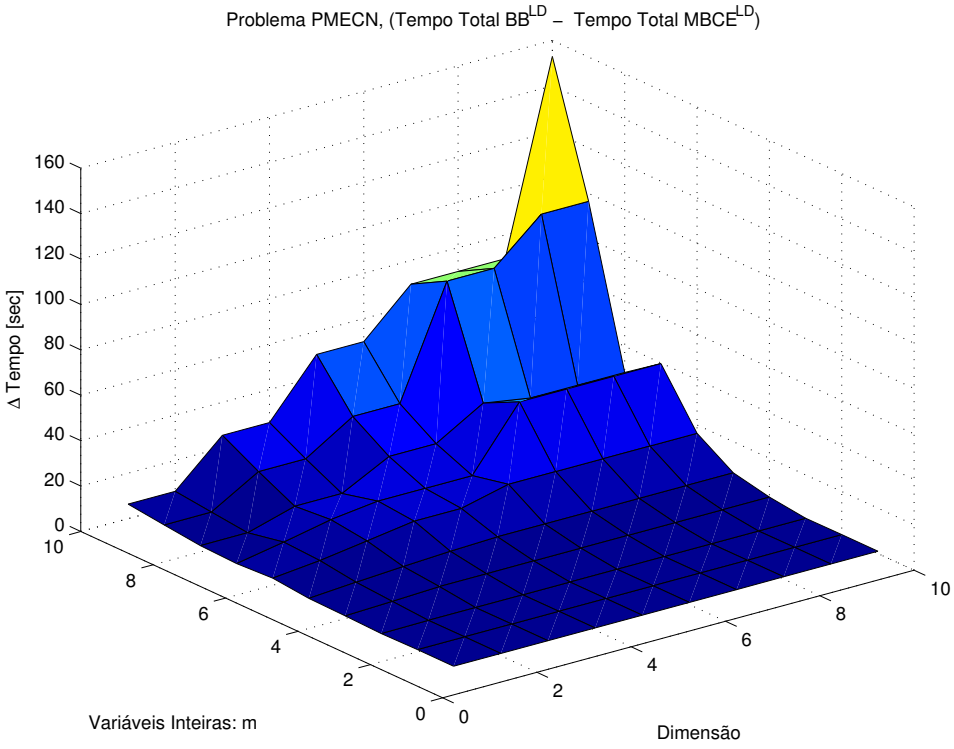


b) Δ Tempo total para $MBCE^{Da}$ e BB^{Da}.

Figura 11.7: Resultados dos problemas PMEC para BB^{Da} e $MBCE^{Da}$: número de variáveis inteiras z x dimensão n.



a) Tempo total para $MBCE^{LD}$.



b) Δ tempo total para $MBCE^{LD}$ e BB^{LD}.

Figura 11.8: Resultados dos problemas PMEC para BB^{LD} e $MBCE^{LD}$: número de variáveis inteiras z x dimensão n.

A figura 11.7 e a figura 11.8 ilustram o crescimento exponencial do esforço computacional para resolver os problemas PMEC com o aumento da dimensão e do número de variáveis inteiras, nas subfiguras 11.7.a e 11.8.a. Já as subfiguras 11.7.b e 11.8.b. mostram que, para os casos testados, a diferença de tempo de convergência total entre os algoritmos propostos e o algoritmo *Branch-and-Bound* também exibe a mesma tendência de comportamento exponencial para $\Delta Tempo$. Conseqüentemente, isto permite a interpretação de que o método de *Branch-and-Cut* aqui proposto possa ser considerado eficiente quando comparado com o método de *Branch-and-Bound* para a solução de problemas PMEC.

11.4 Avaliação da Utilização de Informações Históricas no Algoritmo $MBCE$

Tal como informado no início deste capítulo, o proposto algoritmo $MBCE$ utilizou o algoritmo $MPESC$ para a solução dos problemas obtidos por relaxação das variáveis inteiras. O algoritmo $MPESC$ também foi utilizado em conjunto com o algoritmo de referência de desempenho *Branch-and-Bound*. Desta feita, foi possível realizar a análise das proposições aqui apresentadas sem que o método que solucionava o problema relaxado influenciasse positivamente ou negativamente no resultado global dos testes realizados.

Entretanto, uma vez concluída a análise dos resultados obtidos para os algoritmos *Branch-and-Bound* e $MBCE$ associados ao algoritmo $MPESC$, inevitavelmente surge a pergunta de como seriam os resultados do algoritmo $MBCE$ quando da utilização do proposto algoritmo $HISPE$, apresentado na seção 6.2, para a solução dos problemas relaxados e para a obtenção de cortes eficientes. Esta pergunta é inevitável dados os bons resultados do algoritmo $HISPE$ obtidos na seção **??**, em função da utilização de informações históricas para a redução da região de interesse de busca que deve conter a solução.

