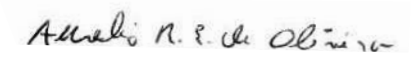
SOLUÇÃO DE PROBLEMAS DE PROGRAMAÇÃO LINEAR COM ALTA PRECISÃO ATRAVÉS DO SISTEMA LINEAR ESTÁVEL

Este exemplar corresponde à redação final da tese devidamente corrigida e defendida por Danilo Elias de Oliveira e aprovada pela comissão julgadora.

Campinas, 21 de Junho de 2010.



Prof. Dr:. Aurelio Ribeiro Leite de Oliveira
Orientador

Prof. Dra:. Maria de Los Angeles González-Lima
Co-orientadora

Banca Examinadora:

1 – Aurelio Ribeiro Leite de Oliveira
2 – Roberto Andreani
3 – Henrique Pacca Loureiro Luna
4 – Marcos Nereu Arenales
5 – Paulo Morelato França

Tese apresentada ao Instituto de Matemática, Estatística e Computação Científica, UNICAMP, como requisito parcial para obtenção do Título de DOUTOR em Matemática Aplicada

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP

Bibliotecária: Silvania Renata de Jesus Ribeiro Cirilo - CRB8 /6592

Oliveira, Danilo Elias de

Ol4s Solução de problemas de programação linear com alta precisão através do sistema linear estável/Danilo Elias de Oliveira-- Campinas, [S.P. : s.n.], 2010.

Orientador : Aurélio Ribeiro Leite Oliveira

Tese (doutorado) - Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Computação Científica.

1. Métodos de pontos interiores. 2. Programação linear. 3. Otimização matemática.. I. Oliveira, Aurélio Ribeiro Leite. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

Título em inglês: Solving linear programming problems with high accuracy through the stable linear system.

Palavras-chave em inglês (Keywords): 1. Interior point methods. 2. Linear programming. 3. Mathematical optimization.

Área de concentração: Otimização

Titulação: Doutor em Matemática Aplicada

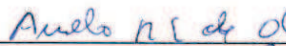
Banca examinadora: Prof. Dr. Aurelio Ribeiro Leite de Oliveira - (IMECC-UNICAMP)
Prof. Dr. Roberto Andreani - (IMECC-UNICAMP)
Prof. Dr. Henrique Pacca Loureiro Luna - (UFAL)
Prof. Dr. Marcos Nereu Arenales - (USP)
Prof. Dr. Paulo Morelato França - (UNESP)

Data da defesa: 21/06/2010

Programa de Pós-Graduação: Matemática Aplicada

Tese de Doutorado defendida em 21 de junho de 2010 e aprovada

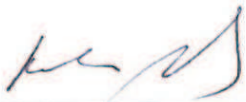
Pela Banca Examinadora composta pelos Profs. Drs.



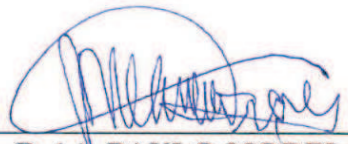
Prof(a). Dr(a). AURELIO RIBEIRO LEITE DE OLIVEIRA

Prof(a). Dr(a). ROBERTO ANDREANI

Prof(a). Dr(a). HENRIQUE PACCA LOUREIRO LUNA



Prof(a). Dr(a). MARCOS NEREU ARENALES



Prof(a). Dr(a). PAULO MORELATO FRANÇA

Aos meus pais, Gilberto e Júlia,
e ao meu irmão, Vilmar,
ofereço.

À minha esposa, Cristiane Ribeiro dos Santos,
por me apoiar e incentivar a realizar o sonho do
doutorado, para isso deixando de lado seus sonhos,
dedico.

Agradecimentos

Primeiramente, devo agradecer ao Divino Pai Eterno por ter me dado a vida e oportunidade de poder conviver ao lado de pessoas maravilhosas. Por me iluminar e abençoar com a sabedoria necessária para as provas e para os estudos desta tese.

Aos meus pais e irmão que sempre me apoiaram e me incentivaram a cursar o doutorado. Além disso, por todo amor, carinho e atenção que tiveram comigo.

À minha esposa Cristiane, querida mocinha, por todo carinho, atenção, paciência e compreensão mesmo nas horas em que mais precisei.

Um agradecimento especial aos professores Aurelio Ribeiro Leite Oliveira e María González-Lima pela dedicação, paciência, amizade e, também, pelos ensinamentos transmitidos, tanto na área profissional quanto na área pessoal, ao longo de minha orientação.

Agradeço aos professores do Instituto de Matemática, Estatística e Computação Científica (IMECC) da UNICAMP, pela experiência e conceitos transmitidos, que contribuíram para a conclusão deste trabalho. Aos funcionários da seção de pós-graduação do IMECC pelo tratamento sempre solícito e pela eficiência no atendimento.

Aos meus amigos de pós-graduação que sempre me apoiaram e incentivaram. Principalmente aos meus amigos Paulo e Cristiane que me ajudaram desde minha chegada na Unicamp. E, também, às minhas amigas de sala Marina e Ana Camila.

Aos meus tios e primos por me acolherem em suas casas em Americana e em Campinas.

À minha querida Uly.

À todos àqueles que não foram citados, mas que de alguma forma contribuíram para a realização deste trabalho.

“ O único homem que está isento de erros,
é aquele que não arrisca acertar. ”
Albert Einstein

Resumo

Apresentamos neste trabalho um método robusto e eficiente para a resolução do sistema linear estável para problemas de programação linear com variáveis canalizadas. O sistema linear estável é uma abordagem que pode ser utilizada para resolver os sistemas lineares que surgem em métodos de pontos interiores na programação linear. Adicionalmente, fazemos uma comparação entre o método apresentado e o sistema de equações normais resolvido por um método direto e, também, por um método iterativo, nas iterações do método preditor-corretor. Essa comparação é realizada pela implementação do método em linguagem C e integrada a uma implementação do método preditor-corretor já existente. Apresentamos também, um estudo numérico e comparativo sobre perturbação para problemas degenerados. Para os testes computacionais foram utilizados os problemas da Netlib.

Palavras-chave: sistema linear estável; métodos de pontos interiores; sistema de equações normais; pré-condicionador separador; programação linear.

Abstract

We present in this thesis a robust and efficient method for solving the stable linear system for linear programming problems with bounded variables. The stable linear system is an approach that can be used to solve linear systems arising in interior point methods in linear programming. Additionally, we perform a comparison among the presented method and the system of normal equations solved by direct and iterative methods in the predictor-corrector version. To perform this comparison, we have implemented the method in the C language and integrated it in an implementation of the predictor-corrector version. We also have developed perturbations for the degenerated case. For the computational experiments we have used the Netlib set of test problems.

Keywords: stable linear system; interior point method; normal equation system; splitting preconditioner; linear programming.

Sumário

Capítulo 1

Introdução

Um problema de programação linear é um problema de minimizar ou maximizar uma função linear, sujeito a restrições lineares de igualdade e desigualdade. Este problema teve sua formulação realizada nos anos de 1930 e 1940, veja por exemplo [14]. Na metade da década de 1940, Dantzing apresentou o método simplex para programação linear. O método simplex tornou-se uma ferramenta fundamental em programação linear, desde então. No entanto, este método possui convergência exponencial no número de iterações, para problemas especialmente construídos [33].

Em 1979, Khachian [32] apresenta um método para programação linear com complexidade polinomial, o método das elipsóides. Este método não mostrou-se prático para resolver problemas de programação linear. Em 1984, Karmarkar [31] apresenta um outro método alternativo para programação linear, também com complexidade polinomial. A publicação deste método iniciou uma nova linha de pesquisa conhecida como métodos de pontos interiores, e uma década depois os métodos primais-duais surgiram como os métodos mais importantes e úteis desta classe de métodos [64].

Nos métodos de pontos interiores, os sistemas lineares se tornam muito mal condicionados à medida em que o ponto se aproxima de uma solução. Esses sistemas podem ser escritos de uma forma simétrica, conhecida como sistema aumentado. Entre as implementações que utilizam métodos de pontos interiores para problemas lineares, a grande maioria reduz o sistema aumentado em um sistema menor e definido positivo, chamado sistema de equações normais e as direções são obtidas a partir da solução desse sistema. No entanto, esse tipo de sistema pode apresentar uma matriz densa mesmo quando a matriz de restrições é esparsa. Além disso, o sistema é ainda mais mal-condicionado que o sistema aumentado quando o ponto está próximo

de uma solução, especialmente para os problemas degenerados.

Muitas implementações de métodos de pontos interiores utilizam a decomposição esparsa de Cholesky para resolver o sistema de equações normais [2, 13, 25, 41, 65]. No entanto, em alguns casos, métodos diretos se tornam muito caros, tanto no armazenamento quanto no tempo de processamento. Assim, os métodos iterativos se tornam mais adequados. Porém, devido ao fato da matriz ser mal-condicionada, os métodos iterativos só serão bem sucedidos com a escolha de um pré-condicionador eficiente. Alguns pré-condicionadores podem ser encontrados em [1, 45, 54, 62]. Muitos deles, são baseados na decomposição incompleta de Cholesky para matrizes definidas positivas. Essa classe de pré-condicionadores é eficiente nas primeiras iterações, no entanto, na medida que o problema converge para uma solução eles se tornam ineficientes.

Em [7] é proposto um método híbrido para resolver o sistema de equações normais. Na primeira fase, durante as primeiras iterações, o método dos gradientes conjugados é pré-condicionado utilizando uma decomposição incompleta chamada decomposição controlada de Cholesky (CCF), proposto por Campos [10] e na segunda fase é utilizando o pré-condicionador separador desenvolvido por Oliveira [50], e apresentado por Oliveira e Sorensen em [51]. O pré-condicionador CCF começa a perder eficiência quando o sistema está altamente mal-condicionado, o que significa que o pré-condicionador separador, especialmente desenvolvido para as últimas iterações, deverá funcionar melhor.

A classe de pré-condicionadores separadores foi desenvolvida especialmente para resolver o sistema aumentado. Uma característica importante dessa classe é a opção de reduzir o sistema indefinido pré-condicionado em um definido positivo, como as equações normais, permitindo o uso do método dos gradientes conjugados. Como essa classe foi desenvolvida para as últimas iterações de métodos de pontos interiores, um pré-condicionador diagonal foi utilizado em [51] para as iterações iniciais. A melhoria apresentada em [7] utiliza um pré-condicionador eficiente em substituição ao diagonal.

Uma outra forma para calcular as direções de um método de pontos interiores foi proposta por González-Lima, Wei e Wolkowicz em [28]. Esse método foi estendido para o caso de programação quadrática convexa, por Dominguez e González-Lima [17]. Em [17] a abordagem sistema linear estável é um sistema indefinido do mesmo tamanho que o sistema aumentado, mas com a vantagem de não conter nenhuma matriz inversa das matrizes de variáveis. Assim, esse sistema não se torna tão mal-condicionado, pois a matriz não possui autovalores grandes, como em outras abordagens. De fato, em [28], é mostrado que o número de condição da matriz é limitado quando

o problema primal e o problema dual não são degenerados. Além disso, este sistema não é afetado por colunas densas na matriz de restrições, ao contrário do sistema de equações normais. Esse sistema linear é resolvido em [28] por métodos diretos e utilizando-se o LSQR, uma generalização do método dos gradientes conjugados desenvolvido por Paige e Saunders [53]. São utilizados dois pré-condicionadores, um diagonal e outro baseado na decomposição incompleta de Cholesky.

Apesar de o método apresentado em [28] se mostrar mais robusto que a abordagem das equações normais para problemas grandes, esparsos e bem condicionados, ele se mostrou mais lento para os problemas degenerados da NETLIB [18, 48]. Uma descrição desses problemas pode ser encontrada em [6].

O primeiro objetivo de nosso trabalho consiste em desenvolver um método eficiente para a resolução da abordagem do sistema linear estável, como já mencionado, apresentado em [28]. Em seguida, estender o sistema linear estável para que possa ser aplicado, também, em problemas com variáveis canalizadas. E, desenvolver um método eficiente para a resolução dessa nova versão da abordagem do sistema linear estável. Por fim, implementar esse método com o intuito de obter uma melhor precisão na resolução dos problemas de programação linear.

A motivação desse estudo consiste em obter resoluções com alta precisão para problemas lineares. Enquanto os métodos tradicionais resolvem problemas com uma tolerância da ordem de 10^{-8}, procuramos uma precisão da ordem de 10^{-14}, ou ainda menor. Porém, o custo computacional para se obter essa melhor precisão, não pode ser muito alto, ou seja, queremos um método com alta precisão e, também, eficiente no que diz respeito ao tempo computacional.

O trabalho, aqui realizado, é apresentado com a seguinte estrutura, no capítulo 2 é realizada uma introdução de programação linear com o objetivo de apresentar o método preditor-corretor para pontos-interiores. No capítulo 3 apresentamos o método híbrido, como já mencionado, proposto por Bocanegra, Campos e Oliveira. A abordagem sistema linear estável, nosso objeto de estudos, é apresentada no capítulo 4. No capítulo 5, apresentamos nosso principal resultado, a extensão da abordagem sistema linear estável para problemas que possuem variáveis canalizadas. Também apresentamos nesse capítulo, o método preditor-corretor para problemas desse tipo. No capítulo 6, apresentamos os resultados dos experimentos numéricos comparados com os resultados obtidos com o método híbrido. Finalmente, apresentamos nossas conclusões e trabalhos futuros no capítulo 7.

Capítulo 2

Programação Linear

A área de programação linear é uma das mais vastas dentro da matemática aplicada. Uma excelente introdução desse tópico pode ser encontrada em [12]. O objetivo deste capítulo é apresentar os fundamentos do método de pontos-interiores mais utilizado nos dias de hoje, o método preditor-corretor.

O método preditor-corretor é um método do tipo primal-dual, que utiliza o polinômio de Taylor de segunda ordem para o cálculo da direção primal-dual.

2.1 Otimização Linear

A programação linear consiste em minimizar, ou maximizar, uma função objetivo linear de forma que essa solução satisfaça um conjunto de equações e inequações lineares e limites de variáveis quando necessário. Dizemos que um problema está na forma padrão quando pode ser escrito da seguinte forma:

$$\begin{aligned} \text{Minimizar} \quad & \phi = c_1x_1 + c_2x_2 + ... + c_nx_n \\ \text{sujeito a} \quad & a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1 \\ & a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = b_2 \\ & \qquad\qquad \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n = b_m \\ & x_1, x_2, ..., x_n \geq 0 \end{aligned}$$

onde $x_1, x_2, ..., x_n \in \mathbb{R}$ são as variáveis, $c_1, c_2, ..., c_n \in \mathbb{R}$ são os coeficientes de custo e ϕ é a função objetivo a ser minimizada. Podemos escrever um problema linear na forma padrão em notação matricial:

$$\begin{array}{lrcl} \text{Minimizar} & \phi & = & c^t x \\ \text{sujeito a} & Ax & = & b \\ & x & \geq & 0 \end{array} \tag{2.1}$$

onde $x = (x_1, x_2, ..., x_n)^t, c = (c_1, c_2, ..., c_n)^t, b = (b_1, b_2, ..., b_m)^t$ e A representa a matriz das restrições, ou seja, a matriz dos coeficientes a_{ij}, onde

$$\sum_{j=1}^{n} a_{ij} x_j = b_i.$$

Se um vetor x satisfaz todas as restrições, dizemos que ele é *factível*. O conjunto de todos os pontos factíveis é chamado *conjunto factível*.

A formulação (2.1) é denominada problema primal e está associada ao problema dual,

$$\begin{array}{ll} \text{Maximizar} & \phi = b^t y \\ \text{sujeito a} & A^t y + z = c \\ & z \geq 0; \quad y \text{ livre} \end{array} \tag{2.2}$$

onde y representa um vetor coluna de variáveis duais e z a variável de folga complementar. Os dois problemas juntos são denominados de *par primal-dual*.

Podemos observar que os problemas (2.1) e (2.2) estão relacionados ao analisarmos o conjunto factível de cada problema. O conjunto factível do problema primal e seu conjunto solução fornecem informações importantes sobre o problema dual, e vice-versa. Além disso, dado um vetor factível x para (2.1) e (y, z) para (2.2), temos

$$b^t y \leq c^t x,$$

ou seja, a função objetivo dual é um limitante inferior para a função objetivo primal, e a primal um limitante superior para a dual, para problemas factíveis. As duas funções objetivo coincidem em uma solução, isto é, $b^t y^* = c^t x^*$, onde x^* é uma solução para (2.1) e (y^*, z^*) é uma solução para (2.2).

Um ponto interior de um problema de programação linear satisfaz a condição que toda variável encontra-se estritamente dentro de seus limites. Na forma padrão, $x > 0$ é um ponto interior no problema primal e (y, z) com $z > 0$ é um ponto interior para o problema dual. Um ponto interior factível, além de ser interior, também, satisfaz todas as restrições. Para o problema

primal, x é um ponto interior factível se $Ax = b$ e $x > 0$; Para o problema dual, (y, z) é um ponto interior factível se $A^t y + z = c$ e $z > 0$.

Para o desenvolvimento dos métodos de pontos interiores, utilizaremos o conceito de *gap*. O *gap* é a diferença entre os valores das funções objetivo dos problemas primal e dual, ou seja, $\gamma = c^t x - b^t y$ [64]. É possível mostrar que para um ponto primal e dual factível, o *gap* é dado por $\gamma = x^t z$.

As condições de otimalidade de primeira ordem, ou condição de Karush-Kuhn-Tucker (KKT), para os problemas (2.1) e (2.2) são:

$$\begin{aligned} &Ax - b = 0 \\ &A^t y + z - c = 0 \\ &XZe = 0 \\ &(x, z) \geq 0, \end{aligned} \tag{2.3}$$

onde $X = \text{diagonal}(x)$, $Z = \text{diagonal}(z)$ e e é o vetor coluna de dimensão n com todos os elementos iguais a 1. Podemos encontrar mais detalhes sobre as condições KKT em livros de programação linear como por exemplo [59, 64].

Analisando as condições em (2.3), podemos ver que o vetor (x^*, y^*, z^*) é uma solução para os sistemas em (2.3) se e somente se x^* é uma solução para o problema primal (2.1) e (y^*, z^*) é uma solução para o problema dual (2.2). O vetor (x^*, y^*, z^*) é chamado *solução primal-dual*.

2.2 Método de Newton

Os métodos de pontos interiores do tipo primal-dual consistem na aplicação do método de Newton às condições de otimalidade partindo de um ponto interior [64]. Apresentaremos nesta seção o método de Newton para uma variável e em seguida para várias variáveis. Finalmente, apresentamos o método de Newton para sistemas de equações não-lineares [15]. Para mais detalhes sobre o método de Newton e suas propriedades, ver [5, 39].

2.2.1 Método de Newton para uma variável

Considerando a função de uma variável $f(x)$, para encontrarmos o valor de x^* tal que $f(x^*) = 0$, utilizamos aproximações sucessivas da função $f(x)$ em torno dos pontos $x^0, x^1, ..., x^k$, até que o ponto x^k seja tal que $f(x^k) \approx 0$. Para tanto, vamos utilizar a fórmula de Taylor em torno do

ponto x^0:

$$0 = f(x^*) = f(x^0) + f^{'}(x^0)(x - x^0) + \frac{f^{''}(x^0)}{2!}(x - x^0)^2 + ... \tag{2.4}$$

Aproximamos $f(x^1)$ até o termo linear da série, ou seja,

$$0 = f(x^*) \approx f(x^0) + f^{'}(x^0)(x - x^0) \Rightarrow x^1 = x^0 - \frac{f(x^0)}{f^{'}(x^0)}. \tag{2.5}$$

Podemos aplicar esta fórmula para obter x^1 e depois calcular o valor de x^k sucessivamente, até que $f(x^k) \approx 0$. Assim, construímos o método de Newton para uma variável.

2.2.2 Método de Newton para várias variáveis

Seja $\bar{x} \in \mathbb{R}^n$ tal que $f_i(\bar{x}) = 0$, para $i = 1, 2, ..., n$. Novamente, para encontrarmos o valor de $\bar{x}$ utilizamos aproximações sucessivas da função $f(x)$ em torno dos pontos $x^0, x^1, ..., x^k$, até que o ponto x^k seja tal que $f_i(x^k) \approx 0$. Aplicando a fórmula de Taylor para várias variáveis em torno de x^0:

$$0 = f_i(\bar{x}) = f_i(x^0) + \left[\nabla f_i(x^0)\right]^t (x - x^0) + ..., \forall\ i, \tag{2.6}$$

onde

$$\nabla f_i(x) = \begin{bmatrix} \dfrac{\partial f_i(x)}{\partial x_1} \\ \vdots \\ \dfrac{\partial f_i(x)}{\partial x_n} \end{bmatrix} \tag{2.7}$$

Aproximando $f_i(\bar{x})$ até o termo linear da série, obtemos

$$0 = f_i(\bar{x}) \approx f_i(x^0) + \left[\nabla f_i(x^0)\right]^t (x^1 - x^0), \forall\ i, \tag{2.8}$$

ou seja

$$\begin{aligned} -f_1(x^0) &= \left[\nabla f_1(x^0)\right]^t (x^1 - x^0) \\ \vdots \qquad &\qquad \vdots \\ -f_n(x^0) &= \left[\nabla f_n(x^0)\right]^t (x^1 - x^0). \end{aligned}$$

Seja $J(x^0) = \begin{bmatrix} \nabla f_1(x^0)^t \\ \vdots \\ \nabla f_n(x^0)^t \end{bmatrix}$ a matriz Jacobiana e $F(x^0) = \begin{bmatrix} f_1(x^0) \\ \vdots \\ f_n(x^0) \end{bmatrix}$. Então,

$$-F(x^0) = J(x^0)(x^1 - x^0) \Rightarrow x^1 = x^0 - \left[J(x^0)\right]^{-1} F(x^0). \tag{2.9}$$

Construímos, assim, o método de Newton para várias variáveis.

2.2.3 Método de Newton para sistema de equações

Vamos agora obter o método de Newton para sistemas de equações, que quando aplicado às condições de otimalidade surgem os métodos de pontos interiores do tipo primal-dual.

Seja $F : \mathbb{R}^n \to \mathbb{R}^n$. Queremos encontrar $x_* \in \mathbb{R}^n$ tal que $F(x_*) = 0$, onde F é continuamente diferenciável.

O método de Newton para o problema acima é obtido encontrando uma aproximação para a função F na iteração x_k. Essa aproximação é calculada utilizando as mesmas técnicas que o método de Newton para várias variáveis. Obtemos então, o seguinte algoritmo.

Algoritmo 2.1 (Método de Newton para sistema de equações).

Sejam $F : \mathbb{R}^n \to \mathbb{R}^n$ continuamente diferenciável e $x_0 \in \mathbb{R}^n$

Para $k = 0, 1, 2, ...,$ faça

1. $J(x_k)s_k = -F(x_k)$;
2. $x_{k+1} = x_k + s_k$.

2.3 Métodos de pontos interiores primais-duais

Logo após Karmarkar [31] apresentar o primeiro método de pontos-interiores para programação linear com complexidade polinomial, surgiram muitas variações desse método. Dentre elas, a classe mais eficiente é a conhecida como primal-dual, introduzida por Megiddo [44] e apresentada na forma algorítmica em [34]. Estes métodos possuem o mesmo esforço computacional por iteração, em comparação com outras classes de métodos de pontos-interiores. Porém, os métodos primais-duais apresentam melhores resultados teóricos para a análise de complexidade no pior caso, como por exemplo em [47], e, também, para a análise assintótica da taxa de convergência (veja [57]). Esses resultados teóricos foram parcialmente motivados pelos experimentos práticos, em que o método primal-dual apresentou um melhor desempenho em relação ao método primal e ao dual, como em [40, 43].

Esses métodos encontram soluções primal-dual (x^*, y^*, z^*) aplicando uma variação do método de Newton às condições de otimalidade, (2.3), alterando a direção de busca e o tamanho do passo a ser tomado, de forma que a desigualdade $(x, z) \geq 0$ seja estritamente satisfeita em todas as iterações.

Vamos reescrever as equações das condições de otimalidade (2.3) na forma de uma função de $\mathbb{R}^{2n+m}$ em $\mathbb{R}^{2n+m}$:

$$F(x,y,z) = \begin{pmatrix} A^t y + z - c \\ Ax - b \\ XZe \end{pmatrix} = 0 \tag{2.10}$$

$$(x,z) \geq 0. \tag{2.11}$$

Podemos ver que F é linear nos dois primeiros conjuntos de equações.

Todos os métodos primais-duais produzem uma sequência (x^k, y^k, z^k) que satisfaz (2.11) estritamente, isto é, $x^k > 0$ e $z^k > 0$. Daí a origem da expressão *pontos interiores*.

Como a maioria dos algoritmos iterativos na otimização, os métodos primais-duais para pontos interiores possuem dois passos principais: o cálculo da direção e o tamanho do passo a ser tomado.

Aplicando o método de Newton para várias variáveis em F, obtemos o seguinte sistema linear de equações:

$$J(x,y,z) \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = -F(x,y,z),$$

onde J é o Jacobiano de F. O sistema acima pode ser reescrito como,

$$\begin{pmatrix} 0 & A^t & I \\ A & 0 & 0 \\ Z & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = - \begin{pmatrix} A^t y + z - c \\ Ax - b \\ XZe \end{pmatrix}. \tag{2.12}$$

Nem sempre é possível tomarmos a solução do sistema acima como passo, pois temos que respeitar a restrição $(x,z) \geq 0$. Assim, através de um teste da razão na direção de Newton, obtemos uma nova solução primal-dual,

$$(x,y,z) + \alpha(\Delta x, \Delta y, \Delta z),$$

onde $\alpha \in (0,1]$.

Esta abordagem não funciona bem na prática, pois não converge para muitos problemas devido ao fato de que, no produto XZe, algumas coordenadas $(x_i \times z_i)$ se aproximam de zero

mais rapidamente que outras. Podemos fazer uma alteração nessa abordagem de forma que fique mais eficiente. Ao realizarmos essa alteração, obtemos o *método primal-dual de trajetória central.*

O método primal-dual de trajetória central é obtido quando acrescentamos uma perturbação μ que procura manter cada coordenada próxima de um mesmo valor, e que reduz a condição de complementaridade simultaneamente. Essa perturbação é definida por,

$$\mu = \sigma\left(\frac{\gamma}{n}\right),$$

onde $\gamma = x^t z$ e $\sigma \in [0, 1)$. Assim, acrescentamos μ na última linha do lado direito do sistema (2.12) e obtemos, então, $\mu e - XZe$.

Algoritmo 2.2 (Método Primal-dual para pontos-interiores).

Dados y^0 e $(x^0, z^0) > 0$.

Para $k = 0, 1, 2, ...$, faça

1. Escolha $\sigma^k \in [0, 1)$ e seja $\mu^k = \sigma^k\left(\frac{\gamma^k}{n}\right)$, onde $\gamma^k = (x^k)^t z^k$.

2. Resolva o seguinte sistema linear:

$$\begin{pmatrix} 0 & A^t & I \\ A & 0 & 0 \\ Z^k & 0 & X^k \end{pmatrix} \begin{pmatrix} \Delta x^k \\ \Delta y^k \\ \Delta z^k \end{pmatrix} = - \begin{pmatrix} r_d^k \\ r_p^k \\ r_c^k \end{pmatrix}, \tag{2.13}$$

onde

$$\begin{cases} r_d^k = A^t y^k + z^k - c \\ r_p^k = Ax^k - b \\ r_c^k = -\mu^k e + X^k Z^k e \end{cases} \tag{2.14}$$

3. Escolha o tamanho do passo $\alpha^k = min(1, \tau^k \rho_p^k, \tau^k \rho_d^k)$ para $\tau^k \in (0, 1)$, onde

$$\rho_p^k = \frac{-1}{\min\limits_i \left(\frac{\Delta x_i^k}{x_i^k}\right)} \qquad \text{e} \qquad \rho_d^k = \frac{-1}{\min\limits_i \left(\frac{\Delta z_i^k}{z_i^k}\right)}. \tag{2.15}$$

4. Calcule a nova iteração

$$(x^{k+1}, y^{k+1}, z^{k+1}) = (x^k, y^k, z^k) + \alpha^k(\Delta x^k, \Delta y^k, \Delta z^k).$$

Diferentes escolhas de σ^k e τ^k caracterizam os vários métodos. Em geral, τ^k é um número fixo escolhido de acordo com experimentos numéricos. Tomando τ^k próximo de um, de forma que o número de iterações tende a ser menor. Dessa forma, mantemos as variáveis não negativas sempre maiores que zero e, podemos utilizar ao máximo a direção de busca obtida. Por outro lado, o parâmetro σ^k, também chamado de parâmetro de centralização, representa uma idéia totalmente oposta. Seu objetivo é evitar que as componentes do ponto se aproximem de zero permitindo assim, que o método obtenha boas direções de busca para as iterações futuras. A escolha particular de $\sigma^k = 0$ nos fornece as versões afim dos métodos. Note que, quando tomamos a direção de busca inteira, isto é, o passo inteiro de Newton e, continuamos com (x, z) não negativos, o método toma o tamanho máximo do passo como $\alpha^k = 1$.

Essa classe de métodos primais-duais para pontos interiores com um ponto inicial factível e com σ e τ escolhidos apropriadamente, apresenta bons resultados teóricos, inclusive complexidade polinomial (veja [27] para os resultados mais importantes) e, também, uma convergência com taxa super-linear [57]. Na prática, é comum iniciarmos o método com um ponto não factível, pois, em geral, é muito caro, computacionalmente, encontrar um ponto factível. Alguns bons resultados teóricos foram obtidos desde então, porém não tão bons quanto os resultados com um ponto inicial factível, veja [64].

Além disso, calculamos o tamanho do passo para os problemas primal e dual separadamente. Assim, o tamanho do passo para o primal é $\alpha_p^k = min(1, \tau^k \rho_p^k)$ e para o dual, $\alpha_d^k = min(1, \tau^k \rho_d^k)$. E, então, substituímos o passo 4 do método por

$$\begin{aligned} x^{k+1} &= x^k + \alpha_p^k \Delta x^k \\ (y^{k+1}, z^{k+1}) &= (y^k, z^k) + \alpha_d^k (\Delta y^k, \Delta z^k). \end{aligned}$$

Essa separação dos passos é uma heurística que na prática tem produzido resultados melhores.

2.4 O método Preditor-Corretor

O método de pontos interiores primal-dual, descrito na seção anterior, não é o mais utilizado na prática. Logo após ser publicado, o método preditor-corretor [46] se tornou a escolha mais popular. Esse método difere do método descrito na Seção 2.3 pela escolha da direção de busca.

A direção de busca em cada iteração consiste de três componentes:

- uma direção afim escala, ou direção preditora, onde é utilizada a direção de Newton para a função $F(x, y, z)$ definida em (2.3), conforme (2.12).
- um termo de centralização σ, onde $\sigma \in [0, 1)$ para atualizar o valor de μ^k.
- uma direção corretora para compensar a não-linearidade do último conjunto de restrições.

As duas primeiras componentes, a direção afim escala e o termo de centralização, formam em conjunto o sistema apresentado em (2.13) para os métodos de pontos interiores. Porém, no método preditor-corretor a direção afim escala é calculada separadamente do termo de centralização. Obtemos então, a vantagem de calcularmos o parâmetro σ, de forma mais eficiente. Assim, se a direção afim escala produzir uma boa redução no valor de γ, de forma que se mantenha $(x, z) > 0$, então não precisaremos fazer grandes correções nessa direção e, portanto, σ assumirá um valor próximo de zero. Porém, se for possível movermos apenas uma pequena distância ao longo da direção afim escala antes de violarmos a restrição $(x, z) > 0$, então a direção afim escala necessitará de grandes correções e, nesse caso, σ assumirá um valor maior.

A direção afim escala é obtida pela resolução do sistema (2.13) com $\mu^k = 0$, ou seja,

$$\begin{pmatrix} 0 & A^t & I \\ A & 0 & 0 \\ Z^k & 0 & X^k \end{pmatrix} \begin{pmatrix} \Delta\tilde{x}^k \\ \Delta\tilde{y}^k \\ \Delta\tilde{z}^k \end{pmatrix} = - \begin{pmatrix} r_d^k \\ r_p^k \\ r_a^k \end{pmatrix}, \tag{2.16}$$

onde $r_a^k = X^k Z^k e$ e r_d^k e r_p^k são definidos como em (2.14)

A solução do sistema acima, $(\Delta\tilde{x}^k, \Delta\tilde{z}^k, \Delta\tilde{y}^k)^t$, é a chamada direção afim escala. Em seguida, calculamos o valor de α como no passo 3 do Algoritmo 2.2.

A desvantagem de calcularmos a perturbação μ separadamente é que precisamos resolver dois sistemas lineares ao invés de um por iteração. No entanto, como os dois sistemas possuem a mesma matriz de coeficientes, precisamos decompô-la apenas uma vez.

Por a direção afim escala ser calculada separadamente, podemos utilizá-la para estimar o erro na aproximação linear. Conhecendo esse erro, podemos calcular uma componente de correção, a direção corretora. Como as componentes central e corretora são obtidas por sistemas lineares com a mesma matriz de coeficientes que a direção afim escala, e por serem independentes, não precisamos calculá-las separadamente. Podemos simplesmente combinar os dois sistemas lineares em um único sistema, somando os vetores do lado direito em cada sistema, e calcular a direção combinada.

O termo de centralização σ^k é obtido pela seguinte heurística [46]

$$\begin{aligned}\tilde{\gamma}^k &= \frac{(x+\alpha_p\Delta\tilde{x}^k)^t(z+\alpha_d\Delta\tilde{z}^k)}{n},\\ \sigma^k &= \left(\frac{\tilde{\gamma}^k}{\gamma^k}\right)^3,\\ \mu^k &= \sigma^k\frac{\gamma^k}{n}.\end{aligned}$$

Novamente, se a direção afim escala produzir uma boa redução no valor de γ^k ($\tilde{\gamma}^k << \gamma^k$), enquanto $(x,z) > 0$, σ^k assumirá um valor próximo de zero. Caso contrário, σ^k assumirá valores mais próximos de 1.

Para calcular a perturbação μ_k, podemos resolver um sistema linear com a mesma matriz de coeficientes do sistema (2.16), mas com o vetor $(0,0,\mu^k e)^t$ no lado direito. No entanto, vimos que é mais eficiente calcular esse termo em conjunto com a direção de correção.

Agora, quando tomamos $\alpha_p^k = \alpha_d^k = 1$, no tamanho dos passos para a direção afim escala, obtemos $r_d^{k+1} = 0$ e $r_p^{k+1} = 0$. Além disso,

$$\begin{aligned}r_a^{k+1} &= -X^{k+1}Z^{k+1}e = -(X^k+\Delta\tilde{X}^k)(Z^k+\Delta\tilde{Z}^k)\\ &= -(X^kz^k+\Delta\tilde{X}^kz^k+X^k\Delta\tilde{z}^k+\Delta\tilde{X}^k\Delta\tilde{z}^k)\\ &= -(r_a^k+Z^k\Delta\tilde{x}^k+X^k\Delta\tilde{z}^k+\Delta\tilde{X}^k\Delta\tilde{z}^k)\\ &= -(-r_a^k+r_a^k+\Delta\tilde{X}^k\Delta\tilde{z}^k)\\ &= -\Delta\tilde{X}^k\Delta\tilde{Z}^ke\end{aligned}$$

onde $\Delta\tilde{X}^k = \text{diagonal}(\Delta\tilde{x}^k)$ e $\Delta\tilde{Z}^k = \text{diagonal}(\Delta\tilde{z}^k)$. Assim, a etapa corretora procura compensar essa não-linearidade, do novo sistema a ser resolvido, alterando a direção de busca de forma que r_a^{k+1} fique próximo de zero. Para isso, resolvemos o seguinte sistema linear,

$$\begin{pmatrix}0 & A^t & I\\ A & 0 & 0\\ Z^k & 0 & X^k\end{pmatrix}\begin{pmatrix}\Delta\hat{x}^k\\ \Delta\hat{y}^k\\ \Delta\hat{z}^k\end{pmatrix} = \begin{pmatrix}0\\ 0\\ -\Delta\tilde{X}^k\Delta\tilde{Z}^ke\end{pmatrix}. \tag{2.17}$$

Portanto, o cálculo do termo de centralização em conjunto com a direção de correção é obtida resolvendo o seguinte sistema:

$$\begin{pmatrix}0 & A^t & I\\ A & 0 & 0\\ Z^k & 0 & X^k\end{pmatrix}\begin{pmatrix}\Delta\hat{x}^k\\ \Delta\hat{y}^k\\ \Delta\hat{z}^k\end{pmatrix} = \begin{pmatrix}0\\ 0\\ \mu^ke-\Delta\tilde{X}^k\Delta\tilde{Z}^ke\end{pmatrix}. \tag{2.18}$$

A solução $(\Delta\hat{x}^k, \Delta\hat{z}^k, \Delta\hat{y}^k)^t$ é a chamada direção de correção.

Finalmente, a direção de busca é dada pela soma das direções afim escala e de correção,

$$\begin{aligned} \Delta x^k &= \Delta\tilde{x}^k + \Delta\hat{x}^k \\ \Delta y^k &= \Delta\tilde{y}^k + \Delta\hat{y}^k \\ \Delta z^k &= \Delta\tilde{z}^k + \Delta\hat{z}^k. \end{aligned}$$

Na prática, calculamos diretamente a direção de busca pela soma dos sistemas lineares (2.16) e (2.18):

$$\begin{pmatrix} 0 & A^t & I \\ A & 0 & 0 \\ Z^k & 0 & X^k \end{pmatrix} \begin{pmatrix} \Delta x^k \\ \Delta y^k \\ \Delta z^k \end{pmatrix} = - \begin{pmatrix} r_d^k \\ r_p^k \\ -r_a^k - \mu e + \Delta\tilde{X}^k \Delta\tilde{Z}^k e \end{pmatrix}. \tag{2.19}$$

Esse é o sistema da etapa corretora a ser resolvido.

Os sistemas (2.16) e (2.19) possuem a mesma matriz dos coeficientes, e portanto, a mesma decomposição LU. Assim, em (2.19) utilizamos a decomposição já realizada em (2.16). Dessa forma, o esforço computacional para resolver o sistema (2.16) é maior que o esforço computacional para resolver o sistema (2.19). O custo computacional para resolver o segundo sistema linear é compensado pela eficiência do método preditor-corretor.

2.5 Calculando as direções de busca

Apresentamos nesta seção duas abordagens para calcular as direções de busca nos métodos do tipo primal-dual, ou seja, duas abordagens para resolver o sistema linear (2.13). Veremos na Seção 2.6.4 o cálculo da direção para o método preditor-corretor e em que ele se diferencia do método padrão.