Para criar as tabelas de resultados 11.11 e 11.12 desta análise, o problema PREQC representado por (**??**) foi parametrizado com os mesmos parâmetros gerados aleatoriamente propostos na seção **??**. Os resultados são apresentados para o algoritmo de referência de desempenho *Branch-and-Bound* e para o algoritmo proposto $MBCE$, ambos utilizando o algoritmo $MPESC$ para a solução dos problemas relaxados, bem como para a configuração do algoritmo proposto $MBCE_{HISPE}$, o qual utilizou o algoritmo $HISPE$ com a heurística do parâmetro de busca histórica $\tau = 3n$ para a solução dos problemas relaxados. As informações históricas também foram utilizadas pela configuração do algoritmo proposto $MBCE_{HISPE}$ para gerar cortes eficientes, vide o Teorema 8, de modo a reduzir o volume do elipsóide inicial dos problemas relaxados. Os problemas PREQC foram gerados

para as dimensões $n = \{8, 9\}$.

Algorithm	$ni : n$	Acessos $f(\mathbf{x})$
BB^{LD}	1 : 8	11219
$MBCE^{LD}$	1 : 8	5592
$MBCE^{LD}_{HISPE}$	1 : 8	1566
BB^{Da}	1 : 8	19244
$MBCE^{Da}$	1 : 8	9381
$MBCE^{Da}_{HISPE}$	1 : 8	9319
BB^{LD}	3 : 8	21080
$MBCE^{LD}$	3 : 8	15054
$MBCE^{LD}_{HISPE}$	3 : 8	4419
BB^{Da}	3 : 8	59908
$MBCE^{Da}$	3 : 8	37020
$MBCE^{Da}_{HISPE}$	3 : 8	38901
BB^{LD}	5 : 8	40849
$MBCE^{LD}$	5 : 8	38339
$MBCE^{LD}_{HISPE}$	5 : 8	14081
BB^{Da}	5 : 8	120961
$MBCE^{Da}$	5 : 8	108770
$MBCE^{Da}_{HISPE}$	5 : 8	95218
BB^{LD}	7 : 8	153107
$MBCE^{LD}$	7 : 8	135551
$MBCE^{LD}_{HISPE}$	7 : 8	98716
BB^{Da}	7 : 8	491778
$MBCE^{Da}$	7 : 8	381299
$MBCE^{Da}_{HISPE}$	7 : 8	372816
BB^{LD}	8 : 8	165385
$MBCE^{LD}$	8 : 8	148877
$MBCE^{LD}_{HISPE}$	8 : 8	108223
BB^{Da}	8 : 8	727982
$MBCE^{Da}$	8 : 8	494146
$MBCE^{Da}_{HISPE}$	8 : 8	488408

Tabela 11.11: Resultados comparativos dos métodos $MBCE_{HISPE}$, $MBCE$ e BB para um problema PMEQC, dimensão 8.

Algorithm	$ni : n$	Acessos $f(\mathbf{x})$
BB^{LD}	1 : 9	16055
$MBCE^{LD}$	1 : 9	8381
$MBCE^{LD}_{HISPE}$	1 : 9	1840
BB^{Da}	1 : 9	28109
$MBCE^{Da}$	1 : 9	13391
$MBCE^{Da}_{HISPE}$	1 : 9	12018
BB^{LD}	3 : 9	29815
$MBCE^{LD}$	3 : 9	20503
$MBCE^{LD}_{HISPE}$	3 : 9	5315
BB^{Da}	3 : 9	85021
$MBCE^{Da}$	3 : 9	44355
$MBCE^{Da}_{HISPE}$	3 : 9	47165
BB^{LD}	5 : 9	55388
$MBCE^{LD}$	5 : 9	48870
$MBCE^{LD}_{HISPE}$	5 : 9	15146
BB^{Da}	5 : 9	170528
$MBCE^{Da}$	5 : 9	128885
$MBCE^{Da}_{HISPE}$	5 : 9	143157
BB^{LD}	7 : 9	95121
$MBCE^{LD}$	7 : 9	90458
$MBCE^{LD}_{HISPE}$	7 : 9	39734
BB^{Da}	7 : 9	382187
$MBCE^{Da}$	7 : 9	335601
$MBCE^{Da}_{HISPE}$	7 : 9	328987
BB^{LD}	9 : 9	114402
$MBCE^{LD}$	9 : 9	112481
$MBCE^{LD}_{HISPE}$	9 : 9	49979
BB^{Da}	9 : 9	623761
$MBCE^{Da}$	9 : 9	537762
$MBCE^{Da}_{HISPE}$	9 : 9	487561

Tabela 11.12: Resultados comparativos dos métodos $MBCE_{HISPE}$, $MBCE$ e BB para um problema PMEQC, dimensão 9.

Os resultados exibidos nas tabelas 11.11 e 11.12 confirmam a expectativa de que a utilização do algoritmo proposto HISPE contribuiria para a aceleração do processo de convergência dos problemas relaxados e para a obtenção de cortes eficientes. Como pode ser observado nas tabelas 11.11 e 11.12, a configuração $MBCE_{HISPE}$ sempre encontrou as soluções utilizando um menor número de chamadas da função $f(\mathbf{x})$. Por serem os problemas utilizados de baixa dimensão e definidos por equações analíticas, não foi avaliado

nesta análise o parâmetro tempo total para convergência. Também não foi analisado o parâmetro cálculos Q_{k+1}, dado que necessariamente o algoritmo $HISPE$ executa mais cálculos do elipsóide E_{k+1} do que o algoritmo $MPESC$.