O grande esforço computacional em cada iteração do método consiste na resolução do sistema em (2.13). Eliminando a variável Δz^k do sistema, obtemos:

$$\begin{pmatrix} -D^k & A^t \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x^k \\ \Delta y^k \end{pmatrix} = - \begin{pmatrix} r_d^k - (X^k)^{-1} r_c^k \\ r_p^k \end{pmatrix}, \tag{2.20}$$

onde $(D^{-1})^k = (X^k)^{-1} Z^k$ e Δz^k pode ser calculado como

$$\Delta z^k = (X^k)^{-1}(-r_c^k - Z^k \Delta x^k).$$

O sistema em (2.20) é chamado *sistema aumentado*. Eliminando a variável Δx^k de (2.20) temos

$$S^k \Delta y^k = -r_p^k - A(D^k)^{-1}(r_d^k - (X^k)^{-1} r_c^k), \tag{2.21}$$

onde $S^k = A(D^k)^{-1}A^t$ é chamado complemento de Schur. O sistema acima é chamado de *sistema de equações normais*. O cálculo de Δx^k é dado por,

$$\Delta x^k = (D^k)^{-1}(A^t \Delta y^k - r_d^k - (X^k)^{-1} r_c^k).$$

Note que, D^k é uma matriz diagonal com seus elementos positivos e é a única alteração na matriz dos sistemas (2.20) e (2.21) em cada iteração. Enquanto alguns elementos de D^k convergem para zero, outros convergem para infinito. Sistemas como em (2.20) e em (2.21) aparecem em outros métodos de pontos interiores, como por exemplo nas versões primal e dual [60]. Na versão primal afim escala [16] temos $D^k = (X^k)^2$ e o lado direito igual $(c^t \ 0^t)^t$. Já na versão dual afim escala [2], $D^k = (Z^k)^2$ e o lado direito é dado por $(0^t \ b^t)^t$.

Observe que para os métodos afins, não existe termo de centralização. Estes métodos não possuem as mesmas boas características teóricas que o método primal-dual e, por isso, foram superados em experimentos numéricos [43].

2.6 Métodos para resolução de sistemas lineares

Métodos diretos ou iterativos podem ser utilizados tanto para resolver o sistema (2.20), quanto para resolver o sistema (2.21). A abordagem mais utilizada nos métodos de pontos interiores para programação linear resolve o sistema de equações normais por um método direto: a decomposição de Cholesky.

2.6.1 Decomposição de matrizes

Um dos métodos diretos mais utilizados para se resolver sistemas lineares consistem em decompor uma permutação da matriz dos coeficientes no produto de duas matrizes triangulares $PAQ = LU$ e, então resolver o sistema por substituições. Esse procedimento é chamado decomposição LU.

Se a matriz do sistema é uma matriz simétrica e definida-positiva, então podemos calcular sua decomposição LU de forma que $U = L^t$, onde os elementos da diagonal de L são positivos. Essa decomposição é conhecida como Decomposição de Cholesky. Existe uma variação desse método, decompondo a matriz no produto de duas matrizes triangulares com elementos na

diagonal principal iguais a 1 e uma matriz diagonal, ou seja, LDL^t. Uma matriz dessa forma possui uma única Decomposição de Cholesky.

2.6.2 Resolvendo as equações normais

Utilizar a decomposição de Cholesky de S^k é, sem dúvida, a forma mais utilizada para se encontrar as direções de busca nos métodos de pontos interiores, como por exemplo em [2, 40, 41, 43]. Assim, utiliza-se todas as vantagens de S^k ser simétrica e positiva-definida. No entanto, S^k pode ser bem menos esparsa e, geralmente, é mais mal-condicionada do que a matriz do sistema (2.20). O caso extremo da perda de esparsidade consiste no caso em que A possui uma coluna com todos os elementos diferentes de zero e, ao calcular o sistema de equações normais, obtemos uma matriz com todos os elementos diferentes de zero.

Uma forma de evitarmos esse problema é utilizarmos métodos iterativos. Estes métodos consistem em construir uma sequência de aproximações da solução para o sistema linear, até que uma certa tolerância seja atingida. Como esses métodos requerem apenas o produto matriz-vetor, não precisamos calcular diretamente o sistema de equações normais, exceto no caso em que o pré-condicionador dependa dele. Portanto, a perda de esparsidade pode não ser um problema para esses métodos.

O método dos gradientes conjugados pré-condicionado é o método iterativo mais utilizado para resolver sistemas positivos-definidos. Esse método é fácil de se implementar e sua convergência é muito rápida com um bom pré-condicionador. Podemos ver em [41, 45] alguns resultados do método para resolver (2.21). Nem sempre vamos obter um bom resultado, principalmente, devido ao fato de que, o sistema linear se torna altamente mal-condicionado quando o método de pontos-interiores se aproxima de uma solução. Outra desvantagem dessas implementações é que ela necessita do cálculo do sistema de equações normais, para construir o pré-condicionador. E, no caso em que temos algumas poucas colunas densas, podemos perder a esparsidade da matriz.

Por esses, e outros, motivos, o sistema aumentado passou a ser foco de estudos e pesquisas.

2.6.3 Resolvendo o sistema aumentado

O sistema aumentado é um sistema indefinido, consequentemente, sua decomposição de Cholesky não existe, pois não é possível decompor sua matriz indefinida no produto LDL^t com D diagonal e positivo definido.

A decomposição de Bunch-Parlett [9] é um método direto que tem sido utilizado para

resolver sistemas indefinidos. Esse método contorna os problemas mencionados acima, decompondo a matriz em LDL^t, com L triangular e D uma matriz bloco diagonal, onde os blocos são de dimensão 1×1 ou 2×2. Uma decomposição estável é garantida escolhendo os blocos diagonais de forma que o "crescimento" dos elementos na matriz reduzida seja controlado. Por crescimento, entendemos o valor do maior elemento da decomposição, comparado com o maior elemento da matriz original. Geralmente, a escolha dos blocos diagonais envolve uma permutação simétrica de linhas e colunas.

A implementação da decomposição de Bunch-Parlett se mostrou mais estável, porém mais lenta em comparação com resolver (2.21), veja [21, 23, 60]. Já um método multifrontal aplicado ao sistema aumentado foi estudado em [19].

O método dos gradientes conjugados foi definido para ser utilizado, apenas, em sistemas definidos, por isso, ele não é utilizado para resolver (2.20). MINRES e SYMMLQ [52] são dois métodos iterativos para resolver sistema indefinidos simétricos. Alguns métodos provenientes do método dos gradientes conjugados foram apresentados em [37, 38], mas o MINRES e SYMMLQ ainda são mais rápidos. Em [23], SYMMLQ é utilizado para resolver o sistema aumentado de alguns problemas de pequeno porte.

2.6.4 Direções de busca para o método preditor-corretor

Todos os resultados apresentados anteriormente se aplicam ao método preditor-corretor, pois a estrutura da matriz nos métodos de pontos-interiores apresentados aqui é a mesma.

Uma diferença importante é que no método preditor-corretor calculamos apenas uma decomposição por iteração, o que nos economiza muito tempo e esforço computacional. Isso porque as matrizes nos dois sistemas lineares são iguais e, portanto, podemos utilizar a mesma decomposição nos dois sistemas. Também utilizamos a mesma decomposição do sistema de equações normais, onde for necessário. Essa é uma das razões para o método preditor-corretor ser tão utilizado nos dias de hoje.

Ainda não está claro qual o melhor momento de se utilizar os métodos iterativos, para resolver os sistemas lineares de forma mais eficiente, no método preditor-corretor. Essa escolha depende do custo computacional do cálculo do pré-condicionador com relação ao custo computacional da resolução dos dois sistemas lineares, e, também, do número de iterações dos métodos iterativos para se obter a convergência.

Os sistemas lineares (2.16) e (2.19) estão relacionados, pois possuem a mesma matriz e o

terceiro bloco do vetor à direita no sistema (2.19) depende da solução do sistema (2.16). Assim, talvez seja possível obtermos melhores resultados ao calcularmos a solução do sistema linear (2.19) pela abordagem de métodos iterativos. Contudo, não é fácil encontrar uma forma de acelerarmos o cálculo dessa solução. Isso ocorre, provavelmente, devido ao fato da perturbação no lado direito ser não linear. Portanto, a idéia de encontrarmos uma solução inicial melhor para o segundo sistema linear não será foco de nosso estudo.

Capítulo 3

Pré-condicionadores híbridos para métodos iterativos

Nessa seção apresentamos uma implementação composta de duas fases para resolver problemas lineares utilizando métodos de pontos interiores. Nas duas fases, é utilizado o método dos gradientes conjugados para resolver o sistema de equações normais em (2.21) pré-condicionado por uma matriz M,

$$M^{-1}(AD^{-1}A^t)M^{-t}\bar{y} = M^{-1}(-r_p - AD^{-1}(r_d - X^{-1}r_a)). \tag{3.1}$$

onde $\bar{y} = M^t\Delta y$. Na primeira fase, a decomposição controlada de Cholesky é utilizada para construir a matriz M, esse método é apresentado na seção 3.2. Após a mudança de fases, é utilizado o pré-condicionador separador, apresentado na seção 3.3.

Antes, porém, vejamos um pouco sobre pré-condicionadores e matrizes pré-condicionadas.

3.1 Pré-condicionadores

A aplicação de métodos iterativos para a solução de sistemas lineares com matrizes mal-condicionadas pode ocasionar a não convergência do método. Um pré-condicionador tem como objetivo transformar um sistema linear com uma matriz mal-condicionada em um sistema equivalente com uma matriz bem-condicionada. Chamamos essa transformação de *pré-condicionamento*. O pré-condicionamento é realizado com o objetivo de transformar o sistema original em um sistema equivalente pré-condicionado, onde sua solução pode ser obtida de uma forma mais simples que o sistema original.

Seja $\hat{A}$ uma matriz quadrada com posto completo. Dizemos que uma matriz é mal-condicionada, se $\kappa_p(\hat{A})$ assume valores grandes, onde $\kappa_p(\hat{A}) = ||\hat{A}||_p||\hat{A}^{-1}||_p$. Podemos observar que, para matrizes simétricas positivas-definidas o número de condição na norma-2, é a divisão do maior pelo menor autovalor.

O método dos gradientes conjugados, [24], bem como os demais métodos iterativos, é muito sensível ao número de condição de uma matriz, ou seja, o erro máximo cometido no método depende do número de condição da matriz:

$$||x - x_k||_{\hat{A}} \leq 2||x - x_0||_{\hat{A}}\alpha^k$$

onde $||w||^2_{\hat{A}} \equiv w^t\hat{A}w$ e $\alpha = \frac{\sqrt{\kappa_2(\hat{A})}-1}{\sqrt{\kappa_2(\hat{A})}+1}$. Assim, o método dos gradientes conjugados é mais eficiente quando $\kappa_2(\hat{A}) \approx 1$.

Podemos aplicar a idéia de pré-condicionamento em qualquer sistema linear. Vejamos um exemplo: Seja o sistema linear $\hat{A}x = b$. Vamos construir o seguinte sistema linear equivalente,

$$\tilde{M}^{-1}\hat{A}\tilde{N}^{-1}\tilde{x} = \tilde{b},$$

onde $\tilde{x} = \tilde{N}x$ e $\tilde{b} = \tilde{M}^{-1}b$. Nesse caso, dizemos que o sistema linear foi pré-condicionado e $\tilde{M}^{-1}\hat{A}\tilde{N}^{-1}$ é chamada de matriz pré-condicionada. Geralmente, não é necessário calcular a matriz pré-condicionada, pois a grande maioria dos métodos iterativos a utiliza apenas para calcular o seu produto com vetores.

Dizemos que um pré-condicionador é simétrico se $\tilde{N}^t = \tilde{M}$, pois, nesse caso, se $\hat{A}$ é simétrica, então a matriz pré-condicionada $\tilde{M}^{-1}\hat{A}\tilde{M}^{-t}$ também será simétrica. Para os pré-condicionadores simétricos, a lei da inércia de Sylvester [24], nos garante que a matriz pré-condicionada possui o mesmo número de autovalores positivos, e negativos, que a matriz original. Portanto, os pré-condicionadores simétricos não podem ser utilizados para transformar um sistema indefinido em um sistema definido.

Como a maioria dos métodos iterativos possui uma taxa de convergência que depende da distribuição dos autovalores, podemos obter uma convergência mais rápida se a matriz $\tilde{M}^{-1}\hat{A}\tilde{N}^{-1}$ estiver próxima da matriz identidade. Uma forma de estarmos próximos dessa situação, é mover os autovalores positivos de $\hat{A}$ para próximo de 1 e os negativos para próximo de -1. Por outro lado, experimentos práticos mostram que as melhores escolhas para $\tilde{M}$ e $\tilde{N}$ são as que transformam o sistema original em um outro com baixo custo computacional, pois o principal objetivo do pré-condicionador é reduzir o tempo utilizado na resolução do sistema linear. Por

isso, os pré-condicionadores, geralmente, são construídos na forma de uma matriz triangular, ou bloco diagonal.

Geralmente, não existe uma forma de se construir o pré-condicionador ideal para determinado problema. Isso porque o pré-condicionador ideal está estritamente ligado ao problema a ser resolvido. Porém, existem alguns resultados que nos fornecem uma boa escolha.

3.2 Decomposição controlada de Cholesky

A decomposição controlada de Cholesky (CCF) é um procedimento desenvolvido para resolver sistemas positivo definidos em geral [10]. Ela também apresentou excelentes resultados para sistemas de dependência de tempo em equações diferenciais parciais [11]. Esse tipo de decomposição possui algumas propriedades muito úteis, como a possibilidade de controlar o número de elementos a ser introduzido na decomposição.

Sejam as decomposições de Cholesky, $AD^{-1}A^t = LL^t = \tilde{L}\tilde{L}^t + R$, onde L é a matriz obtida na decomposição completa, $\tilde{L}$ quando a decomposição é incompleta e R é a matriz da diferença das duas decomposições. Definindo $E = L - \tilde{L}$, então a matriz dos coeficientes pré-condicionada é dada por:

$$\tilde{L}^{-1}(AD^{-1}A^t)\tilde{L}^{-t} = (\tilde{L}^{-1}L)(\tilde{L}^{-1}L)^t = (I + \tilde{L}^{-1}E)(I + \tilde{L}^{-1}E)^t. \tag{3.2}$$

Agora, quando fazemos a decomposição incompleta próxima da decomposição completa de Cholesky, isto é, quando a matriz $\tilde{L}$ se aproxima da matriz L, então a matriz E se aproxima da matriz nula e o produto de matrizes $\tilde{L}^{-1}(AD^{-1}A^t)\tilde{L}^{-t}$ se aproxima da matriz identidade I. Nesse caso, dizemos que a matriz $AD^{-1}A^t$ foi escalonada de forma a obtermos a matriz identidade [20], a fim de uma maior robustez. O CCF é baseado no problema de minimizar a norma de Frobenius da matriz E. Assim, seja o problema

$$\textit{minimizar } ||E||_F^2 = \sum_{j=1}^{m} c_j \quad \text{com} \quad c_j = \sum_{i=1}^{m} |l_{ij} - \tilde{l}_{ij}|^2. \tag{3.3}$$

Agora, c_j pode ser dividido em dois somatórios:

$$c_j = \sum_{k=1}^{t_j+\eta} |l_{i_k j} - \tilde{l}_{i_k j}|^2 + \sum_{k=t_j+\eta+1}^{m} |l_{i_k j}|^2, \tag{3.4}$$

onde t_j representa o número de elementos diferentes de zero abaixo da diagonal na j-ésima coluna de $AD^{-1}A^t$, e η representa o número de elementos extras permitido por coluna. O

primeiro somatório contém todos os $t_j + \eta$ elementos diferentes de zero da j-ésima coluna de $\tilde{L}$. O segundo possui apenas os elementos da decomposição completa L que não possuem elementos correspondentes em $\tilde{L}$. Considerando que $\tilde{l}_{ij} \approx l_{ij}$ quando $\eta \to m$ e l_{ij} não é calculado, $||E||_F$ é minimizada através de uma heurística que consiste na modificação do primeiro somatório. Aumentando o valor de η, isto é, permitindo que mais elementos sejam acrescentados, c_j irá reduzir pois o primeiro somatório contém mais termos que o segundo. Além disso, $||E||_F$ é minimizada escolhendo os $t_j + \eta$ maiores valores de $\tilde{L}$, em módulo, de forma que o primeiro somatório fique próximo de zero, e o segundo somatório com apenas os elementos l_{ij} pequenos, em módulo, também fique próximo de zero. Essa é uma minimização seletiva, semelhante ao esquema de tolerância. O pré-condicionador $\tilde{L}$ é construído por colunas, ou seja, necessita apenas de uma coluna de $AD^{-1}A^t$ por vez. As principais características do pré-condicionador controlado de Cholesky são apresentas nas próximas subseções.

3.2.1 Escolha dos elementos por valor

O pré-condicionador controlado de Cholesky contém no máximo $t_j+\eta$ elementos diferentes de zero em cada coluna, sendo que a respectiva coluna da matriz $AD^{-1}A^t$ contém, exatamente, t_j elementos diferentes de zero, abaixo da diagonal. Assim, quando uma coluna do pré-condicionador é calculada, apenas os $t_j + \eta$ maiores elementos são armazenados e os demais são descartados. Portanto, a melhor aproximação da decomposição completa é obtida pelos elementos escolhidos. O pré-condicionador CCF não mantém a estrutura esparsa da matriz. Isso porque a escolha dos elementos é feita levando em consideração seus valores e não suas posições e, assim, a posição dos elementos diferentes de zero no pré-condicionador pode fazer com que alguns elementos da matriz original sejam perdidos. Uma vantagem do CCF em relação aos pré-condicionadores que mantém a estrutura esparsa da matriz é que ele pode requerer um espaço de armazenamento menor.

3.2.2 Generalização do ICF

Jones e Plassmann [30] apresentaram um resultado que permite o acréscimo de um número, η, de elementos diferentes de zero em cada linha, ou coluna, do pré-condicionador. Esse η é tomado como sendo o número de elementos não nulos em cada linha, ou coluna, da matriz de coeficientes original. Novamente, apenas os maiores valores em módulo são mantidos, o que significa que a estrutura esparsa da matriz é ignorada. Assim, o CCF pode ser visto como uma generalização

do método de Jones e Plassmann.

3.2.3 Shift exponencial

Manteufell [42] provou que, se a matriz de coeficientes V é simétrica e positiva-definida, então existe uma constante $\sigma > 0$ tal que a matriz $V + \sigma I$ possui uma decomposição incompleta de Cholesky. Durante o cálculo da j-ésima coluna de $\tilde{L}$, o elemento da diagonal principal pode ser muito pequeno, ou até mesmo negativo. Assim, para evitar que a matriz $\tilde{L}$ deixe de ser positiva-definida, o CCF desconsidera toda a matriz $\tilde{L}$, calculada até esse ponto, acrescenta σ_i na diagonal de $AD^{-1}A^t$ e calcula $\tilde{L}$ novamente. Ao invés de usar um shift linear como o de Jones e Plassmann [30], $\sigma_i = 10^{-2}i$, na i-ésima tentativa de construir o pré-condicionador $\tilde{L}$, o CCF utiliza um shift exponencial $\sigma_i = 5 \times 10^{-4} \times 2^{i-1}$ de forma a obter uma perturbação diagonal inicialmente menor.

3.2.4 Pré-condicionador versátil

Na Tabela 3.1, abaixo, apresentamos as características de uma matriz M pré-condicionadora versátil para (3.1). Como se observa, tomando $\eta = -m$, obtemos, apenas, um escalonamento diagonal, enquanto que, tomando $\eta = m$, obtemos uma decomposição completa de Cholesky. Portanto, η pode ser tomando no intervalo $[-m, m]$ de forma que M necessitará de mais, ou menos, memória para armazenamento que $AD^{-1}A^t$. Em outras palavras, *fill-in* ($\eta > 0$) e *drop-out* ($\eta < 0$) podem ser utilizados com CCF.

η	M	Memória
-m	$diag(AD^{-1}A^t)^{-1/2}$	menos que $AD^{-1}A^t$
0	$\tilde{L}$	igual a $AD^{-1}A^t$
m	L	mais que $AD^{-1}A^t$

Tabela 3.1: Fill-in e drop-out com CCF(η).