Dentre os resultados exibidos, destaca-se o melhor desempenho da codificação proposta por Land e Doig com a configuração $MBCE_{HISPE}$. Para alguns problemas, a configuração $MBCE_{HISPE}$ reduziu em até 80% o número de chamadas da função objetivo, em comparação à configuração $MBCE$ e ao algoritmo *Branch-and-Bound*. Conseqüentemente, é razoável supor que em problemas com elevado EAIP o desempenho da configuração $MBCE_{HISPE}$ tornar-se-á substancialmente superior ao desempenho do algoritmo *Branch-and-Bound*. Comparando-se a tabela 11.11 com a 11.12, outro ponto a se ressaltar é que a diferença percentual entre os resultados obtidos pela configuração $MBCE_{HISPE}$ e os resultados do algoritmo *Branch-and-Bound* aumentou com o aumento da dimensão do problema, para os problemas testados. Este fato também corrobora a idéia de que, em problemas com elevado EAIP, o desempenho da configuração $MBCE_{HISPE}$ tornar-se-á substancialmente superior ao desempenho do algoritmo *Branch-and-Bound*.

Por fim, a implementação proposta por Dakin com a configuração $MBCE_{HISPE}$, embora tenha apresentado melhores resultados que os obtidos com a configuração $MBCE$ e com o algoritmo *Branch-and-Bound*, não apresentou ganhos tão relevantes. A principal razão para este fato deve estar relacionada com a estrutura de criação de desigualdades do processo de *branch*. Diferentemente da proposição de Land e Doig que fixa a variável inteira em um valor e reduz a dimensão do problema relaxado a ser solucionado, a proposição de Dakin apenas acrescenta uma desigualdade com o valor arredondado de uma variável inteira. Assim, a precisão que define se uma variável é considerada inteira ou não pode determinar que uma variável gere duas restrições de desigualdades antes que o valor da solução para esta variável seja considerada inteira. Este fato certamente aumentará o número de acessos à função $f(\mathbf{x})$, tal como pode ser visto nas tabelas 11.11 e 11.12.

11.5 Conclusões do Capítulo

Neste Capítulo foram apresentados os resultados obtidos mediante a aplicação dos algoritmos que implementam família dos métodos *Elipsoidais* e dos algoritmos utilizados como referência para a avaliação do desempenho associados à solução dos problemas QCND com variáveis inteiras e mistas inteiras e contínuas.

Inicialmente, foi definida a classe de problemas PIELC para a análise da característica de convergência do algoritmo $MEHC$. Os resultados exibidos na seção 11.1 demonstraram

que, embora o algoritmo $MEHC$ seja capaz de resolver problemas PIELC, em função dos erros inerentes ao processo de geração de novas restrições o algoritmo apresentou problemas de convergência para dimensões maiores que $n = 4$. Por conseqüência, sua utilização prática se torna bem restrita.

Em continuidade aos testes, a classe de problemas PMEQC foi definida na seção 11.2.1. Os problemas PMEQC foram inicialmente utilizados para comparar os algoritmos propostos $MEEBB$, $MEEBC$ e $MBCE$ entre si, bem como compará-los ao algoritmo BB utilizado como referência de desempenho. A partir dos resultados obtidos, pôde ser observado que todos os métodos propostos são capazes de solucionar problemas PMEQC. Todavia, apenas o método $MBCE$ apresentou resultados superiores aos obtidos pelo algoritmo BB, tanto para problemas puramente discretos quanto para problemas com variáveis mistas inteiras e contínuas. Estes resultados superiores ocorreram para ambas as propostas de codificação do algoritmo BB. Um resultado a ser ressaltado nestes testes diz respeito ao comportamento exponencial apresentado pelo parâmetro de desempenho $\Delta Tempo = Tempo\ Total\ MBEC$ - $Tempo\ Total\ BB$ com o aumento da dimensão dos problemas testados.

Em seguida, a seção 11.2.2 definiu o problema químico de sintetização e apresentou os resultados obtidos para dois casos existentes na literatura. Mesmo sendo estes problemas referentes à classe de problemas binários, o algoritmo $MBCE$ apresentou os melhores resultados para o parâmetro tempo total para convergência.

Em continuidade, a seção 11.3 demonstrou os mesmos bons resultados apresentados anteriormente para os problemas PMEQC agora para os problemas PMECD. Novamente, o método $MBCE$ apresentou resultados superiores aos obtidos pelo algoritmo BB em todos os casos testados.

Por fim, a seção 11.4 analisou a influência da utilização do algoritmo $HISPE$ com a heurística do parâmetro de busca histórica $\tau = 3n$ para a solução dos problemas relaxados. Tal como era esperado, a configuração do algoritmo $MBCE_{HISPE}$, a qual utiliza o algoritmo $HISPE$, apresentou os melhores resultados para os problemas PMEQC testados, com destaque para a implementação do algoritmo pela proposição de Land e Doig. Também foi discutida a hipótese de que, em problemas com elevado EAIP, o desempenho da configuração $MBCE_{HISPE}$ tornar-se-á substancialmente superior ao desempenho do algoritmo *Branch-and-Bound*.

No capítulo seguinte serão indicadas as conclusões desta Tese, bem como recomendados trabalhos futuros para a continuidade desta linha de pesquisa.

Parte III

Conclusões Finais

Capítulo 12

Conclusões e Trabalhos Futuros

12.1 Conclusões

Esta Tese investigou a utilização da formulação do poliedro de busca e a reutilização de informações já calculadas no decorrer da convergência de um algoritmo baseado em *Elipsóide* para definir uma nova família de métodos de otimização. Esta família de métodos é aqui denominada de família dos métodos *Poliedro-Elipsoidais*. Os métodos *Poliedro-Elipsoidais* propõem o armazenamento de informações calculadas durante as iterações, a manipulação destas informações e a construção do poliedro de busca para acelerar o processo de convergência. Como conseqüência, embora obrigatoriamente ocorra o aumento do ECAO, os métodos *Poliedro-Elipsoidais* pretendem reduzir o tempo total para a obtenção da solução de um problema, por meio da redução do EAIP. Esta mesma proposição é aplicada no contexto dos problemas discretos e mistos para a construção de métodos baseados em *Elipsóide* capazes de solucionar tais problemas e de acelerar a obtenção da solução dos mesmos.

As principais conclusões aqui obtidas dizem respeito às sete teses formuladas na seção 1.3. Os capítulos 5 e 6 fundamentaram a família dos métodos *Poliedro-Elipsoidais* que sustentam a *Tese "i"* e a *Tese "ii"*, na medida que demonstram como construir regiões de interesse de busca de menor volume do que as propostas na literatura atual e como utilizar informações já calculadas para obter uma redução ainda maior desta região de interesse de busca.

Quanto à proposição descrita na *Tese "iii"*, a implementação dos algoritmos $MPESC$ e $HISPE$ descritos no capítulo 6.2 e a aplicação destes algoritmos na solução dos problemas descritos na seção 7.1 dão suporte à formulação proposta. Os resultados dos problemas escalares expressos nas tabelas **??** a **??** mostram a concreta diminuição da taxa de redução do volume e a drástica redução do número de acessos às funções-objetivo e

funções-restrição dos problemas testados. De fato, estes resultados acabaram por ampliar o conjunto de classes de problemas nos quais os métodos *Poliedro-Elipsoidais* poderão apresentar menor tempo total para convergência, em comparação aos algoritmos baseados em *Elipsóide* existentes na literatura. Isto porque, inicialmente, acreditava-se que os métodos *Poliedro-Elipsoidais* apresentariam desempenho superior apenas em problemas com elevado EAIP. Como decorrência da análise dos resultados obtidos no capítulo 7, os métodos *Poliedro-Elipsoidais* constituem uma alternativa eficaz à utilização dos algoritmos baseados em *Elipsóide* existentes na literatura, mesmo para problemas de baixa dimensão e de baixo EAIP.

A implementação dos algoritmos $MPESC$ e $HISPE$ descritos no capítulo 6.2 e a aplicação destes algoritmos na solução dos problemas descritos na seção 7.2 demonstram a validade formulação proposta na *Tese "iv"*. Dentre os resultados apresentados, deve-se ainda ressaltar três pontos. O primeiro ponto diz respeito à obtenção de soluções sempre não-dominadas pelos diferentes algoritmos propostos nos problemas PRVQC. O segundo ponto se refere ao baixo erro resultante da soma $|\frac{\nabla f_1(x^*)}{|\nabla f_1(x^*)|} + \frac{\nabla f_2(x^*)}{|\nabla f_2(x^*)|}|$ para os problemas PRVQI. E o terceiro ponto é definido pela obtenção pelo algoritmo $HISPE$ de soluções que dominaram as soluções obtidas com a utilização de LMI.

A comprovação da proposição definida na *Tese "v"*, se dá pela formalização no capítulo 4 da nova condição necessária e suficiente de infactibilidade e pela apresentação dos resultados obtidos na seção **??**. Particularmente, a tabela **??** exibe a capacidade do algoritmo $HISPE$ de identificar a factibilidade ou a infactibilidade de um problema.

Quanto à proposição descrita na *Tese "vi"*, esta é sustentada pelos métodos propostos nos capítulos 8, 9 e 10 e pelos resultados apresentados no capítulo 11. Faz-se também necessário registrar que em todos os testes realizados nas seções 11.2.1 e 11.2.2 as soluções obtidas com os métodos propostos foram correspondentes às soluções obtidas pelo método de referência $Branch-and-Bound$. Este fato permite afirmar que os métodos propostos sempre encontraram as soluções ótimas para os problemas testados. Cabe ainda a observação de que os métodos baseados em *Elipsóide* com enumeração resultantes do estudo da proposição descrita na *Tese "vi"* poderão possibilitar a definição de um algoritmo capaz de solucionar problemas vetoriais quasi-convexos não necessariamente diferenciáveis e com variáveis mistas discretas e contínuas.