3.2.5 Armazenamento de matriz

Na implementação utilizada, a matriz dos coeficientes $AD^{-1}A^t$ e o pré-condicionador $\tilde{L}$ são armazenados segundo o formato de Harwell-Boeing. Cada matriz é armazenada em três vetores: um vetor do tipo inteiro com dimensão $m + 1$ para colunas; um outro vetor, também do tipo inteiro com dimensão t (número de elementos diferentes de zero abaixo da diagonal principal),

para o índice das linhas; e um vetor do tipo real, também com dimensão t, para os valores numéricos. A decomposição controlada de Cholesky necessita de outros vetores para construir $\tilde{L}$. Na decomposição incompleta de Cholesky (ICD), o fill-in é determinado pela soma dos níveis, resultando em uma demanda de memória sem controle. No entanto, a memória necessária para o CCF já é conhecida, como mostra a tabela 3.2, pois apenas os $t_j + \eta$ maiores elementos em cada coluna do pré-condicionador são mantidos. A terceira coluna nos mostra a quantidade de memória necessária para vetores inteiros com 4 bytes e vetores reais com 8 bytes.

A decomposição controlada de Cholesky com η elementos diferente de zero, CCF(η), necessita mais memória que ICD(0), decomposição incompleta de Cholesky com o mesmo número de elementos diferentes de zero do que a matriz original, quando $\eta > (1-2t-m)/(4m) \approx -t/(2m)$, para $i = 4$ e $r = 8$ bytes. No entanto, CCF(η) possui uma estrutura de dados mais versátil, o que permite que o número de elementos diferentes de zero de $\tilde{L}$ varie.

Matriz	Máximo de bytes	Para $i = 4$ e $r = 8$
$AD^{-1}A^t$	$(i+r)t + im + i$	$12t + 4m + 4$
$AD^{-1}A^t + \tilde{L}_{ICD(0)}$	$(i+2r)t + im + i$	$20t + 4m + 4$
$AD^{-1}A^t + \tilde{L}_{CCF(\eta)}$	$(3i+2r)t + ((2i+r)\eta + 2i)m + 2i$	$28t + (16\eta + 8)m + 8$

Tabela 3.2: Comparação entre o armazenamento de ICD(0) e CCF(η).

Na Tabela 3.2, m representa a ordem da matriz $AD^{-1}A^t$, t representa o número de elementos diferentes de zero abaixo da diagonal principal de $AD^{-1}A^t$, η representa o número de elementos extras por coluna e i, r representam o número de bytes para representar as variáveis do tipo inteiro e real, respectivamente.

3.3 Pré-condicionador separador

O pré-condicionador separador foi proposto para os sistemas indefinidos oriundos dos métodos de pontos interiores [50] para problemas lineares, tais como o sistema aumentado. Este pré-condicionador é uma generalização daquele proposto por Resende e Veiga [55] no contexto do problema de fluxo de rede de custo mínimo. A principal característica dessa classe de pré-condicionadores é que ela se comporta melhor quando está próxima de uma solução do problema linear. Essa é uma característica importante, pois o fato desses sistemas serem mal-condicionados

próximo do conjunto solução faz com que eles sejam difíceis de resolver por métodos iterativos. Além disso, o pré-condicionador separador não necessita do cálculo das equações normais. No entanto ele falha nas iterações iniciais para muitos problemas lineares.

Existe uma versão do pré-condicionador separador para o sistema de equações normais, que será estudado nessa seção.

De forma resumida o pré-condicionador separador pode ser obtido da seguinte forma: Seja $A = (B \quad E)P$ onde P é uma matriz de permutação tal que B é não singular. Então,

$$AD^{-1}A^t = BD_B^{-1}B^t + ED_E^{-1}E^t.$$

Agora, multiplicando à esquerda por $D_B^{\frac{1}{2}}B^{-1}$ e à direita por sua transposta, obtemos

$$T = D_B^{\frac{1}{2}}B^{-1}(AD^{-1}A^t)B^{-t}D_B^{\frac{1}{2}} = I + WW^t,$$

onde $W = D_B^{\frac{1}{2}}B^{-1}ED_E^{-\frac{1}{2}}$.

Observe que a matriz pré-condicionada é positiva definida e seus autovalores são maiores ou iguais a 1, ou seja, ela não possui autovalores próximos de zero.

Note que $B^{-1}E$ pode ser visto como um escalonamento do problema linear. Próximo da solução, a matriz D^{-1} possui pelo menos $n-m$ elementos diagonais pequenos. Então, com uma escolha apropriada das colunas de B, os elementos da diagonal de D_B e D_E^{-1} são muito pequenos. Nesse caso, W se aproxima da matriz nula, T se aproxima da identidade e os maiores autovalores de T e $\kappa_2(T)$ se aproximam de 1.

Assim, utilizando o pré-condicionador separador, $M^{-1} = D_B^{\frac{1}{2}}B^{-1}$, não precisamos calcular as equações normais para resolver o sistema. Porém, precisamos encontrar a matriz B e resolver sistemas lineares com ela. No entanto, a decomposição $QB = LU$ é, geralmente, mais esparsa do que a decomposição de Cholesky. De fato, sabemos por [22] que a estrutura esparsa de L e U está contida na estrutura de R, onde $AA^T = R^tR$, para qualquer matriz de permutação Q. Na prática, o número de elementos diferentes de zero de R é bem maior que número total de elementos diferentes de zero de L e U juntas.

3.3.1 Escolha da matriz B

Podemos escolher a matriz B minimizando $||W||$, pois próximo da solução a matriz pré-condicionada se aproxima da identidade, para uma escolha adequada da matriz B, uma vez que os elementos das matrizes diagonais $||D_B||$ e $||D_E^{-1}||$ se tornam muito pequenos. Porém, esse problema é de

difícil solução, mas uma aproximação dessa solução pode ser obtida selecionando as primeiras m colunas linearmente independentes de A, ordenadas de acordo com os valores da norma-1 das colunas de AD^{-1}, em ordem crescente.

Uma propriedade interessante dos pré-condicionadores separadores é que a mesma matriz B pode ser utilizada em algumas iterações sem perda acentuada de sua eficiência. Por isso, esse pré-condicionador é, computacionalmente, muito barato para tais iterações. Observe que, mesmo mantendo a matriz B da iteração anterior, isso não significa que teremos o mesmo pré-condicionador, pois a matriz D será alterada e o pré-condicionador depende dela também.

Para essa aplicação, a forma mais eficiente de se calcular a decomposição LU é utilizando a forma de atualização atrasada das colunas. Esse processo é muito útil pois, quando aparece uma coluna linearmente dependente, ela é descartada e o processo continua com a próxima coluna na ordem dada pela heurística de reordenamento.

Um fator inconveniente da implementação do pré-condicionador separador é o excesso de *preenchimento* na decomposição LU. Isso porque o reordenamento das colunas não considera a esparsidade da matriz A. Uma solução para esse problema consiste em interromper a decomposição quando ocorre um excesso de preenchimento e reordenar as colunas independentes encontradas até o momento pelo número de elementos diferentes de zero, por exemplo. A decomposição é, então, reiniciada e o processo é repetido até que m colunas independentes sejam encontradas. Na implementação descrita em [51], considera-se que uma decomposição tem um excesso de preenchimento quando o número de elementos diferentes de zero é maior do que o número de elementos diferentes de zero do sistema de equações normais.

Depois de encontrada a decomposição LU da matriz B, é feita uma nova decomposição LU desse conjunto de colunas linearmente independentes. Essa nova decomposição é realizada de forma que sua estrutura esparsa torne o cálculo na resolução dos sistemas lineares mais rápido. Essa abordagem produz resultados significativos para alguns problemas. Uma outra vantagem dessa abordagem é o fato de não ser necessário o armazenamento da matriz U para se determinar a matriz B.

Uma dificuldade para se determinar o conjunto das colunas linearmente independentes está no número de colunas dependentes encontradas durante o processo de decomposição. As técnicas apresentadas em [51] para a determinação do sub-conjunto das colunas e o cálculo do pré-condicionador separador são muito sofisticadas, pois o sub-conjunto das colunas de A que formam B não é conhecido a priori. No entanto, esta abordagem em uma implementação mais

sofisticada se mostrou melhor do que a decomposição de Cholesky para problemas de grande porte, quando a decomposição de Cholesky contém muitos elementos diferentes de zero.

Assim, podemos destacar as seguintes características do pré-condicionador separador:

1. B é mantido para algumas iterações;
2. O cálculo da decomposição LU utiliza a forma atrasada de atualização das colunas de A;
3. Caso ocorra um fill-in excessivo, o cálculo da decomposição LU é refeito;
4. Quando encontramos uma nova matriz B, re-calculamos sua decomposição LU;
5. Determina-se um conjunto de colunas simbolicamente (in)dependentes.

3.4 Mudança de fases

Nessa abordagem, a mudança de fases consiste na seguinte heurística: O pré-condicionador decomposição controlada de Cholesky é substituído pelo pré-condicionador separador quando o *gap* inicial ($x_0^t z_0$), na programação linear, é reduzido por um fator da ordem de 10^6, ou o número de iterações internas, para resolver o sistema linear, atinge $m/2$ iterações, onde m é a dimensão da matriz $AD^{-1}A^t$.

Na iteração inicial, o número de elementos diferentes de zero permitido no pré-condicionador controlada de Cholesky é definido pelo parâmetro inicial η_0

$$\eta_0 = \begin{cases} -|AD^{-1}A^t|/m, & \text{se} \quad |AD^{-1}A^t|/m > 10 \\ |AD^{-1}A^t|/m, & \text{caso contrário} \end{cases}$$

onde $|\cdot|$ representa o número de elementos não nulos de uma matriz. Se a matriz $AD^{-1}A^t$ é densa, então η_0 faz com que a matriz pré-condicionada seja a matriz diagonal. Agora, se $AD^{-1}A^t$ é esparsa, então a matriz pré-condicionadora poderá ser mais densa.

Na medida em que o método dos gradientes conjugados perde eficiência, o valor de η é incrementado. Se o número de iterações para se conseguir convergência é maior que $m/4$, η é incrementado por 10. Esse processo continua até η atingir o seu valor máximo permitido, $\eta_{máx}$, ou até ocorrer a mudança de fase. O valor de $\eta_{máx}$ é definido de acordo com a quantidade de memória disponível.

Uma descrição mais detalhada dessa implementação pode ser encontrada em [7]. Uma nova heurística, para esse método híbrido, desenvolvida em [61] apresentou resultados superiores.

Capítulo 4

O Sistema linear estável

Resolver o sistema linear (2.12) em cada iteração dos métodos primais-duais para pontos interiores representa o maior esforço computacional para tais métodos. Resolver (2.12) diretamente pode ser muito caro, computacionalmente falando, pois, geralmente, a matriz dos coeficientes nesses sistemas é muito grande e esparsa. A estrutura do sistema em (2.12) nos permite reformulá-lo como sistemas menores e simétricos, que são mais baratos para se decompor do que o sistema original.

Na Seção 2.5, vimos como obter dois desses sistemas mais simples, o sistema aumentado (2.20), e o sistema de equações normais (2.21), ou equações normais. Veremos agora uma terceira abordagem, o sistema linear estável.

Na Seção 4.2, apresentamos um novo método para se resolver o sistema estável, de modo que ele se torne competitivo com os métodos tradicionais para problemas lineares grandes e esparsos.

4.1 Processo de eliminação de blocos

O sistema estável, apresentado por González-Lima, Wei e Wolkowicz em [28], assim como o sistema aumentado e o sistema de equações normais, pode ser obtido pelo processo de eliminação de blocos aplicado em (2.12). Veremos esse processo a seguir.

O primeiro passo da eliminação de blocos, mais utilizado na literatura, consiste em eliminar Δz utilizando o primeiro conjunto de equações de (2.12). Podemos observar a linearidade e o coeficiente I para Δz em (2.12). Para isso, aplicamos operações elementares de matrizes em $J(x, y, z)$, ou seja, encontramos uma matriz P_Z tal que a multiplicação $P_Z J(x, y, z)$ resulta em

uma matriz com a coluna correspondente a Δz, ou seja, a primeira coluna da direita, formada apenas pela matriz identidade, isto é,

$$\begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ -X & 0 & I \end{pmatrix} \begin{pmatrix} 0 & A^t & I \\ A & 0 & 0 \\ Z & 0 & X \end{pmatrix} = \begin{pmatrix} 0 & A^t & I \\ A & 0 & 0 \\ Z & -XA^t & 0 \end{pmatrix}. \tag{4.1}$$

Agora, multiplicando o lado direito do sistema (2.12) por P_Z, obtemos

$$\begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ -X & 0 & I \end{pmatrix} \begin{pmatrix} A^t y + z - c \\ Ax - b \\ ZXe - \mu e \end{pmatrix} = - \begin{pmatrix} rd \\ r_p \\ -Xrd + ZXe - \mu e \end{pmatrix}. \tag{4.2}$$

Temos então,

$$P_Z = \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ -X & 0 & I \end{pmatrix}, \qquad K = \begin{pmatrix} 0 & A^t & I \\ A & 0 & 0 \\ Z & -XA^t & 0 \end{pmatrix} \tag{4.3}$$

Para o segundo passo, assumimos que a matriz A possui a estrutura $A = (B \quad E)$, obtida por uma permutação de linhas e colunas, onde B é uma matriz quadrada de ordem m, não singular, bem-condicionada e, além disso, o sistema $Bu = d$ é fácil de ser resolvido, ou seja, a matriz B pode ser decomposta no produto LU onde L e U são esparsas. Por exemplo, a melhor escolha para B poderia ser $B = I$, obtida no caso em que existe uma variável de folga em cada restrição.

Em seguida, particionamos as matrizes diagonais Z, X usando os vetores $z = \begin{pmatrix} z_B \\ z_E \end{pmatrix}$, $x = \begin{pmatrix} x_B \\ x_E \end{pmatrix}$, onde as dimensões de z_B e x_B são m e as dimensões de z_E e x_E são $n - m$. Seja K dado em (4.3), definimos as matrizes F_B, P_B como

$$\begin{aligned} F_B \equiv P_B K &= \begin{pmatrix} I_n & 0 & 0 & 0 \\ 0 & B^{-1} & 0 & 0 \\ 0 & -Z_B B^{-1} & I_B & 0 \\ 0 & 0 & 0 & I_E \end{pmatrix} \begin{pmatrix} 0 & 0 & A^t & I_n \\ B & E & 0 & 0 \\ Z_B & 0 & -X_B B^t & 0 \\ 0 & Z_E & -X_E E^t & 0 \end{pmatrix} \\ &= \left(\begin{array}{cc|cc} 0 & 0 & A^t & I \\ I_m & B^{-1}E & 0 & 0 \\ \hline 0 & -Z_B B^{-1} E & -X_B B^t & 0 \\ 0 & Z_E & -X_E E^t & 0 \end{array}\right). \end{aligned}$$

E no lado direito obtemos,

$$-P_B P_Z \begin{pmatrix} A^t y + z - c \\ Ax - b \\ ZXe - \mu e \end{pmatrix} = -P_B \begin{pmatrix} r_d \\ r_p \\ -X_B(r_d)_B + Z_B X_B e - \mu e \\ -X_E(r_d)_E + Z_E X_E e - \mu e \end{pmatrix}$$

$$= \begin{pmatrix} -r_d \\ -B^{-1} r_p \\ Z_B B^{-1} r_p + X_B(r_d)_B - Z_B X_B e + \mu e \\ X_E(r_d)_E - Z_E X_E e + \mu e \end{pmatrix}$$

A abordagem do sistema estável utiliza os dois últimos conjuntos de equações para calcular Δx_E e Δy. Depois, utiliza o segundo conjunto de equações para calcular Δx_B e, finalmente, utiliza o primeiro conjunto de equações para calcular Δz. A matriz B^{-1} não é calculada, apenas a decomposição LU de B para a resolução do sistema.

Assim, utilizando o índice k para denotar o número da iteração, a abordagem do sistema estável para se obter a direção de busca consiste em:

$$\begin{pmatrix} -Z_B^k B^{-1} E & -X_B^k B^t \\ Z_E^k & -X_E^k E^t \end{pmatrix} \begin{pmatrix} \Delta x_E^k \\ \Delta y^k \end{pmatrix} = \begin{pmatrix} Z_B^k B^{-1} r_p + X_B^k (r_d)_B - M_B \\ X_E^k (r_d)_E - M_E \end{pmatrix} \tag{4.4}$$

onde $M_B = Z_B^k X_B^k e_B + \mu^k e_B$ e $M_E = Z_E^k X_E^k e_E + \mu^k e_E$. Em seguida, obtemos o valor de Δx_B^k e Δz^k por

$$\Delta x_B^k = B^{-1}(-r_p - E\Delta x_E^k) \quad \text{e} \quad \Delta z^k = -r_d - A^t \Delta y^k.$$

Podemos ver que, o sistema não possui as matrizes $(X^k)^{-1}$ e $(Z^k)^{-1}$, seja na matriz do sistema ou em seu lado direito. No entanto, como a matriz desse sistema não possui uma forma especial, é difícil encontrar métodos iterativos ou decomposições eficientes para resolvê-lo.

O sistema estável pode ser transformado em sistemas equivalentes do mesmo tamanho, mas que sejam simétricos ou simétricos quase-definidos, de forma que possamos explorar sua estrutura com decomposições apropriadas. Durante essa transformação, devemos tomar o cuidado para não introduzirmos as matrizes $(X^k)^{-1}$ e $(Z^k)^{-1}$ para que a principal propriedade da abordagem do sistema estável não seja perdida.

4.2 Sistema estável simétrico em estrutura

Podemos transformar o sistema estável (4.4) em um sistema equivalente simétrico em estrutura,

$$\begin{pmatrix} Z_E^k & -X_E^k E^t B^{-t} \\ -Z_B^k B^{-1} E & -X_B^k \end{pmatrix} \begin{pmatrix} \Delta x_E^k \\ \Delta \tilde{y}^k \end{pmatrix} = \begin{pmatrix} X_E^k (r_d)_E - M_E \\ Z_B^k B^{-1} r_p + X_B^k (r_d)_B - M_B \end{pmatrix}. \tag{4.5}$$

Para isso, primeiro escrevemos a matriz identidade, em (4.4), entre a matriz dos coeficientes e o vetor das incógnitas como

$$\begin{pmatrix} I & 0 \\ 0 & B^{-t} \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & B^t \end{pmatrix}.$$

Obtemos então, o seguinte sistema linear que é equivalente ao sistema estável (4.4):

$$\begin{pmatrix} -Z_E^k B^{-1} E & -X_B^k \\ Z_E^k & -X_E^k E^t B^{-t} \end{pmatrix} \begin{pmatrix} \Delta x_E^k \\ B^t \Delta y^k \end{pmatrix} = \begin{pmatrix} Z_B^k B^{-1} r_p + X_B^k (r_d)_B - M_B \\ X_E^k (r_d)_E - M_E \end{pmatrix}.$$

Podemos observar que o lado direito do sistema (4.4) não foi alterado. Finalmente, através de uma troca de linhas no sistema, e chamando $B^t \Delta y^k = \Delta \tilde{y}^k$, obtemos o sistema em (4.5). Esse é o conjunto de equações da abordagem do sistema estável que iremos considerar nos demais capítulos e seções. Dessa forma, quando nos referirmos à abordagem sistema estável, estaremos nos referindo a esse sistema em particular.

Qualquer problema linear pode ser reescrito como um problema onde a matriz das restrições possui a forma $(I \ E)$. Porém, esse procedimento pode aumentar a dimensão do problema de forma indesejável. Por isso, precisamos de uma matriz B de forma que o sistema estável possa ser resolvido de maneira eficiente, pelo menos próximo do conjunto solução. A matriz B pode ser obtida, por exemplo, da mesma forma como visto na Seção 3.3.1 para o pré-condicionador separador.

A Seção 4.3 apresenta alguns resultados para a resolução de sistemas como em (4.5). Esses sistemas são chamados de sistemas por blocos.

4.3 Resolvendo Sistemas por Blocos

Seja um sistema por blocos geral,

$$\begin{pmatrix} M_1 & M_2 \\ M_3 & M_4 \end{pmatrix} \begin{pmatrix} \Delta w_1 \\ \Delta w_2 \end{pmatrix} = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix}, \tag{4.6}$$

onde M_1 e M_4 são quadradas e invertíveis.