Para a última proposição, os fundamentos do método $MBCE$ propostos na seção 10.3.2 e demonstrados no Teorema 8 (página 206) e no Teorema 9 (página 207) e no Lema 8 (página 207) formalizam a proposição descrita na *Tese "vii"*. Ainda mais, resultados apresentados no capítulo 11 permitem inferir que o algoritmo $MBCE$ constitui uma alternativa eficaz à utilização do algoritmo $Branch - and - Bound$ para a solução

de problemas escalares quasi-convexos e com variáveis mistas discretas e contínuas. Por fim, pode-se ainda considerar que, mesmo para problemas de dimensão elevada, o método $MBCE$ constituirá uma alternativa eficaz ao algoritmo $Branch-and-Bound$ por duas razões. Como caso geral, os resultados do algoritmo $MBCE$ tenderão a se aproximar assintoticamente dos resultados do algoritmo $Branch-and-Bound$, dado que o aumento do número de pontos a serem enumerados no interior do elipsóide fará com que o ECAO seja primordialmente definido pelo processo de ramificação. Como este processo é comum a ambos os algoritmos, o algoritmo $MBCE$ não apresentará resultados inferiores aos resultados do algoritmo $Branch-and-Bound$. Como caso particular, quando da existência de cortes eficientes o algoritmo $MBCE$ será capaz de reduzir consideravelmente o volume do elipsóide no qual os pontos discretos devem ser enumerados e isto produzirá resultados superiores aos resultados do algoritmo $Branch-and-Bound$. Este caso particular certamente ocorrerá para problemas com região factível muito menor que a região de busca definida pelo elipsóide inicial. Nesta classe de problemas os inúmeros pontos infactíveis proporcionarão diversos cortes eficientes para a redução do volume do elipsóide inicial, o que, conseqüentemente, ocasionará a aceleração da obtenção da solução ótima dos problemas.

12.2 Questões a Serem Respondidas por Trabalhos Futuros

Embora esta Tese tenha buscado cobrir a maior parte das questões relacionadas às proposições descritas pelas *Teses "i"* a *"vii"*, diversas questões ainda permanecem para serem respondidas por trabalhos futuros. Dentre elas se destacam:

a) *Existirá uma dimensão limite para a aplicação dos algoritmos da família de métodos Poliedro-Elipsoidais em problemas práticos?*

Embora teoricamente os métodos baseados em *Elipsóide* possam ser aplicados a problemas de qualquer dimensão, a eficiência destes métodos claramente reduz com o aumento da dimensão do problema. Mesmo restringindo a região de interesse de busca, em função da utilização do poliedro de busca, a aplicação prática dos métodos *Poliedro-Elipsoidais* em problemas de dimensão elevada permanece como uma questão ainda não respondida.

b) *Será possível sempre obter menores tempos totais de convergência em problemas escalares quasi-convexos e não necessariamente diferenciáveis e com variáveis mistas*

binárias e contínuas, em comparação aos tempos obtidos pelo algoritmo Branch-and-Bound?

Mesmo sendo capazes de resolver problemas binários, o método $MEBC$ acaba por ter seu desempenho prejudicado pelo fato de existirem poucos cortes eficientes gerados durante o processo de convergência. Logo, apesar dos resultados apresentados na seção 11.2.2, a garantia de obtenção de melhores resultados em comparação com os resultados do algoritmo Branch-and-Bound permanece uma questão em aberto.

c) *Será possível definir um algoritmo baseado em Elipsóide capaz de solucionar problemas vetoriais quasi-convexos e com variáveis mistas discretas e contínuas com garantia de convergência global?*

A idéia de fundamentar um algoritmo baseado em *Elipsóide* capaz de solucionar problemas vetoriais quasi-convexos e com variáveis mistas discretas e contínuas parece ser uma extensão quase imediata da união das proposições apresentadas nos capítulos 5 e 10. Todavia, a garantia de convergência global ainda precisa ser demonstrada.

d) *Será possível definir o valor do parâmetro τ que produz o menor tempo total de convergência para um problema com base apenas nas informações deste problema, tais como dimensão e tempo para avaliação das funções-objetivo e funções-restrição?*

Para os problemas testados na seção 7.1, foi realizado o estudo da influência do parâmetro τ no tempo total para a obtenção da solução. Embora este estudo tenha proposto a heurística de se definir $\tau \approx 3n$, a definição do valor de τ baseado apenas nos parâmetros que definem o EAIP seria mais precisa. Assim, esta permanece sendo uma questão a ser respondida.

e) *Será possível aplicar o conceito do poliedro de busca a métodos de Pontos Interiores?*

Os métodos de *Pontos Interiores* e suas variações têm sido amplamente estudados nos últimos anos. Dado que estes métodos apresentam similaridades com alguns conceitos dos métodos *Elipsoidais*, é razoável supor que o conceito do poliedro de busca poderia ser adaptado para este método. Isto cria mais uma questão a ser respondida.

Referências Bibliográficas

Adán, M. & Novo, V. (2003). Efficient and weak efficient points in vector optimization with generalized cone convexity, *Applied Mathematics Letters* **16**(2): 221–225.

Agmon, S. (1954). The relaxation method for linear inequalities, *Canadian Journal of Mathematics* **6**: 382–392.

Balas, E. (1965). An additive algorithm for solving linear programs with zero-one variables, *Operation Research* **13**: 517–546.

Balas, E. (1969). The intersection cut - a new cutting plane for integer programming, *Technical Report Management Sciences Research Report No. 187*, Carnegie-Mellon University.

Bazaraa, M. S. & Shetty, C. M. (1979). *Nonlinear Programming*, John Wiley.

Ben-Tal, A. & Nemirovski, A. (2001). *Lectures on Modern Convex Optimization - Analysis, Algorithms and Engineering Applications*, Journal of Society for Industrial and Applied Mathematics.

Benson, H. P. (1984). Optimization over the efficient set, *Journal of Mathematical Analysis and Applications* **98**: 562–580.

Benson, H. P. & Aksoy, Y. (1991). Using efficient feasible directions in interactive multiple objective linear programming, *Operations Research Letters* **10**: 203–209.

Bierman, G. J. (1976). Measurement updating using the U-D factorization, *Automatica* **12**: 375–382.

Bierman, G. J. (1977). *Factorization methods for discrete sequential estimation*, Mathematics in Science and Engineering. Vol. 128. New York - San Francisco - London: Academic Press. XVI, 241 p. .

Bland, R. G., Goldfarb, D. & Todd, M. J. (1981). The ellipsoid method: A survey, *Operation Research* **29**: 1039–1091.

Boggs, P. T. (1995). *Sequential Quadratic Programming*, Actica Numerica.

Boyd, S. & Vandenberghe, L. (2004). *Convex Optimization*, Cambridge University Press.

Bradley, S., Hax, A. & Magnati, T. (1977). *Applied Mathematical Programming*, Addison-Wesley Publishing Company Addison-Wesley.

Braga, A. P., Carvalho, A. P. L. & Ludemir, T. B. (2000). *Redes Neurais Artificiais:Teoria e Aplicações*, LTC - Livros Técnicos e Científicos Editora S.A.

Caminhas, W. M., Pereira, G. A. & Tavares, F. A. C. (1998). Identificação de sistemas dinâmicos: Abordagem baseada em neurônio nebuloso, *Anais... Rio de Janeiro, Brasil: V Simpósio Brasileiro de Redes Neurais. CD-ROM* .

Chankong, V. & Haimes, Y. Y. (1982). On the characterization of noninferior solutions of the vector optimization problem, *Automatica* **18**(6): 697–707.

Chankong, V. & Haimes, Y. Y. (1984). *Multiobjective Decision Making: Theory and Methodology*, Elsevier.

Chvátal, V. (1973). Edmonds polytopes and a hierarchy of combinatorial problems.

Conn, A. R., Gould, N. I. M. G. & Toint, P. L. (1987). *Trust-Region Methods*, Society for Industrial Mathematics.

Cordier, C., Marchand, H., Laundy, R. & Wolsey, L. A. (1999). *bc-opt*: A branch-and-cut code for mixed integer programs, *Mathematical Programming* **86**: 335–353.

Crowder, H., Johnson, E. L. & Padberg, M. (1983). Solving large-scale zero-one linear programming problems, *Operation Research* **31**: 803–834.

Dakin, R. (1965). A tree-search algorithm for mixed integer programming problems, *The Computer Journal* **8**: 250–255.

Dantzig, G. B. (2002). Linear programming, *Operation Research* **50 (1): 42**. This article originally appeared in History of Mathematical Programming: A Collection of Personal Reminiscences, 1991.

Dias, W. F. (2003). *Algoritmos Cone-Elipsoidais para Geração de Soluções Eficientes em Otimização Vetorial*, Programa de Pós Graduação em Engenharia Elétrica, UFMG. Tese de Doutorado.

DLPO (2005). O dicionário da língua portuguesa on-line, Internet. Disponível em: <http://www.priberam.pt/dlpo>. Acesso em: 20 maio 2005.

Duran, M. A. & Grossmann, I. E. (1986). An outer-approximation algorithm for a class of mixed integer nonlinear programs, *Mathematical Programming* **36**: 307–339.

Ech-Cherif, A. & Ecker, J. (1984). A class of rank-two ellipsoid algorithms for convex programming, *Mathematical Programming* **29**: 187–202.

Ehrgott, M. (2000). *Multicriteria Optimization*, Berlin: Springer-Verlag.

Floudas, C. A. (n.d.). *Nonlinear and Mixed-Integer Optimization - Fundamentals and Applications*, Oxford University Press.

Garcia, A. L. (1989). *Probability and Random Process for Electrical Engineering*, Massachusetts: Addison-Wesley Publishing Company Addison-Wesley.

Gill, P. E., Murray, W. & Saunders, M. A. (1975). Methods for computing and modifying the ldl factors of a matrix, *Mathematics of Computation* **29**: 1051–1077.

Glover, F. (1973). Convexity cut and cut search, *Operation Research* **21**: 123–134.

Goffin, J.-L. & Vial, J.-P. (1999). A two-cut approach in the analytic center cutting plane method, *Mathematical Methods of Operations Research* **49**: 149–169.

Goldberg, D. E. (1989). Genetic algorithms in search, optimization and machine learning. Reading MA: Addison-Wesley.

Gomory, R. E. (1958). Outline of an algorithm for integer solutions to linear programs, *Bulletin of the American Mathematical Society* **64**: 275–278.

Gomory, R. E. (1960). Solving linear programming problems in integers, *Processing Symposium and Applied Mathematics* **10**: 211–215.

Gu, Z., Nemhauser, G. L. & Savelsbergh, M. W. P. (1995). Lifted cover inequalities for 0-1 integer programs, *School of Industrial and Systems Engineering, Georgia Institute of Technology* .