Seja $\bar{M}$ a matriz do sistema (4.6). Podemos destacar algumas propriedades do sistema que nos garantem resultados, como por exemplo a existência e unicidade da solução:

1. $\bar{M}$ é invertível $\Leftrightarrow S_1 = I - M_4^{-1}M_3M_1^{-1}M_2$ é invertível.
2. $\bar{M}$ é invertível $\Leftrightarrow S_2 = I - M_1^{-1}M_2M_4^{-1}M_3$ é invertível.
3. $(M_4 - M_3M_1^{-1}M_2)^{-1} = M_4^{-1} + M_4^{-1}M_3(M_1 - M_2M_4^{-1}M_3)^{-1}M_2M_4^{-1}$.
4. $\bar{M} = \begin{pmatrix} I & O \\ M_3M_1^{-1} & I \end{pmatrix} \begin{pmatrix} M_1 & 0 \\ 0 & M_4 - M_3M_1^{-1}M_2 \end{pmatrix} \begin{pmatrix} I & M_1^{-1}M_2 \\ 0 & I \end{pmatrix}$.
5. $\bar{M} = \begin{pmatrix} I & M_2M_4^{-1} \\ 0 & I \end{pmatrix} \begin{pmatrix} M_1 - M_2M_4^{-1}M_3 & 0 \\ 0 & M_4 \end{pmatrix} \begin{pmatrix} I & 0 \\ M_4^{-1}M_3 & I \end{pmatrix}$.
6. Se $\bar{M}$ é invertível, então a única solução do sistema (4.6) é $(\Delta w_1^*, \Delta w_2^*)$, onde

$$\begin{aligned} \Delta w_2^* &= (S_1)^{-1}M_4^{-1}(p_2 - M_3M_1^{-1}p_1) \\ &= M_4^{-1}p_2 - M_4^{-1}M_3(S_2)^{-1}M_1^{-1}(p_1 - M_2M_4^{-1}p_2), \\ \Delta w_1^* &= -M_1^{-1}M_2(S_1)^{-1}M_4^{-1}(p_2 - M_3M_1^{-1}p_1) + M_1^{-1}p_1 \\ &= (S_2)^{-1}M_1^{-1}(p_1 - M_2M_4^{-1}p_2). \end{aligned}$$

Essas propriedades podem ser encontradas em [24].

O sistema (4.6) pode ser resolvido por diferentes métodos iterativos de ponto fixo, pois a matriz $\bar{M}$ pode ser escrita como $P - T$ com P invertível, conforme as matrizes D_1, D_2, T_1 e T_2 da seção 4.4. Aplicando o método de Gauss-Seidel no sistema (4.6), obtemos o seguinte método,

Método 4.1 (Bloco iterativo).

1. Escolha Δw_1^0 inicial
2. Para $j = 0, 1, ...,$ até se obter a convergência.
3. Resolva $M_4\Delta w_2^j = p_2 - M_3\Delta w_1^j$
4. Resolva $M_1\Delta w_1^{j+1} = p_1 - M_2\Delta w_2^j$.

De maneira análoga, podemos iniciar o método com Δw_2^0 e inverter os passos 3 e 4. Chamamos esse método de *método bloco iterativo*. Podemos encontrar mais resultados sobre o método de Gauss-Seidel em livros de Cálculo Numérico, como por exemplo [56].

Agora, se as matrizes M_2 e M_3 são próximas da matriz nula, é natural pensarmos que o método bloco iterativo converge para uma solução do sistema linear. Esse resultado é obtido conforme o seguinte teorema.

Teorema 4.1. *Sejam o método bloco iterativo para resolver o sistema linear (4.6) e* $\Delta w^k = \begin{pmatrix} \Delta w_1^k \\ \Delta w_2^k \end{pmatrix}$ *a sequência gerada pelo método. Seja também,* $M = M_1^{-1}M_2$ *e* $N = M_4^{-1}M_3$. *Se* $||MN|| < 1$ *ou* $||NM|| < 1$ *então existe uma única solução* $\Delta w^* = \begin{pmatrix} \Delta w_1^* \\ \Delta w_2^* \end{pmatrix}$ *de (4.6) tal que* $\{\Delta w^k\}$ *converge para* Δw^*. *Se* $C = max(||MN||, ||NM||) < 1$ *então a convergência é pelo menos Q-linear com fator C.*

Demonstração: Seja $||MN|| < 1$. Então, $S_2 = I - MN$ é invertível e, de acordo com a propriedade 2, existe uma única solução para (4.6).

A outra parte do teorema pode ser demonstrada observando que

$$\begin{aligned} \Delta w_2^k &= M_4^{-1}p_2 - M_4^{-1}M_3\Delta w_1^k \text{ e} \\ \Delta w_1^{k+1} &= M_1^{-1}p_1 - M_1^{-1}M_2\Delta w_2^k \quad \text{para todo k.} \end{aligned}$$

Então

$$\Delta w_1^j = (MN)^j\Delta w_1^0 + \sum_{k=0}^{j-1}(MN)^k v$$

onde $v = M_1^{-1}p_1 - M_1^{-1}M_2M_4^{-1}p_2$. Portanto, $\Delta w_1^j - \Delta w_1^{j-1} = MN(\Delta w_1^{j-1} - \Delta w_1^{j-2})$ e Δw_1^k converge se $||MN|| < 1$. Como consequência, Δw_2^k também converge. Do modo como estão definidas as duas sequências, podemos ver que o ponto limite satisfaz a equação (4.6). De maneira análoga, o resultado é demonstrado para o caso $||NM|| < 1$.

Para demonstrarmos a taxa de convergência, note que $\Delta w_1^* = (I - MN)^{-1}v$. Então,

$$\Delta w_1^k - \Delta w_1^* = \left(\sum_{j=0}^{k-1}(MN)^j - (I - MN)^{-1}\right)v + (MN)^k\Delta w_1^0.$$

Mas,

$$\begin{aligned} \sum_{j=0}^{k-1}(MN)^j - (I - MN)^{-1} &= \sum_{j=0}^{k-1}((MN)^j(I - MN) - I)(I - MN)^{-1} \\ &= -(MN)^k(I - MN)^{-1}. \end{aligned}$$

Então, $\Delta w_1^k - \Delta w_1^* = (MN)(\Delta w_1^{k-1} - \Delta w_1^*)$. Sabendo que, $\Delta w_2^k = M_4^{-1}p_2 - M_4^{-1}M_3\Delta w_1^k$ temos,

$$\begin{aligned}\Delta w_2^k - \Delta w_2^* = M_4^{-1}M_3(\Delta w_1^* - \Delta w_1^{k-1}) &= -M_4^{-1}M_3(MN(\Delta w_1^{k-1} - \Delta w_1^*)) \\ &= -NMM_4^{-1}M_3(\Delta w_1^{k-1} - \Delta w_1^*) = NM(\Delta w_2^{k-1} - \Delta w_2^*).\end{aligned}$$

■

É interessante destacar nesse teorema que $||MN||$, ou $||NM||$, é um limite superior para o erro cometido.

O próximo teorema nos fornece uma condição fraca de convergência.

Teorema 4.2. *Se $\rho(MN) < 1$, ou $\rho(NM) < 1$, onde ρ é o raio espectral (isto é, o maior autovalor em módulo), então existe uma única solução $w^* = \begin{pmatrix} w_1^* \\ w_2^* \end{pmatrix}$ de (4.6) tal que $\lim_{k\to\infty}\{\Delta w^k\} = w^*$.*

Demonstração: Vamos considerar o caso em que $\rho(MN) < 1$. A existência da solução w^* vem do fato de $I - MN$ ser invertível. A convergência pode ser obtida através da demonstração do teorema anterior e, utilizando as propriedades de convergência de séries geométricas. Temos,

$$\Delta w_1^j = (MN)^j \Delta w_1^0 + \sum_{k=0}^{j-1}(MN)^k v$$

onde $v = M_1^{-1}p_1 - M_1^{-1}M_2M_4^{-1}p_2$. Como $\rho(MN) < 1$ então a série $\sum(MN)^k$ converge para $(I-MN)^{-1}$ e $(MN)^k$ converge para matriz nula quando k tende ao infinito. Portanto, tomando o limite quando $j \to \infty$ na equação anterior, obtemos $\lim_{k\to\infty}\Delta w_1^k = (I - MN)^{-1}v = w_1^*$. De maneira análoga, podemos provar que $\{\Delta w_2^k\}$ converge para w_2^*. ■

4.4 Resolvendo o sistema estável

Observe que tanto o pré-condicionador separador quanto o sistema estável necessitam do cálculo da matriz não singular B. Portanto, as ferramentas apresentadas em [51] para encontrá-la, também podem ser utilizadas na abordagem do sistema estável.

Para trabalharmos com uma notação mais simples e fácil, omitiremos o índice de iteração nessa seção e, além disso, reescreveremos o sistema estável (4.5) da seguinte forma,

$$\begin{pmatrix} Z_E & -X_E V^t \\ -Z_B V & -X_B \end{pmatrix}\begin{pmatrix} \Delta\tilde{x}_E \\ \Delta\tilde{y} \end{pmatrix} = \begin{pmatrix} r_2 \\ r_1 \end{pmatrix}, \tag{4.7}$$

onde $V = B^{-1}E$ e

$$\begin{pmatrix} r_2 \\ r_1 \end{pmatrix} = \begin{pmatrix} X_E(r_d)_E - Z_E X_E e_E + \mu e_E \\ Z_B B^{-1} r_p + X_B(r_d)_B - Z_B X_B e_B + \mu e_B \end{pmatrix}. \tag{4.8}$$

Resolvendo o sistema (4.7), podemos obter a solução do sistema original usando as equações: $\Delta x_E = \Delta\tilde{x}_E$, $\Delta y = B^{-t}\Delta\tilde{y}$, $\Delta x_B = B^{-1}(r_p - E\Delta x_E)$ e $\Delta z = r_d - A^t\Delta y$.

A matriz $\bar{M}$ do sistema estável (4.7) é uma matriz por blocos que pode ser escrita como

$$\bar{M} = D_1 + D_2 = \begin{pmatrix} Z_E & 0 \\ 0 & -X_B \end{pmatrix} + \begin{pmatrix} 0 & -X_E V^t \\ -Z_B V & 0 \end{pmatrix}, \tag{4.9}$$

onde D_1 é uma matriz diagonal e invertível. De forma alternativa,

$$\bar{M} = T_1 + T_2 = \begin{pmatrix} Z_E & -X_E V^t \\ 0 & -X_B \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ -Z_B V & 0 \end{pmatrix},$$

onde T_1 é uma matriz triangular invertível.

O sistema (4.7) possui a mesma estrutura do sistema (4.6). Portanto, podemos utilizar o método bloco iterativo para resolvê-lo. Aqui, as matrizes M_1 e M_4 são, respectivamente, Z_E e $-X_B$. Adicionalmente, $M_2 = -X_E V^t$ e $M_3 = -Z_B V$. Chamando $D = Z^{-1}X$, $D_E = Z_E^{-1}X_E$ e $D_B = Z_B^{-1}X_B$, então $MN = -D_E V^t D_B^{-1} V$ e $NM = -D_B^{-1} V D_E V^t$. Dessa forma, dos teoremas anteriores, se $||D_E||$ ou $||D_B||$ for suficientemente pequena, ou seu raio espectral é menor que um, então para alguma matriz V o método converge.

Note que, próximo de uma solução, pelo menos $n-m$ valores de D estão próximos de zero. Então, com uma escolha adequada das colunas de A correspondentes a B, os valores diagonais de D_B^{-1} e D_E estão muito próximos de zero.

Temos então, o seguinte método para resolução do sistema (4.7).

Método 4.2 (Sistema linear estável).

1. Calcule a matriz de permutação P dos índices $\{1, ..., n\}$ correspondentes à ordem decrescente da norma das colunas da matriz AD^{-1}.

2. Encontre as primeiras m colunas linearmente independentes de A, seguindo o ordenamento calculado.

3. Sejam B a matriz invertível formada pelas colunas do passo anterior e E a matriz formada pelas demais colunas de A.

4. Calcule $P^t x$, $P^t z$ e $P^t c$. Portanto, x_B e z_B são as componentes de x e z correspondentes às colunas linearmente independentes de A utilizadas para formar a matriz B; x_E e z_E são as componentes de x e z correspondentes às colunas de A que foram utilizadas para formar a matriz E.

5. Calcule r_1 e r_2 definidos em (4.8).

6. Calcule $|| - Z_B E Z_E^{-1} r_2 + r_1||$ e $||X_E E^t X_B^{-1} r_1 - r_2||$.

7. **Se** $|| - Z_B E Z_E^{-1} r_2 + r_1|| < ||X_E E^t X_B^{-1} r_1 - r_2||$, **então**

   Seja $\Delta x_E^0 = 0$ e, para $j = 0, 1, 2, ...$, até convergir, faça

   (a) $\Delta \tilde{y}^{j+1} = -X_B^{-1}(r_1 + Z_B B^{-1} E \Delta x_E^j)$

   (b) $\Delta x_E^{j+1} = Z_E^{-1}(r_2 + X_E E^t B^{-t} \Delta \tilde{y}^{j+1})$

   Senão

   Seja $\Delta \tilde{y}^0 = 0$ e, para $j = 0, 1, 2, ...$, até convergir, faça

   (a) $\Delta x_E^{j+1} = Z_E^{-1}(r_2 + X_E E^t B^{-t} \Delta \tilde{y}^j)$

   (b) $\Delta \tilde{y}^{j+1} = -X_B^{-1}(r_1 + Z_B B^{-1} E \Delta x_E^{j+1})$

As opções no passo 7 são motivadas pela proposição abaixo e podem ser facilmente derivadas da forma como as sequências são geradas.

Proposição 4.1. *Se Δw_1^0 é escolhido próximo de zero, então*

$$||\Delta w_2^1 - \Delta w_2^0|| \leq ||D_B^{-1}||||B||max(1/z_E)||X_E V^t X_B^{-1} r_1 - r_2||.$$

Se Δw_2^0 é escolhido próximo de zero, então

$$||\Delta w_1^1 - \Delta w_1^0|| \leq ||D_E||||B||max(1/x_B)|| - Z_B V Z_E^{-1} r_2 + r_1||.$$

Então, escolhemos iniciar com Δw_1^0 ou Δw_2^0 de acordo com o valor das normas $||X_E V^t X_B^{-1} r_1 - r_2||$ e $||-Z_B V Z_E^{-1} r_2 + r_1||$, que são fáceis de se calcular. Para uma maior eficiência no cálculo das normas, assumimos que $V = E$, isto é que a matriz B é a matriz identidade. Assim, utilizando as hipóteses dos teoremas anteriores, temos a convergência desse método.

4.4.1 A matriz estável próxima de uma solução

Vimos que, em cada iteração do método de pontos-interiores, a matriz estável M pode ser escrita como $M = D_1 + D_2$ com

$$D_1 = \begin{pmatrix} Z_E & 0 \\ 0 & -X_B \end{pmatrix} \text{ invertível, e } D_2 = \begin{pmatrix} 0 & -X_E V^t \\ -Z_B V & 0 \end{pmatrix}.$$

Portanto, se as variáveis x_E e z_B estão suficientemente próximas de zero, a matriz estável se aproxima de uma matriz diagonal. Vejamos a proposição abaixo.

Proposição 4.2. *Seja $M = D_1 + D_2$ a matriz estável dada em (4.9). Então, $||M - D_1|| \leq max(z_B, x_E)||V||$. Além disso, se $\Delta\widetilde{w}$ é a solução do sistema $D_1\Delta\widetilde{w} = r$ então $||M\Delta\widetilde{w} - r|| \leq \frac{max(x_E, z_B)}{min(z_E, x_B)}||r||||V||$.*

Demonstração: A demonstração é feita através da norma $||D_2 y||^2$ para qualquer y. Ou seja,

$$\begin{aligned} ||(M - D_1)y||^2 &= ||D_2 y||^2 = \left|\left| \begin{pmatrix} 0 & -X_E V^t \\ -Z_B V & 0 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \right|\right|^2 = \\ &= ||-X_E V^t y_1||^2 + ||Z_B V y_2||^2 \leq max(||X_E||^2, ||Z_B||^2)||V||^2(||y_1||^2 + ||y_2||^2) = \\ &= max(x_E^2, z_B^2)||V||^2||y||^2. \end{aligned}$$

Para a segunda parte da demonstração utilizamos o fato que, se $D_1\Delta\tilde{w} = r$, então $||M\Delta w - r|| = ||D_2 D_1^{-1} r|| \leq ||D_2||||D_1^{-1}||||r||$. ∎

Vejamos agora o comportamento do sistema estável em uma solução do problema linear (2.1). Se o problema é primal-dual não degenerado, então existe uma única solução (x^*, y^*, z^*) para os problemas primal e dual e, os conjuntos de índices

$$\begin{aligned} \mathcal{B} &= \{j \in \{1, 2, \ldots, n\} / x_j^* \neq 0\} \\ \mathcal{N} &= \{j \in \{1, 2, \ldots, n\} / z_j^* \neq 0\} \end{aligned}$$

formam uma partição do conjunto de índices $\{1, 2, \ldots, n\}$, isto é, $\mathcal{B} \cup \mathcal{N} = \{1, 2, \ldots, n\}$ e $\mathcal{B} \cap \mathcal{N} = \emptyset$. Além disso, o conjunto $\mathcal{B}$ possui exatamente m índices, enquanto o conjunto $\mathcal{N}$ possui exatamente $n - m$ índices. Os m índices do conjunto $\mathcal{B}$ são os que formam a partição x_B^*, $(x_B^* > 0)$ e, os $n - m$ índices do conjunto $\mathcal{N}$ são os que formam a partição z_E^*, $(z_E^* > 0)$. Portanto, se $V = B^{-1}E$, a matriz do sistema estável, na solução, é igual à matriz diagonal

$$\begin{pmatrix} Z_E^* & 0 \\ 0 & -X_B^* \end{pmatrix}.$$

Observe que esse resultado é válido para qualquer permutação das colunas da matriz B. Portanto, no caso não degenerado, devido a convergência Q-superlinear das variáveis nulas x_E e z_B, o sistema estável é próximo de um sistema diagonal, quando próximo de uma solução, e o método bloco iterativo necessitará de algumas poucas iterações para resolvê-lo.

No entanto, para os casos degenerados, a situação é diferente. Vejamos o caso em que apenas o problema dual é degenerado, ou seja, existem infinitas soluções para o problema primal e, para qualquer solução primal-dual $w^* = (x^*, y^*, z^*)$, existe uma única partição $\mathcal{B} \cup \mathcal{N} = \{1, ..., n\}$ tal que $|\mathcal{B}| = m$ e a correspondência $x^*_B > 0$, $z^*_E \geq 0$ e $A = (B \ \ E)P$. Portanto, nesse caso, existe pelo menos uma componente de z^*_E igual a zero. Então, podemos escrever z^*_E como $((z^*_E)_1; (z^*_E)_2)$, com $(z^*_E)_1 > 0$. Então, $V = B^{-1}E = \begin{pmatrix} V_1 & V_2 \end{pmatrix}$ onde V_1 são as colunas de V correspondentes a $(z^*_E)_1$. A matriz do sistema estável, calculado em w^*, é

$$\begin{pmatrix} (Z^*_E)_1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -X^*_B \end{pmatrix}.$$

Para o caso em que apenas o problema primal é degenerado, a situação é similar. Existem infinitas soluções para o problema dual e, para qualquer solução primal-dual, existe uma única partição $\mathcal{B} \cup \mathcal{N} = \{1, ..., n\}$ tal que $|\mathcal{N}| = n - m$ e a correspondência $x^*_B \geq 0$, $z^*_E > 0$ e $A = (B \ \ E)P$. Portanto, nesse caso, existe pelo menos uma componente de x^*_B igual a zero. Então, podemos escrever as soluções z^*_B como $((x^*_B)_1; (x^*_B)_2)$, com $(x^*_B)_1 > 0$. então, $V = B^{-1}E = \begin{pmatrix} V_1 \\ V_2 \end{pmatrix}$, onde V_1 são linhas de V correspondentes a $(x^*_B)_1$. A matriz do sistema estável, calculado em w^*, é

$$\begin{pmatrix} Z^*_E & 0 & 0 \\ 0 & -(X^*_B)_1 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Como os métodos de pontos-interiores primal-dual abordam soluções complementares estritas, temos, pela continuidade, que um dos blocos fora da diagonal da matriz, na abordagem do sistema estável, é próximo da matriz nula, próximo do conjunto solução. Isso implica que, D_B^{-1} ou D_E está próximo da matriz nula. No entanto, isso não nos garante a convergência do método bloco iterativo, a menos que a norma $||D_B^{-1} V D_E V^t||$ seja pequena.