Hartman, H. L. & Mutmansky, J. M. (2002). *Introductory Mining Engineering*, New Jersey: John Wiley and Sons.

Heing, M. I. (1982). Proper efficiency with respect to cones, *Journal of Optimization Theory and Applications* **36**: 387–407.

Ho, Y. & Pepyne, D. (2002). Simple explanation of the no-free-lunch theorem and its implications, *Journal of Optimization Theory and Applications* **115**(3): 549–570.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor, Mich.

Iudin, D. B. & Nemirovskii, A. S. (1976). Informal complexity and effective methods of solution for convex extremal problems, *Translation of Russian and East European Mathematical Economics* **13**: 25–45.

Jones, D. R., Schonlau, M. & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions, *Journal of Global Optimization* **13**(4): 455–492.

Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming, *Kibernetica Sb.* **26**: 84–112.

Kelley, J. E. (1960). The cutting plane method for solving convex programs, *Journal of the Society for Industrial and Applied Mathematics* **8**: 703–712.

Khachiyan, L. (1979). A polynomial algorithm in linear programming, *Doklady Akademii Nauk SSSR* **244**: 1093–1096.

Kim, S., Kim, D. & Chang, K.-N. (1994). Using two successive subgradients in the ellipsoid method for nonlinear programming, *Journal of Optimization Theory and Applications* **82**(3): 543–554.

Kiwiel, K. C. (1995). Block-iterative surrogate projection methods for convex feasibility problems, *Linear Algebra and Its Applications* **215**: 225–259.

Kou, S. (2003). *Welding Metallurgy*, New Jersey: John Wiley and Sons.

Land, A. & Doig, A. (1960). An automatic method of solving discrete programming problems, *Econometrica* **28**: 497–520.

Levin, A. (1965). On an algorithm for the minimization of convex functions, *Doklady Soviet Mathematics* **6**: 268–290.

Li, D. & Sun, X. (n.d.). *Nonlinear Integer Programming*, Springer Science+Business Media, LLC.

Liao, A. & Todd, M. J. (1996). Solving LP problems via weighted centers, *SIAM Journal on Optimization* **6**(4): 933–960.

Linderoth, J. T. & Savelsbergh, M. W. P. (1997). A computational study of search strategies for mixed integer programming, *Report LEC 97-12, Georgia Institute of Technology* .

Little, J. D. C., Murty, K. G., Sweeney, D. W. & Kare, C. (1963). An algorithm for traveling salesman problem, *Operation Research* **11**: 972–989.

Liu, J. S. (2001). *Monte Carlo Strategies in Scientific Computing*, Springer.

Luc, D. T. (n.d.). *Theory of Vector Optimization*, Springer-Verlag.

Luenberger, D. G. (1984). *Linear and Nonlinear Programming*, Addison-Wesley.

Luenberger, D. G. (n.d.). *Optimization by Vector Space Methods*, John Wiley and Sons, Inc.

Marchand, H., Martin, A., Weismantel, R. & Wosley, L. (2002). Cutting planes in integer and mixed integer programming, *Discrete Applied Mathematics* **123**: 397–446.

Miettinen, K. (n.d.). *Nonlinear Multiobjective Optimization*, Kluwer Academic Publishers.

Miettinen, K. & Mäkelä, M. M. (2001). One cone characterizations of weak, proper and Pareto optimality in multiobjective optimization, *Mathematical Methods of Operations Research* **53**: 233–245.

Motzkin, T. & Schoenberg, I. (1954). The relaxation method for linear inequalities, *Canadian Journal of Mathematics* **6**: 393–404.

Moura, A. S. J. & Leal, S. B. (2005). Sistema de otimização, estudo conceitual sobre a sua adoção na indústria baseado em retorno sobre investimentos e baixa imobilização de ativos, *Anais... São Paulo, Brasil: 5º Congresso Internacional de Automação, Sistemas e Instrumentação - ISA. CD-ROM* .

Moura, A. S. J., Smith, J. M. & Takahashi, R. (2007). An ellipsoidal branch-and-cut method for solving mixed integer quasi-convex problems, *Annals... University of Montreal, Montreal, Canada: MIP 2007 conference poster section. CD-ROM* .

Moura, A. S. J. & Takahashi, R. (2007). An ellipsoid based method for solving quasi-convex vector optimization problems, *Annals... MacMaster University, Hamilton, Canada: ICCOPT 2007 and MOPTA 2007 conference poster section. CD-ROM* .

Narenda, K. S. & Parthasarathy, K. (1990). Identification and control of dynamical systems using neural network, *IEEE Transactions on Neural Networks* **1**: 4–27.

Nemhauser, G. L. & Wolsey, L. A. (n.d.). *Integer and Combinatorial Optimization*, John Wiley and Sons, Inc.

Nesterov, Y., Todd, M. J. & Ye, Y. (1999). Infeasible-start primal-dual methods and infeasibility detectors for nonlinear programming problems, *Mathematical Programming* **84**: 227–267.

Newman, D. (1965). Location of the maximum on unimodal surfaces, *Journal of the Association for Computing Machinery* **12**: 395–398.

Padberg, M. (2005). Classical cuts for mixed-integer programming and branch-and-cut, *Annals of Operations Research* **139**(1): 321–352.