4.4.2 Resolvendo problemas degenerados

Observe que, para os problemas degenerados, próximo do conjunto solução, o passo 7 do método está mais sujeito a erros, pois algumas coordenadas das variáveis z_E e/ou x_B podem estar próximas de zero e a coordenada correspondente do lado direito pode ser uma valor grande. De fato, a matriz da abordagem do sistema estável se torna próxima de ser singular, o que faz com que o sistema estável, nesse caso, seja de difícil solução. Para contornarmos essa situação, perturbamos a matriz do sistema estável, M, adicionando uma matriz diagonal δI, com $\delta > 0$, quando qualquer um dos elementos da diagonal é pequeno e o *gap* é menor que uma tolerância previamente escolhida. É claro que, dessa forma, estamos calculando uma outra direção de busca Δw_p^k, mas se δ é suficientemente pequeno, as duas direções devem estar próximas. Vejamos agora, como podemos encontrar esse parâmetro δ e o erro, por ele, provocado.

Inicialmente, assumimos que $|| - D_E V^t D_B^{-1} V|| \geq 1$ e $|| - D_B^{-1} V D_E V^t|| \geq 1$, ou seja, os valores $máx(z_B x_B^{-1})$ ou $máx(x_E z_E^{-1})$ são grandes. Supondo que os dois valores sejam grandes. Isso corresponde ao caso em que os problemas primal e dual são degenerados. Nesse caso, alguma componente de x_B está mais próxima de zero do que a componente correspondente em z_B e, alguma componente de z_E está mais próxima de zero do que a correspondente em x_E.

Gostaríamos de escolher α tal que,

1. A matriz perturbada $M + \alpha I$ seja bem condicionada.
2. O erro $||M\Delta\widetilde{w} - r||$ seja o menor possível, com $(M + \alpha I)\Delta\widetilde{w} = r$.
3. O método bloco iterativo convirja, isto é,

$$\begin{aligned} \rho((Z_E + \alpha I)^{-1} X_E V^t (X_B + \alpha I)^{-1} Z_B V) &< 1, \text{ ou} \\ \rho((X_B + \alpha I)^{-1} Z_B V (Z_E + \alpha I)^{-1} X_E V^t) &< 1. \end{aligned} \tag{4.10}$$

Observe que $\Delta\widetilde{w} = (M + \alpha I)^{-1} r$ e, $M\Delta\widetilde{w} = r - \alpha\Delta\widetilde{w}$. Logo, $||M\Delta\widetilde{w} - r|| \leq \alpha||r||||(M + \alpha I)^{-1}||$. Portanto, para podermos satisfazer a condição 2, precisamos limitar $||(M + \alpha I)^{-1}||$ e, para satisfazermos a condição 1, precisamos limitar $||M + \alpha I||$. Vejamos alguns resultados que nos permitem limitar essas normas.

Lema 4.1. *Seja $A = B + C$ com A e B invertíveis satisfazendo $||I - AB^{-1}|| < 1$. Então $||A^{-1}|| \leq ||B^{-1}||/(1 - ||CB^{-1}||)$.*

Demonstração: $A^{-1} - B^{-1} = A^{-1}(I - AB^{-1})$. Então,

$$||A^{-1}|| - ||B^{-1}|| \leq ||A^{-1} - B^{-1}|| \leq ||A^{-1}||||I - AB^{-1}||.$$

Portanto, $||A^{-1}|| \leq ||B^{-1}||/(1 - ||I - AB^{-1}||)$. Mas, $I - AB^{-1} = I - (B + C)B^{-1} = CB^{-1}$. ■

Lema 4.2. *Seja $A = B + C$ com A e B invertíveis. Então, $||A^{-1}|| \geq ||B^{-1}||(1 - ||C(I + B^{-1}C)^{-1}B^{-1}||)$ e $||A^{-1}|| \geq ||B^{-1}||/||I + B^{-1}C||$.*

Demonstração: A inversa de A pode ser escrita como $A^{-1} = B^{-1} - B^{-1}C(I + B^{-1}C)^{-1}B^{-1}$. Então, $||A^{-1}|| \geq ||B^{-1}|| - ||D||$ com $D = B^{-1}C(I + B^{-1}C)^{-1}B^{-1}$. Agora, podemos limitar $||D||$ por $||B^{-1}||||C(I + B^{-1}C)^{-1}B^{-1}||$ e, então, obtemos a primeira desigualdade. A segunda desigualdade é obtida desenvolvendo a primeira, ou utilizando a igualdade $A^{-1} = (B + C)^{-1}$ e, então $||A^{-1}|| \geq ||B^{-1}||/||I + B^{-1}C||$. ■

Assim, podemos escrever, $M_\alpha = M + \alpha I$ e $D_{1_\alpha} = D_1 + \alpha I$. Então, pelo Lema 4.1 temos que $||M_\alpha^{-1}|| \leq ||(D_{1_\alpha})^{-1}||/(1 - ||D_2||||D_{1_\alpha}||)$, se α é escolhido tal que $1 - ||D_2||||D_{1_\alpha}^{-1}|| > 0$.

4.4.3 Perturbações

Para os problemas degenerados, utilizamos três abordagens para a perturbação dos vetores x_B, z_L e w_U. A primeira perturbação, α_1, consiste em dividir a menor coordenada desses vetores, pela norma do vetor à direita no sistema (4.7), ou seja

$$\alpha_1 = min(x_B, z_E) \Big/ \left|\left|\begin{pmatrix} r_1 \\ r_2 \end{pmatrix}\right|\right|. \tag{4.11}$$

Considerando apenas os elementos da diagonal principal na matriz do sistema estável (4.7), podemos escrevê-lo da seguinte forma:

$$\begin{pmatrix} Z_E & \\ & -X_B \end{pmatrix} \begin{pmatrix} \Delta\tilde{y} \\ \Delta\tilde{x}_E \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix}.$$

Assim, a perturbação α_1 surgiu com a idéia de se tomar o menor dos elementos da diagonal principal e dividir pela norma do vetor à direita.

A segunda perturbação, α_2, surge da desigualdade (4.10). Tomando $V = [I \;\; 0]P$, onde P é uma matriz de permutação na correspondente dimensão, e assumindo que x_B e z_E possuem todas as coordenadas nulas, podemos reescrever (4.10) da seguinte forma:

$$\rho\left(\frac{1}{\alpha}X_E V^t \frac{1}{\alpha}Z_B V\right) < 1.$$

Agora, como o maior autovalor, ρ, de uma matriz diagonal é o maior elemento de sua diagonal principal, então, obtemos α_2 como sendo o menor β satisfazendo

$$\beta^2 > max\left(X_E, Z_B\right).$$

Portanto, α_2 é calculado como

$$\alpha_2 = max\left(X_{E_{1:\mathrm{m}}} Z_B\right)^{\frac{1}{2}},$$

se $n - m > m$, ou

$$\alpha_2 = max\left(X_E Z_B(1 : n - m)\right)^{\frac{1}{2}},$$

se $n - m < m$. Aqui, m é a dimensão da matriz B.

A terceira perturbação, α_3, também surge da desigualdade (4.10). Ainda assumindo que x_B e z_E possuem todas as coordenadas nulas mas, agora, tomando B como a matriz identidade, ou seja, tomando $V = E$, obtemos então,

$$\rho\left(\frac{1}{\alpha} X_E E^t \frac{1}{\alpha} Z_B E\right) < 1 \tag{4.12}$$

Através dos Círculos de Gershgorin [24], estimamos o maior autovalor da matriz acima como sendo o maior elemento de sua diagonal principal, ϵ. Dessa forma, obtemos a seguinte perturbação,

$$\alpha_3 = \epsilon \times \left|\left|\begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}\right|\right|$$

se $\epsilon > 1$. Caso contrário

$$\alpha_3 = \epsilon \Big/ \left|\left|\begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}\right|\right|.$$

Observando os experimentos numéricos, verificamos que α_3 produz uma perturbação grande comparada com as outras duas perturbações. Assim, em busca de uma melhor perturbação unimos as três perturbações em uma única perturbação, obtivemos então a perturbação α_4.

Primeiramente, α_4 consiste na escolha de qual das três perturbações anteriores devemos utilizar. Essa escolha depende da norma do vetor à direita do sistema estável. Caso essa norma seja maior, ou igual a 1, utilizamos a perturbação α_1, caso a norma seja menor que 1, mas maior que 10^{-8}, utilizamos a perturbação α_2 e, caso contrário, utilizamos a perturbação α_3.

Em seguida, verificamos o tamanho da perturbação. Se a perturbação for menor que 1 e maior que 10^{-10}, então α_4 é a perturbação calculada. Caso a perturbação seja maior que 1, então tomamos $\alpha_4 = 1$. Caso a perturbação seja menor que 10^{-10}, então tomamos $\alpha_4 = 10^{-10}$.

Em seguida, caso o sistema estável não seja resolvido em 100 iterações, multiplicamos α_4 por 10 e iniciamos o método novamente. Essa etapa acontece até que o sistema seja resolvido, ou que obtenhamos $\alpha_4 = 1$.

Em nossos experimentos numéricos, os vetores x_B e z_E são perturbados com α_4 se alguma de suas coordenadas é menor que 10^{-8}.

No capítulo seguinte, veremos como aplicar o sistema estável para problemas canalizados.

Capítulo 5

Sistema Estável para Variáveis Canalizadas

A grande maioria dos problemas de programação linear que modelam situações reais, possuem variáveis canalizadas. Ou seja, variáveis com um limite superior e um limite inferior. Por isso, a abordagem do sistema linear estável deve ser desenvolvido também, para resolver problemas com esse tipo de variáveis.

Nesse capítulo, apresentamos a abordagem do sistema linear estável para problemas de programação linear com variáveis do tipo canalizadas. Apresentamos, também, um método eficiente para resolver essa nova versão do sistema linear estável. Antes, porém, apresentamos o método preditor-corretor para problemas de variáveis canalizadas.

5.1 Variáveis canalizadas

Um problema com variáveis canalizadas é um problema de programação linear na forma:

$$\begin{aligned} min \quad & c^t x \\ \text{s.a} \quad & Ax = b \\ & l \leq x \leq u, \qquad \text{onde l} < \text{u}. \end{aligned} \tag{5.1}$$

Podemos transformar este problema na forma padrão, como em (2.1), e aplicar todos os métodos vistos no capítulo 2, porém essa abordagem não é eficiente. Ao transformar (5.1) na forma padrão, o número de restrições e variáveis fica muito grande em relação ao problema original.

Para simplificar o problema, fazemos uma mudança de variáveis de tal forma que o limite inferior seja nulo, isto é,

$$0 \leq x - l \leq u - l \quad \Rightarrow \quad 0 \leq \tilde{x} \leq \tilde{u},$$

onde $x = \tilde{x} + l$ e $u = \tilde{u} + l$

Agora, substituindo x por $\tilde{x} + l$ em (5.1), temos:

$$\begin{aligned} min \quad & c^t\tilde{x} + c^t l \\ \text{s.a} \quad & A\tilde{x} = b - Al = \tilde{b} \\ & 0 \leq \tilde{x} \leq \tilde{u}. \end{aligned}$$

Dessa forma, resolver (5.1) é equivalente a resolver o problema,

$$\begin{aligned} min \quad & c^t x \\ \text{s.a} \quad & Ax = b \\ & 0 \leq x \leq u. \end{aligned}$$

Escrevendo o problema acima na forma padrão, temos:

$$\begin{aligned} min \quad & c^t x \\ \text{s.a} \quad & Ax \quad = b \\ & x + s = u \\ & x, s \geq 0. \end{aligned} \tag{5.2}$$

A matriz de restrições na forma padrão possui dimensão $(m+n) \times (n+n)$ e uma base teria dimensão $(m+n) \times (m+n)$. Dessa forma, a dimensão do problema aumenta muito o que faz com que os métodos percam eficiência.

Podemos contornar essa situação, definindo solução básica factível para problemas escritos na forma (5.2), com relação a matriz A. Assim, uma solução básica factível de um problema de programação linear é uma solução factível em que $n - m$ variáveis são iguais a seu limite inferior, ou superior, e as demais m variáveis, chamadas variáveis básicas, correspondem a colunas linearmente independentes da matriz de restrições, com $0 \leq x_B \leq u$.

A forma padrão do problema dual associado ao problema (5.2) é escrita como,

$$\begin{aligned} max \quad & b^t y - u^t w \\ \text{s.a} \quad & A^t y - w + z = c \\ & w, z \geq 0 \\ & y \quad \text{livre.} \end{aligned} \tag{5.3}$$

Agora, vejamos as condições de otimalidade, (KKT), para os problemas (5.2) e (5.3):

$$\begin{aligned} \text{Factibilidade primal} \quad & \begin{cases} b - Ax = 0 \\ u - x - s = 0 \\ x, s \geq 0 \end{cases} \\ \text{Factibilidade dual} \quad & \begin{cases} c - A^t y - z + w = 0 \\ z, w \geq 0 \end{cases} \\ \text{Complementaridade} \quad & \begin{cases} XZe = 0 \\ SWe = 0 \end{cases} \end{aligned} \tag{5.4}$$

Aplicando o método de Newton às condições de otimalidade, obtemos o seguinte sistema linear:

$$\begin{pmatrix} 0 & 0 & A^t & I & -I \\ A & 0 & 0 & 0 & 0 \\ I & I & 0 & 0 & 0 \\ Z & 0 & 0 & X & 0 \\ 0 & W & 0 & 0 & S \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta y \\ \Delta z \\ \Delta w \end{pmatrix} = - \begin{pmatrix} A^t y + z - w - c \\ Ax - b \\ x + s - u \\ XZe \\ SWe \end{pmatrix}. \tag{5.5}$$

As condições de otimalidade e o sistema acima se referem a problemas de programação linear em que todas as variáveis são canalizadas, ou seja, limitadas. Na prática, muitos problemas de programação linear possuem apenas algumas variáveis canalizadas. E, para tais problemas, o sistema (5.5) pode ser escrito como,

$$\begin{pmatrix} 0 & 0 & A^t & I & -J^t \\ A & 0 & 0 & 0 & 0 \\ J & I & 0 & 0 & 0 \\ Z & 0 & 0 & X & 0 \\ 0 & W & 0 & 0 & S \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta y \\ \Delta z \\ \Delta w \end{pmatrix} = - \begin{pmatrix} r_d \\ r_p \\ r_u \\ r_a \\ r_b \end{pmatrix} = - \begin{pmatrix} A^t y + z - J^t w - c \\ Ax - b \\ Jx + s - u \\ XZe \\ SWe \end{pmatrix}, \tag{5.6}$$

onde J é a matriz formada pelas linhas da matriz identidade, de ordem n, que correspondem às variáveis canalizadas. O número de linhas da matriz J é igual ao número de variáveis canalizadas.

Vimos na seção 2.5 duas abordagens para calcular as direções de busca no métodos do tipo primal-dual. São as abordagens sistema aumentado e sistema de equações normais. Essas duas abordagens também podem ser estendidas aos problemas de programação linear com variáveis canalizadas.

Considerando os índices de iteração k, a abordagem sistema aumentado pode ser obtida para o sistema (5.6) eliminado as variáveis Δz^k, Δs^k e Δw^k de (5.6). Obtemos então,

$$\begin{pmatrix} -D^k & A^t \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x^k \\ \Delta y^k \end{pmatrix} = - \begin{pmatrix} r_d^k - (X^k)^{-1} r_c^k + (S^k)^{-1} r_b \\ r_p^k \end{pmatrix}, \tag{5.7}$$

onde $(D^k)^{-1} = (Z^k)^{-1} X^k + J^t (W^k)^{-1} S^k J$. O sistema (5.7) é chamado sistema aumentado para variáveis canalizadas. Para o sistema de equações normais temos,

$$S^k \Delta y^k = -r_p^k - A(D^k)^{-1}(r_d^k - (X^k)^{-1} r_c^k + (S^k)^{-1} r_b^k), \tag{5.8}$$

onde $S^k = A(D^k)^{-1} A^t$

5.2 Método preditor-corretor canalizado

Em 1992, Mehrotra [46] publicou um artigo com uma implementação extremamente eficiente que, ainda hoje, é a base para muitas implementações atuais de métodos de pontos interiores.

Apresentamos nessa seção os principais passos do método preditor-corretor de Mehrotra, para o caso em que apenas algumas variáveis são canalizadas. Podemos encontrar mais detalhes sobre esse método em Wright [64].

O método preditor-corretor de Mehrotra [46] é baseado no método de Newton para as condições de KKT (5.6), de forma que as variáveis (x, s, z, w) permaneçam sempre positivas, para incorporar o termo de centralização na direção de busca, e melhorar a ordem da precisão na qual a direção de busca se aproxima das equações não lineares, as equações de complementaridade.

A direção de busca em cada iteração do método preditor-corretor é obtido resolvendo dois sistemas lineares, que possuem a mesma matriz de coeficientes, mas diferentes vetores no lado direito do sistema.

Na seção 2.4, vimos o método preditor-corretor para problemas sem variáveis canalizadas. Na seção 5.1, vimos as condições de complementaridade para variáveis canalizadas, e a construção

da matriz de restrições para o caso em que, apenas, algumas variáveis são canalizadas. No método preditor-corretor para variáveis canalizadas, precisamos substituir o sistema (2.16) por

$$\begin{pmatrix} 0 & 0 & A^t & I & -J^t \\ A & 0 & 0 & 0 & 0 \\ J & I & 0 & 0 & 0 \\ Z^k & 0 & 0 & X^k & 0 \\ 0 & W^k & 0 & 0 & S^k \end{pmatrix} \begin{pmatrix} \Delta\tilde{x} \\ \Delta\tilde{s} \\ \Delta\tilde{y} \\ \Delta\tilde{z} \\ \Delta\tilde{w} \end{pmatrix} = - \begin{pmatrix} r_d^k \\ r_p^k \\ r_u^k \\ r_a^k \\ r_b^k \end{pmatrix}. \tag{5.9}$$

Para o cálculo da direção preditora, utilizamos

$$r_d^k = A^t y + z - w - c, \qquad r_p^k = Ax - b \qquad \text{e} \qquad r_u^k = x + s - u.$$

As demais componentes do lado direito do sistema são

$$r_a^k = X^k Z^k, \qquad \text{e} \qquad r_b^k = S^k W^k.$$

A segunda direção, $(\Delta\hat{x}, \Delta\hat{s}, \Delta\hat{y}, \Delta\hat{z}, \Delta\hat{w})$, é obtida resolvendo o sistema acima, mas com o seguinte vetor à direita,

$$r_d^k = 0, \qquad r_p^k = 0, \qquad r_u^k = 0, \qquad r_a^k = \Delta\tilde{X}\Delta\tilde{Z}e - \mu^k e, \qquad \text{e} \qquad r_b^k = \Delta\tilde{S}\Delta\tilde{W}e - \mu^k e,$$

onde μ^k é calculado como na seção (2.4).

A direção utilizada é a direção obtida pela soma da solução dos dois sistemas lineares, resolvidos anteriormente,

$$(\Delta x^k, \Delta s^k, \Delta y^k, \Delta z^k, \Delta w^k) = (\Delta\tilde{x}^k, \Delta\tilde{s}^k, \Delta\tilde{y}^k, \Delta\tilde{z}^k, \Delta\tilde{w}^k) + (\Delta\hat{x}^k, \Delta\hat{s}^k, \Delta\hat{y}^k, \Delta\hat{z}^k, \Delta\hat{w}^k).$$

Para a resolução dos sistema lineares, cálculo da primeira e da segunda direção, é utilizado o sistema de equações normais, seção 2.5.

O passo primal, α_p^k, e o passo dual, α_d^k, são calculados da seguinte forma:

$$\alpha_p^k = \gamma_p \times \alpha_{max,p}^k \qquad \text{e} \qquad \alpha_d^k = \gamma_d \times \alpha_{max,d}^k,$$

onde

$$\begin{aligned} \alpha_{max,p}^k &= inf\{\alpha \in [0,1] | (x,s) + \alpha(\Delta x, \Delta s) \geq 0\}, \\ \alpha_{max,d}^k &= inf\{\alpha \in [0,1] | (z,w) + \alpha(\Delta z, \Delta w) \geq 0\}, \end{aligned}$$

e γ_p e γ_d são calculados através de uma heurística de Mehrotra, descrita em [46].

O método converge quando os seguintes testes são satisfeitos:

$$\begin{aligned} \frac{||(r_p, r_a)||}{1 + ||(b^t, u^t)||} &\leq \epsilon_1, \\ \frac{||(r_d)||}{1 + ||c)||} &\leq \epsilon_2, \\ \frac{\left|c^t x - \left(b^t y - u^t s\right)\right|}{1 + ||1 + |c^t x|||} &\leq \epsilon_3, \end{aligned} \tag{5.10}$$

onde ϵ_1, ϵ_2 e ϵ_3 são três valores de tolerância com valor padrão igual a 10^{-8}.

5.3 Sistema estável para variáveis canalizadas

Assim como na seção 4.1, a abordagem do sistema estável para problemas com variáveis canalizadas pode ser obtido através do processo de eliminação de blocos, porém, aqui, aplicamos o processo de eliminação de blocos na equação (5.6).

Dessa forma, o primeiro passo consiste em "mover W", ou seja, multiplicamos a terceira linha por $-W$ e somamos com a última linha. Obtemos então,

$$\begin{pmatrix} 0 & 0 & A^t & I & -J^t \\ A & 0 & 0 & 0 & 0 \\ J & I & 0 & 0 & 0 \\ Z & 0 & 0 & X & 0 \\ -WJ & 0 & 0 & 0 & S \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta y \\ \Delta z \\ \Delta w \end{pmatrix} = - \begin{pmatrix} r_d \\ r_p \\ r_u \\ r_a \\ r_c \end{pmatrix}, \tag{5.11}$$

onde $r_c = r_b - W r_u$.

Agora, assumindo que a matriz A possui a forma: $A = (B \quad L \quad U)$, onde B são as colunas referentes às variáveis básicas, L são as colunas referentes às variáveis que estão próximas de zero e U são as colunas referentes às variáveis que estão próximas de seu limite superior, para um determinado vetor x. Assim, a matriz J pode ser escrita,

$$J = \begin{pmatrix} J_{\tilde{B}} & & \\ & J_{\tilde{L}} & \\ & & J_{\tilde{U}} \end{pmatrix},$$

onde a dimensão da matriz S, e da matriz W, é igual a soma das dimensões $\tilde{B}$, $\tilde{L}$ e $\tilde{U}$. Assumimos aqui que os blocos $J_{\tilde{B}}$, $J_{\tilde{L}}$ e $J_{\tilde{U}}$ estão em suas formas escadas, ou formas reduzidas. Podemos encontrar mais sobre a forma escada de uma matriz em [8].

Dessa forma, o sistema (5.11) é igual ao sistema formado pela matriz A_1 e pelo vetor d_1, onde

$$A_1 = \begin{pmatrix}
 & & & & & & & & & & & \\
 & & & & & & B^T & I_{\tilde{B}} & & -J_{\tilde{B}}^t & & \\
 & & & & & & L^T & & I_{\tilde{L}} & & -J_{\tilde{L}}^t & \\
 & & & & & & U^T & & & I_{\tilde{U}} & & -J_{\tilde{U}}^t \\
B & L & U & & & & & & & & & \\
J_{\tilde{B}} & & & I_{\tilde{B}} & & & & & & & & \\
 & J_{\tilde{L}} & & & I_{\tilde{L}} & & & & & & & \\
 & & J_{\tilde{U}} & & & I_{\tilde{U}} & & & & & & \\
Z_B & & & & & & X_B & & & & & \\
 & Z_L & & & & & & X_L & & & & \\
 & & Z_U & & & & & & X_U & & & \\
-W_{\tilde{B}}J_{\tilde{B}} & & & & & & & & & S_{\tilde{B}} & & \\
 & -W_{\tilde{L}}J_{\tilde{L}} & & & & & & & & & S_{\tilde{L}} & \\
 & & -W_{\tilde{U}}J_{\tilde{U}} & & & & & & & & & S_{\tilde{U}}
\end{pmatrix}$$

e $d_1 = -\left(r_{d_B}, r_{d_L}, r_{d_U}, r_p, (r_u)_{\tilde{B}}, (r_u)_{\tilde{L}}, (r_u)_{\tilde{U}}, r_{a_B}, r_{a_L}, r_{a_U}, (r_c)_{\tilde{B}}, (r_c)_{\tilde{L}}, (r_c)_{\tilde{U}}\right)^t$.

Recordando que $A \in \mathbb{R}^{m\times n}$, vejamos as dimensões dos blocos da matriz A_1:

- B, X_B, $Z_B \in \mathbb{R}^{m\times m}$;
- $L \in \mathbb{R}^{m\times l}$;
- Z_L, $X_L \in \mathbb{R}^{l\times l}$;
- $I_{\tilde{B}}, S_{\tilde{B}}, W_{\tilde{B}} \in \mathbb{R}^{\tilde{b}\times\tilde{b}}$;
- $J_{\tilde{B}} \in \mathbb{R}^{\tilde{b}\times m}$;
- $I_{\tilde{L}}, S_{\tilde{L}}, W_{\tilde{L}} \in \mathbb{R}^{\tilde{l}\times\tilde{l}}$;
- $J_{\tilde{L}} \in \mathbb{R}^{\tilde{l}\times l}$;
- $U \in \mathbb{R}^{m\times\eta}$;
- $Z_U, X_U \in \mathbb{R}^{\eta\times\eta}$,

onde $m + l + \eta = n$, $\tilde{b} \leq b$ e $\tilde{l} \leq l$. Finalmente, nos resta verificar a dimensão dos blocos $\tilde{U}$.

Agora, a matriz $J_{\tilde{U}}$ não pode conter colunas nulas, pois dessa forma, no cálculo de $(r_u)_{\tilde{U}}$,

$$(r_u)_{\tilde{U}} = J_{\tilde{U}}\, x_U + s_{\tilde{U}} - u_{\tilde{U}},$$

teríamos uma coordenada de x_U, digamos $(x_U)_j$, próxima de seu limite superior, mas sem uma correspondente coordenada no vetor s e no vetor u. De forma análoga, a matriz $J_{\tilde{U}}$ não pode conter linhas nulas. Além disso, como a matriz J é obtida da matriz identidade e assumimos que o bloco $J_{\tilde{U}}$ está na forma escada, concluímos que a matriz $J_{\tilde{U}}$ tem que ser quadrada e todos os elementos de sua diagonal principal devem ser iguais a 1. Ou seja,

$$J_{\tilde{U}} = I_{\tilde{U}}.$$

Adicionalmente, podemos verificar que as matrizes U e $J_{\tilde{U}}$ possuem o mesmo número de colunas, logo concluímos que a matriz $J_{\tilde{U}}$ é a matriz identidade de ordem η. A mesma dimensão dos blocos Z_U e X_U. Dessa forma, $Z_U, X_U, S_{\tilde{U}}, W_{\tilde{U}}, I_{\tilde{U}} \in \mathbb{R}^{\eta\times\eta}$. Assim, podemos substituir os índices $\tilde{U}$ por, simplesmente, U.

Agora, o segundo passo para obtermos a abordagem do sistema estável para variáveis canalizadas, consiste em "mover" X_B, X_L e S_U. Para isso, multiplicamos o primeiro conjunto de equações por $-X_B$ e somamos ao sexto conjunto de equações, multiplicamos o segundo conjunto de equações por $-X_L$ e somamos ao sétimo conjunto de equações e, finalmente, multiplicamos o terceiro conjunto de equações por S_U e somamos ao último conjunto de equações. Assim, o sistema (5.11) é equivalente ao sistema formado pela matriz A_2 e pelo vetor d_2 dados por

$$\begin{pmatrix}
 & & & & B^T & I_{\tilde{B}} & & -J^t_{\tilde{B}} & & \\
 & & & & L^T & & I_{\tilde{L}} & & -J^t_{\tilde{L}} & \\
 & & & & U^T & & & I_U & & -I^t_U \\
B & L & U & & & & & & & \\
J_{\tilde{B}} & & & I_{\tilde{B}} & & & & & & \\
 & J_{\tilde{L}} & & & I_{\tilde{L}} & & & & & \\
 & & I_U & & & I_U & & & & \\
Z_B & & & & -X_B B^T & & & X_B J^t_{\tilde{B}} & & \\
 & Z_L & & & -X_L L^T & & & & X_L J^t_{\tilde{L}} & \\
 & & Z_U & & & & X_U & & & \\
-W_{\tilde{B}} J_{\tilde{B}} & & & & & & & S_{\tilde{B}} & & \\
 & -W_{\tilde{L}} J_{\tilde{L}} & & & & & & & S_{\tilde{L}} & \\
 & & -W_U I_U & & S_U U^T & & S_U & & &
\end{pmatrix}$$

e

$$-(r_{d_B}, r_{d_L}, r_{d_U}, r_p, (r_u)_{\tilde{B}}, (r_u)_{\tilde{L}}, r_{uU}, -X_B r_{d_B} + r_{a_B}, -X_L r_{d_L} + r_{a_L}, r_{a_U}, (r_c)_{\tilde{B}}, (r_c)_{\tilde{L}}, S_U r_{d_U} + r_{c_U})^t.$$

O passo seguinte consiste em eliminar Z_B e $-W_{\tilde{B}} J_{\tilde{B}}$ da primeira coluna. Primeiro, multiplicamos o quarto conjunto de equações por $-Z_B$ e somamos ao oitavo conjunto de equações. Em seguida, multiplicamos o quarto conjunto de equações por $W_{\tilde{B}} J_{\tilde{B}}$ e somamos ao décimo primeiro conjunto de equações.

Tendo em vista que os primeiros sete conjuntos de equações da matriz A_2 não sofrem mudanças nos próximos passos, iremos omiti-las quando nos referirmos às próximas matrizes semelhantes a A_2.

Dessa forma, obtemos a matriz A_3 e o vetor d_3, que juntos formam um sistema equivalente ao sistema (5.11), onde

$$A_3 = \begin{pmatrix} -Z_B B^{-1} L & -Z_B B^{-1} U & -X_B B^T & & X_B J_{\tilde{B}}^t & \\ Z_L & & -X_L L^T & & & X_L J_{\tilde{L}}^t \\ & Z_U & & X_U & & \\ W_{\tilde{B}} J_{\tilde{B}} B^{-1} L & W_{\tilde{B}} J_{\tilde{B}} B^{-1} U & & & S_{\tilde{B}} & \\ -W_{\tilde{L}} J_{\tilde{L}} & & & & & S_{\tilde{L}} \\ & -W_U I_U & S_U U^T & S_U & & \end{pmatrix}$$

e

$$d_3 = -\begin{pmatrix} -Z_B B^{-1} r_p - X_B r_{d_B} + r_{a_B} \\ -X_L r_{d_L} + r_{a_L} \\ r_{a_U} \\ W_{\tilde{B}} J_{\tilde{B}} B^{-1} r_p + (r_c)_{\tilde{B}} \\ (r_c)_{\tilde{L}} \\ S_U r_{d_U} + r_{c_U} \end{pmatrix}.$$

Finalmente, nosso último passo consiste em eliminar $X_B J_{\tilde{B}}^t$, $X_L J_{\tilde{L}}^t$ e S_U. Para isso, utilizamos, respectivamente, $S_{\tilde{B}}$, $S_{\tilde{L}}$ e X_U, todos invertíveis. Para tal, utilizamos apenas a matriz A_3 e o vetor d_3. Obtemos então, a matriz A_4 e o vetor d_4 semelhantes a A_3 e d_3, respectivamente,

onde

$$A_4 = \begin{pmatrix} K_1L & K_1U & -X_BB^T & & \\ Z_L + X_LJ_{\tilde{L}}^tS_{\tilde{L}}^{-1}W_{\tilde{L}}J_{\tilde{L}} & & -X_LL^T & & \\ & Z_U & & X_U & \\ W_{\tilde{B}}J_{\tilde{B}}B^{-1}L & W_{\tilde{B}}J_{\tilde{B}}B^{-1}U & & & S_{\tilde{B}} \\ -W_{\tilde{L}}J_{\tilde{L}} & & & & S_{\tilde{L}} \\ & -(W_U + S_UX_U^{-1}Z_U) & S_UU^T & & \end{pmatrix},$$

$$d_4 = \begin{pmatrix} r_1 \\ r_2 \\ -r_{a_U} \\ -W_{\tilde{B}}J_{\tilde{B}}B^{-1}r_p - (r_c)_{\tilde{B}} \\ -(r_c)_{\tilde{L}} \\ r_3 \end{pmatrix}$$

e

$$\begin{aligned} K_1 &= -(Z_B + X_BJ_{\tilde{B}}^tS_{\tilde{B}}^{-1}W_{\tilde{B}}J_{\tilde{B}})B^{-1}, \\ r_1 &= X_BJ_{\tilde{B}}^tS_{\tilde{B}}^{-1}\left(W_{\tilde{B}}J_{\tilde{B}}B^{-1}r_p + (r_c)_{\tilde{B}}\right) + Z_BB^{-1}r_p + X_Br_{d_B} - r_{a_B}, \\ r_2 &= X_LJ_{\tilde{L}}^tS_{\tilde{L}}^{-1}(r_c)_{\tilde{L}} + X_Lr_{d_L} - r_{a_L}, \\ r_3 &= S_UX_U^{-1}r_{a_U} - S_Ur_{d_U} - r_{c_U}. \end{aligned} \tag{5.12}$$

O sistema procurado está no primeiro, segundo e sexto conjunto de equações de A_4 e de d_4. Finalmente, a abordagem do sistema estável para variáveis canalizadas é escrito da forma,

$$\begin{pmatrix} -X_B & -(Z_B + X_BJ_{\tilde{B}}^tS_{\tilde{B}}^{-1}W_{\tilde{B}}J_{\tilde{B}})B^{-1}L & -(Z_B + X_BJ_{\tilde{B}}^tS_{\tilde{B}}^{-1}W_{\tilde{B}}J_{\tilde{B}})B^{-1}U \\ -X_LL^tB^{-t} & Z_L + X_LJ_{\tilde{L}}^tS_{\tilde{L}}^{-1}W_{\tilde{L}}J_{\tilde{L}} & \\ S_UU^tB^{-t} & & -(W_U + S_UX_U^{-1}Z_U) \end{pmatrix} \begin{pmatrix} \Delta\tilde{y} \\ \Delta x_L \\ \Delta x_U \end{pmatrix}$$

$$= \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}, \tag{5.13}$$

onde $\Delta\tilde{y} = B^t\Delta y$.

Podemos observar que, os três blocos na diagonal principal são invertíveis, enquanto os demais quatro blocos fora da diagonal principal convergem para zero, quando o sistema está próximo de uma solução.

Após encontrarmos a solução do sistema (5.13), podemos encontrar a solução completa do sistema (5.11), ou seja, os demais ($\Delta's$), através das seguintes equações:

$$\begin{cases} \Delta w_{\tilde{B}} = S_{\tilde{B}}^{-1}(W_{\tilde{B}} J_{\tilde{B}} B^{-1}[-r_p - L\Delta x_L - U\Delta x_U] - (r_c)_{\tilde{B}}) \\ \Delta w_{\tilde{L}} = S_{\tilde{L}}^{-1}(W_{\tilde{L}} J_{\tilde{L}} \Delta x_L - (r_c)_{\tilde{L}}) \\ \Delta z_B = -r_{d_B} - \Delta\tilde{y} + J_{\tilde{B}}^t \Delta w_{\tilde{B}} \\ \Delta z_L = -r_{d_L} - L^t \Delta y + J_{\tilde{L}}^t \Delta w_{\tilde{L}} \\ \Delta z_U = X_U^{-1}(-r_{a_U} - Z_U \Delta x_U) \\ \Delta w_U = -r_{d_u} - \Delta z_u - U^t \Delta y \\ \Delta x_B = B^{-1}(-r_p - L\Delta x_L - U\Delta x_U) \\ \Delta s_{\tilde{B}} = -(r_u)_{\tilde{B}} - J_{\tilde{B}} \Delta x_B \\ \Delta s_{\tilde{L}} = -(r_u)_{\tilde{L}} - J_{\tilde{L}} \Delta x_L \\ \Delta s_U = -r_{u_U} - \Delta x_U \end{cases} . \tag{5.14}$$

É fácil verificar que a abordagem do sistema (5.13) se aplicado à problemas sem variáveis canalizadas, se transforma na abordagem do sistema (4.5), ou seja, o sistema (4.5) é um caso particular do sistema (5.13). Dessa forma, podemos substituir o método (4.4.2) por um novo método. Método esse, capaz de resolver a abordagem do sistema linear estável para problemas com variáveis canalizadas.

Antes de apresentar o método para resolver o sistema (5.13), vamos mostrar que o sistema (5.13) é um sistema por blocos.

5.4 Método sistema estável para variáveis canalizadas

O método bloco iterativo para solução de um sistema por blocos (4.1), também pode ser utilizado para encontrar a solução do sistema (5.13). Podemos verificar que o sistema linear (5.13) possui a mesma estrutura do sistema (4.6), ou seja, também é um sistema por blocos. Para isso, assumimos que

$$\begin{aligned} M_1 &= -X_B \\ M_2 &= -\left(Z_B + X_B J_{\tilde{B}}^t S_{\tilde{B}}^{-1} W_{\tilde{B}} J_{\tilde{B}} \right) B^{-1} E \\ M_3 &= \begin{pmatrix} -X_L & \\ & S_U \end{pmatrix} E^t B^{-t} \end{aligned}$$

$$
\begin{aligned}
M_4 &= \begin{pmatrix} Z_L + X_L J_{\tilde{L}}^t S_{\tilde{L}}^{-1} W_{\tilde{L}} J_{\tilde{L}} & \\ & -(W_U + S_U X_U^{-1} Z_U) \end{pmatrix}, \\
p_1 &= r_1 \\
p_2 &= \begin{pmatrix} r_2 \\ r_3 \end{pmatrix}
\end{aligned}
$$

e $\Delta w_1 = \Delta \tilde{y}$ e $\Delta w_2 = (\Delta x_L \quad \Delta x_U)^t$, onde $E = (L \quad U)$.

No método bloco iterativo, método (4.1), o bloco Δw_1 é calculado em função do bloco Δw_2 e vice-versa. Claramente, as matrizes M_1 e M_4, definidas acima, são invertíveis, para os problemas não degenerados. Adicionalmente, podemos verificar que a matriz M_4 é uma matriz diagonal, concluímos então que Δx_L e Δx_U são calculados, apenas, em função de Δw_1, ou seja, Δx_L não é calculado em função de Δx_U e Δx_U não é calculado em função de Δx_L. Dessa forma, as propriedades e os teoremas da seção 4.3 podem ser aplicados ao sistema linear (5.13). Portanto, podemos utilizar o método bloco iterativo para resolvê-lo.