Papadimitriou, C. H. & Steiglitz, K. (n.d.). *Combinatorial Optimization - Algorithms and Complexity*, Pretince-Hall, Inc.

Parker., R. G. & Rardin, R. L. (1988). *Discrete Optimization*, Academic Press.

Reed-Hill/Abbaschian, R. & Abbaschian, R. (1991). *Physical Metallurgy Principles (The Pws-Kent Series in Engineering)*, New Jersey: Thomson-Engineering.

Roberts, A. W. & Varberg, D. E. (n.d.). *Convex Functions*, Academic Press.

Rockafellar, R. T. (n.d.). *The Theory of Subgradients and Its Applications to Problems of Optimization*, Heldermann Verlag.

Salkin, H. M. & Mathur, K. (n.d.). *Foundations of Integer Programming*, Elsevier Science Publishing Co., Inc.

Savelsbergh, M. W. P. & Nemhauser, G. L. (1993). Functional description of MINTO, a mixed integer optimizer, *Report COC-91-03A, Georgia Institute of Technology, Atlanta, Georgia* .

Sergienko, I. V., Lebedeva, T. T. & Semenova, N. V. (2000). Existence of solutions in vector optimization problems, *Cybernetics and Systems Analysis* **36**(6): 823–828.

Shor, N. & Gershovich, V. (1979). Family of algorithms for solving convex programming problems, *Cybernetics* **15**: 502–508.

Shor, N. Z. (1964). *On the structure of algorithms for the numerical solution of optimal plannig and design problems*, Master's thesis, Cybernetics Institute, Academy of Sciences of Ukrainian SSR, Kiev.

Shor, N. Z. (1970a). Convergence rate of the gradient descent method with dilatation space, *Cybernetics* **6**: 102–108.

Shor, N. Z. (1970b). Utilization of the operation of space dilatation in the minimization of convex functions, *Cybernetics* **6**: 7–15.

Shor, N. Z. (1977). Cut-off method with space extension in convex programming problems, *Cybernetics* **13**: 94–96.

Simmons, D. (1972). *Linear Programming for Operations Research*, New Jersey: Prentice Hall.

Smith, J. M. (n.d.). Evacuation networks.
*citeseer.ist.psu.edu/smith98evacuation.html

Soleimani-damaneh, M. (2007). Characterization of nonsmooth quasiconvex and pseudoconvex functions, *Journal of Mathematical Analysis and Applications* **330**: 1387–1392.

Takahashi, R. H. C. (2003). *Otimização Escalar e Vetorial, Notas de Aula de Curso de Verão*, UFMG.

Tawarmalani, M. & Sahinidis, N. V. (2001). Semidefinite relaxations of fractional programs via novel convexification techniques, *Journal of Global Optimization* **20**(2): 133–154.

Todd, M. J. (1982). On minimum volume ellipsoids containing part of a given ellipsoid, *Mathematics of Operations Research* **7**: 253–261.

Topkis, D. M. & Veinott, A. F. (1967). On the convergence of some feasible direction algorithms for nonlinear programming, *SIAM Journal of Control* **5**: 268–279.

Tui, H. (1964). Concave programming under linear constraints, *Doklady Soviet Mathematics* **5**: 1437–1440.

Verhoeven, J. D. (1975). *Fundamentals of Physical Metallurgy*, New Jersey: John Wiley and Sons.

Westerlund, T. & Pörn, R. (2002). Solving pseudo-convex mixed integer optimization problems by cutting plane techniques, *Optimization and Engineering* **3**: 253–280.

Wolsey, L. A. (n.d.). *Integer Programming*, John Wiley and Sons, Inc.

Young, R. (1968). New cuts for a special class of 0-1 integer programs, *Technical Report Department of Economics and Mathematical Sciences Research Report*, Rice University.

Apêndice A

A função de fatorização $ud(M)$ utilizada para a estabilização numérica do cálculo da matriz Q_{k+1} é apresentada algoritmo abaixo:

Algorithm 19 Fatorização UDU^T

Input: Uma matriz M de ordem n.

Output: Uma matriz triandular U de ordem n. Uma matriz diagonal D de ordem n.

```
 1: function UD(M)
 2:     D(n,n) ← M(n,n)
 3:     U(n,n) ← 1
 4:     j ← n − 1
 5:     repeat
 6:         U(j,n) ← M(j,n)/D(n,n)
 7:         j ← j − 1
 8:     until j ≤ 1
 9:     j ← n − 1
10:     repeat
11:         D(j,j) ← M(j,j)
12:         k ← n − 1
13:         repeat
14:             D(j,j) ← D(j,j) − D(k,k) ∗ U(j,k)^2
15:             k ← k − 1
16:         until k ≤ 1
17:         U(j,j) ← 1
18:         i ← j − 1
19:         repeat
20:             U(i,j) ← M(i,j)
21:             k ← j + 1
22:             repeat
23:                 U(i,j) ← U(i,j) − D(k,k) ∗ U(i,k) ∗ U(j,k)
24:                 k ← k + 1
25:             until k ≤ n
26:             U(i,j)U(i,j)/D(j,j)
27:             i ← i − 1
28:         until i ≤ 1
29:         j ← j − 1
30:     until  j ≤ 1
31: end function
```