Os teoremas (4.1) e (4.2) utilizam as matrizes M e N. Para o sistema linear (5.13), essas matrizes são,

$$
\begin{aligned}
M &= X_B^{-1} \left(\; Z_B + X_B J_{\tilde{B}}^t S_{\tilde{B}}^{-1} W_{\tilde{B}} J_{\tilde{B}} \; \right) B^{-1} E, \\
N &= \begin{pmatrix} Z_L + X_L J_{\tilde{L}}^t S_{\tilde{L}}^{-1} W_{\tilde{L}} J_{\tilde{L}} & \\ & -(W_U + S_U X_U^{-1} Z_U) \end{pmatrix}^{-1} \begin{pmatrix} -X_L & \\ & S_U \end{pmatrix} E^t B^{-t}.
\end{aligned}
$$

Portanto, de acordo com esses dois teoremas, o método bloco iterativo irá convergir para a solução do sistema linear (5.13), se $||MN|| < 1$ ou $||NM|| < 1$, teorema (4.1), ou se $\rho(MN) < 1$, ou $\rho(NM) < 1$, onde ρ é o raio espectral (isto é, o maior autovalor em módulo), teorema (4.1).

Método 5.1 (Sistema estável para variáveis canalizadas).

1. Calcule a matriz de permutação P dos índices $\{1, ..., n\}$ correspondentes à ordem decrescente da norma das colunas da matriz AD^{-1}.

2. Encontre as primeiras m colunas linearmente independentes de A, seguindo o ordenamento calculado. Chamamos essas m colunas de matriz B.

3. Identifique quantas e quais coordenadas de X estão próximas ao limite inferior, l, e ao limite superior, u. Ao conjunto das colunas de A correspondentes a essas coordenadas damos o nome de matriz L e U, respectivamente. Portanto, x_B e z_B são as componentes de x e z

correspondentes às colunas de B; x_L e z_L são as componentes de x e z correspondentes às colunas de L e, x_U e z_U são as componentes de x e z correspondentes às colunas de U;

4. Identifique quantas e quais coordenadas de S correspondem às matrizes B, L e U. Chamamos de $s_{\tilde{B}}$, $s_{\tilde{L}}$ e s_U respectivamente. Isto é, identificar as colunas de A que são canalizadas.

5. Calcule r_1, r_2 e r_3 definidos em (5.12).

6. Seja $\Delta\tilde{y}^0 = 0$ e, para $j = 0, 1, 2, ...$, faça

   (a) $\Delta x_L^{j+1} = (Z_L + X_L J_{\tilde{L}}^t S_{\tilde{L}}^{-1} W_{\tilde{L}} J_{\tilde{L}})^{-1}[r_2 + X_L L^T B^{-t} \Delta\tilde{y}^j]$

   (b) $\Delta x_U^{j+1} = (W_U + S_U X_U^{-1} Z_U)^{-1}[S_U U^T B^{-t} \Delta\tilde{y}^j - r_3]$

   (c) $\Delta\tilde{y}^{j+1} = -X_B^{-1}[r_1 + (Z_B + X_B J_{\tilde{B}}^t S_{\tilde{B}}^{-1} W_{\tilde{B}} J_{\tilde{B}}) B^{-1}(L\Delta x_L^{j+1} + U\Delta x_U^{j+1})]$

7. No último passo, encontre as demais direções como em (5.14).

No passo **3** do método acima, uma coordenada de x, digamos x_i, está próxima de seu limite superior, u_j, se $x_i > s_j$, onde s_j é a coordenada do vetor s que corresponde à coordenada x_i. Caso contrário, x_i está próxima de seu limite inferior. Se a coordenada x_i não possui uma coordenada correspondente no vetor s, então classificamos x_i como próxima de seu limite inferior.

Também utilizamos os passos **6** e **7** do método 4.4.2, no lugar do passo **6** do método acima, em nossos experimentos. Em seguida, substituímos as normas calculadas no passo **6**, pelas normas

$$\left\| M_2 M_4^{-1} \begin{pmatrix} r_2 \\ r_3 \end{pmatrix} - r_1 \right\| \qquad \text{e} \qquad \left\| M_3 M_1^{-1} r_1 - \begin{pmatrix} r_2 \\ r_3 \end{pmatrix} \right\|$$

e mantivemos o passo 7. Porém, o método 5.1 foi o que apresentou melhor desempenho.

Durante nossos estudos, desenvolvemos um outro método para resolução da abordagem do sistema linear estável. Como esse método calcula o valor máximo entre as coordenadas de X_B e as coordenadas de $S_{\tilde{B}}$, demos o nome de método sistema linear estável - versão MAX. Esse método não se aplica ao sistema em (5.13), mas pode ser aplicado a um sistema linear semelhante a esse. Apresentamos no Apêndice A o desenvolvimento do método MAX.

5.5 Problemas degenerados

Já analisamos, no capítulo anterior, o comportamento do sistema estável próximo de uma solução. Vimos na seção 4.4.2, que a matriz do sistema estável quando próxima de uma solução, de um

problema degenerado, se torna próxima de ser singular, o que pode causar uma falha no método do sistema estável.

Podemos contornar essa situação, utilizando uma perturbação na matriz do sistema estável. Na seção 4.4.3, vimos o cálculo das perturbações α_1, α_2, α_3 e α_4, utilizadas para problemas sem canalizações. Podemos utilizar perturbações semelhantes para os problemas com canalizações. Para isso, precisamos fazer algumas alterações no cálculo das perturbações.

Em (4.11), calculamos α_1 como sendo o menor elemento da diagonal principal da matriz do sistema estável, sem variáveis canalizadas, dividido pela norma do vetor a direita. Utilizando esse mesmo raciocínio para variáveis com canalização, devemos, então, considerar não mais o sistema sem canalização (4.7), mas sim o sistema com canalização (5.13). Dessa forma, nossa primeira tentativa para o cálculo de δ_1 foi,

$$min(-X_B, (Z_L + X_L J_{\tilde{L}}^t S_{\tilde{L}}^{-1} W_{\tilde{L}} J_{\tilde{L}}), -(W_U + S_U X_U^{-1} Z_U)) \Bigg/ \left\| \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} \right\|. \tag{5.15}$$

Porém, na prática, os resultados numéricos se mostraram melhores quando δ_1 é calculado como sendo,

$$\delta_1 = min(X_B, Z_L, W_U) \Bigg/ \left\| \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} \right\|. \tag{5.16}$$

Assim como α_2 e α_3, as perturbações δ_2 e δ_3 também surgem da desigualdade (4.10). Escrevendo essa desigualdade para a abordagem do sistema estável com variáveis canalizadas, obtemos

$$\begin{aligned} &\rho\left(V_1 \begin{pmatrix} -X_L & \\ & S_U \end{pmatrix} V^t (X_B + \delta_2 I)^{-1} (Z_B + X_B J_{\tilde{B}}^t S_{\tilde{B}}^{-1} W_{\tilde{B}} J_{\tilde{B}}) V\right) \;<\; 1, \text{ ou} \\ &\rho\left((X_B + \delta_2 I)^{-1} (Z_B + X_B J_{\tilde{B}}^t S_{\tilde{B}}^{-1} W_{\tilde{B}} J_{\tilde{B}}) V V_1 \begin{pmatrix} -X_L & \\ & S_U \end{pmatrix} V^t\right) \end{aligned} \tag{5.17}$$

onde

$$V_1 = \begin{pmatrix} (Z_L + X_L J_{\tilde{L}}^t S_{\tilde{L}}^{-1} W_{\tilde{L}} J_{\tilde{L}}) + \delta_2 I & \\ & -(W_U + S_U X_U^{-1} Z_U) + \delta_2 I \end{pmatrix}^{-1}.$$

De forma análoga ao calculo de α_2, tomamos $V = [I \;\; 0]P$, e assumimos que x_B, z_L e w_U

possuem todas as coordenadas nulas. Obtemos δ_2 como sendo o menor β satisfazendo

$$\beta^2 > máx\left(\begin{pmatrix} -X_L & \\ & S_U \end{pmatrix}(Z_B + X_B J_{\tilde{B}}^t S_{\tilde{B}}^{-1} W_{\tilde{B}} J_{\tilde{B}})\right).$$

Portanto, δ_2 é calculado da seguinte forma:

$$\delta_2 = max\left(\begin{pmatrix} -X_L & \\ & S_U \end{pmatrix}_{1:\mathrm{m}} (Z_B + X_B J_{\tilde{B}}^t S_{\tilde{B}}^{-1} W_{\tilde{B}} J_{\tilde{B}})\right)^{\frac{1}{2}},$$

se $n - m \geq m$, ou

$$\delta_2 = max\left(\begin{pmatrix} -X_L & \\ & S_U \end{pmatrix}(Z_B + X_B J_{\tilde{B}}^t S_{\tilde{B}}^{-1} W_{\tilde{B}} J_{\tilde{B}})_{1:(\mathrm{n-m})}\right)^{\frac{1}{2}},$$

se $n - m < m$. Novamente, m é a dimensão da matriz B.

Para a perturbação δ_3, tomamos $V = E$ em (5.17). Ainda assumindo que x_B, z_L e w_U possuem todas as coordenadas nulas. Obtemos então,

$$\rho\left(\begin{pmatrix} -X_L & \\ & S_U \end{pmatrix} V_1 E^t \left((Z_B + X_B J_{\tilde{B}}^t S_{\tilde{B}}^{-1} W_{\tilde{B}} J_{\tilde{B}})(X_B + \delta_2)^{-1} \right) E\right) \quad < \quad 1. \quad (5.18)$$

Novamente, através dos Círculos de Gershgorin, estimamos o maior autovalor da matriz acima como sendo o maior elemento de sua diagonal principal, ϵ. Dessa forma, obtemos a seguinte perturbação,

$$\delta_3 = \epsilon \times \left|\left|\begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}\right|\right|$$

se $\epsilon > 1$. Caso contrário

$$\delta_3 = \epsilon \Big/ \left|\left|\begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}\right|\right|.$$

Podemos verificar que, as perturbações α's, são casos particulares das perturbações δ's. Ou seja, se utilizarmos essas três abordagens de perturbação em problemas sem canalização, então obteremos,

$$\delta_1 = \alpha_1, \qquad \delta_2 = \alpha_2 \qquad \text{e} \qquad \delta_3 = \alpha_3.$$

Assim como na seção 4.4.3, também utilizamos uma quarta perturbação, δ_4. Essa perturbação é calculada de forma idêntica àquela apresentada na seção 4.4.3.

Caso a menor coordenada de x_B, z_L ou w_U, seja menor que 10^{-8}, então perturbamos esse vetor com a perturbação δ_4. Se for necessário perturbar dois desses vetores, ou os três vetores, utilizamos a mesma perturbação para esses vetores.

Os resultados apresentados no capítulo 6 foram obtidos utilizando-se δ_4 como perturbação para todos os problemas degenerados, ou seja, que satisfazem a condição acima.

Capítulo 6

Resultados Numéricos

Nesse capítulo apresentamos os resultados numéricos da implementação do método do sistema estável no cálculo da direção de busca de um conjunto de problemas lineares. Além disso, também apresentamos uma comparação entre as abordagens sistema estável e sistema de equações normais, resolvido por um método direto, decomposição de Cholesky, e resolvido também por um método iterativo, pré-condicionador híbrido.

Realizamos experimentos numéricos com 4 implementações diferentes, na primeira utilizamos simplesmente o código PCx, que é uma implementação de uma variação do método preditor corretor de Mehrotra, como visto na seção 5.2, e em detalhes em [46]. Veremos um pouco mais sobre esse código na seção 6.1. Na segunda implementação, utilizamos o método sistema linear estável para variáveis canalizadas, como visto no capítulo 5, para ser ativado nas últimas iterações do código PCx.

Para a terceira implementação, utilizamos o pré-condicionador híbrido, como visto no capítulo 3, Finalmente, em nossa quarta implementação utilizamos o sistema linear estável para variáveis canalizadas nas últimas iterações da abordagem com o pré-condicionador híbrido.

As rotinas referentes ao sistema estável foram implementadas em linguagem C. Assim como as rotinas para resolução dos sistemas lineares do pré-condicionador híbrido, exceto as rotinas referentes à decomposição controlada de Cholesky, que foram implementadas em Fortran.

Os problemas utilizados nos experimentos numéricos pertencem à coleção NETLIB [48] de problemas de programação linear. A NETLIB é uma coleção de problemas da vida real provenientes de várias situações. Por exemplo, enquanto os problemas STOCFOR2 e STOCFOR3 são provenientes da ocorrência de queimadas em determinadas áreas de uma floresta, o problema BLEND é obtido de uma refinaria de petróleo.

6.1 O código PCx

O PCx é um software matemático desenvolvido em conjunto pelo Optimization Technology Center do Argonne National Laboratory e pela Universidade Northwestern. Ele consiste na implementação de uma variação do método preditor-corretor de Mehrotra [46], com a correção de Gondzio [26], em linguagem *C*. Na solução dos sistemas de equações lineares é utilizado o pacote Cholesky esparso de Ng e Peyton [36] desenvolvido em Fortran.

Veremos nessa seção como o PCx faz a leitura de um problema e calcula a sua solução. O PCx pode ser encontrado na seguinte página da internet: *http://www.mcs.anl.gov/otc/Tools/PCx*.

6.1.1 A leitura do problema

O PCx lê os dados do problema linear no formato MPS. O formato MPS pode descrever variáveis com limites superior, ou inferior, restrições de igualdade, ou desigualdades e variáveis livres. O PCx define uma estrutura de dados que contém uma especificação completa de um único problema de programação linear. Essa estrutura de dados, também contém o nome das linhas, colunas e valores da função objetivo.

Em seguida, o PCx converte essa estrutura de dados para um formato mais fácil e rápido de trabalhar. Essa formulação é a mesma apresentada em (5.2). O PCx também utiliza o problema dual associado ao problema original. Aqui, o problema dual possui a mesma fórmula que a apresentada em (5.3).

Assim como nos demais algoritmos primal-dual-infactíveis, o método de Mehrotra, implementado no PCx, gera uma sequência de termos que satisfazem estritamente à condição de não negatividade.

6.1.2 Álgebra Linear

A versão padrão do PCx utiliza o sistema de equações normais (2.21), para resolver os sistemas lineares. Um algoritmo da decomposição esparsa de Cholesky é usado para decompor a matriz $S^k = A(D^k)^{-1}A^t$, e a solução Δy^k é obtida através de substituições triangulares com a matriz L, da decomposição de Cholesky. Esse algoritmo é a decomposição esparsa de Cholesky de NG e Peyton [49], com uma pequena alteração para pivôs próximos de zero, que freqüentemente aparecem nas iterações dos métodos de pontos interiores. Essa decomposição é realizada da

seguinte forma,

$$P(A(D^k)^{-1}A^t)P^t = LL^t,$$

onde P é uma matriz de permutação, obtida independentemente dos elementos de $A(D^k)^{-1}A^t$, e L é uma matriz triangular superior.

A implementação de Ng e Peyton usa uma estratégia de *multiple minimum degree ordering*, que é idêntica à encontrada no SPARSPAK. Essa estratégia foi apresentada por Liu [35]. A abordagem utilizada na decomposição foi parcialmente descrita por Liu [36] e Gilbert, Ng e Peyton [29]. A decomposição numérica é realizada com um *left-looking* algoritmo esparso de Cholesky, idêntico ao descrito por Ng e Peyton em [49]. Essa abordagem utiliza memória hierárquica, através da divisão de supernós em blocos do tamanho da memória cache disponível.

Sabendo que a estrutura de elementos não nulos na matriz a ser decomposta é a mesma em cada iteração, a ordenação e a criação dos vetores que armazenam essa decomposição são realizados apenas uma vez, durante o cálculo do ponto inicial. Em cada iteração, a decomposição numérica é realizada uma única vez. Em seguida, são resolvidos dois sistemas lineares, um no passo preditor e outro no passo corretor, ambos resolvidos pelo processo de substituição de sistema triangular.

A modificação do algoritmo de Ng e Peyton consiste em substituir pivôs muito pequenos por 10^{128}. Essa substituição faz com que os elementos fora da diagonal principal na $i-$ésima coluna de L sejam muito próximos de zero, o que por sua vez, faz com que a $i-$ésima coordenada do vetor solução seja, também, muito próxima de zero. Wright em [63] realizou estudos sobre essa abordagem. Um pivô é considerado pequeno se

$$M_{ii}^{(i-1)} \leq 10^{-30} max_{j=1,2,...,m} M_{jj}^2,$$

onde $M^{(i-1)}$ é a sub-matriz restante de A após $i-1$ passos da decomposição de Cholesky e M é a matriz semi-definida positiva simétrica original.

Se a matriz A possui colunas densas, ou seja, com muitos elementos não-nulos, então o produto $A(D^k)^{-1}A^t$ pode ocasionar em uma matriz ainda mais densa que a matriz original, o que tornaria essa estratégia muito ineficiente. Para evitar colunas densas em $A(D^k)^{-1}A^t$, a matriz A é dividida em A_{es} e A_{den}, onde A_{es} são as colunas esparsas de A e A_{den} são as colunas densas de A. Podemos encontrar mais sobre essa decomposição da matriz A em A_{den} e A_{es} em [4] e [58]. A matriz D também pode ser dividida de forma análoga e, portanto, podemos escrever

$$A(D^k)^{-1}A^t = A_{es}(D^k)_{es}^{-1}A_{es}^t + A_{den}(D^k)_{den}^{-1}A_{den}^t = M + A_{den}(D^k)_{den}^{-1}A_{den}^t. \tag{6.1}$$

Aplicando a fórmula de Sherman-Morrison-Woodbury para (6.1), temos

$$[M + A_{den}(D^k)^{-1}_{den}A^t_{den}]^{-1} = M^{-1} - (M^{-1}A_{den})[(D^k)_{den} + A^t_{den}M^{-1}A_{den}]^{-1}A^t_{den}M^{-1}$$

Dessa forma, a decomposição esparsa de Cholesky é aplicada apenas na matriz M, onde obtemos

$$PMP^t = LL^t.$$

A solução de um sistema linear com a matriz $A(D^k)^{-1}A^t$ e lado direito um vetor r é,

$$\begin{aligned}(A(D^k)^{-1}A^t)^{-1}r = \\ P^tL^{-t}\left\{I - L^{-1}PA_{den}[(D^k)_{den} + A^t_{den}P^tL^{-t}L^{-1}PA_{den}]^{-1}A^T_{den}P^tL^{-t}\right\}L^{-1}Pr.\end{aligned} \tag{6.2}$$

Dados L e P, o maior custo para se aplicar a fórmula acima é o produto $(L^{-1}PA_{den})$, um custo total de $n_{den} + 2$ substituições triangulares, onde n_{den} é o número de colunas de A_{den}. Para os demais sistemas com a mesma matriz de coeficiente, mas um vetor diferente do lado direito, o custo para se aplicar a fórmula (6.2) é apenas duas substituições triangulares.

Para se determinar as colunas que devem ser classificadas como densas, é criado um vetor com o número de elementos não-nulos em cada coluna, em ordem decrescente. Em seguida, escolhe-se as colunas tais que a proporção de elementos não-nulos é maior ou igual a τ, onde

$$\begin{aligned}\tau = &\quad 1 \quad , \quad \text{se} \quad m \leq 500, \\ \tau = &\quad 0,1 \quad , \quad \text{se} \quad 500 < m \leq 2000, \\ \tau = &\quad 0,05 \quad , \quad \text{se} \quad m > 2000,\end{aligned}$$

e calcula-se um *gap* na sequência de número de elementos não-nulos. As colunas que possuem número de elementos não-nulo acima do *gap* são classificadas como densas.

O código PCx termina com um dos quatro status: **ótimo**, **infactível**, **desconhecido**, ou **sub-ótimo**. O status de ótimo é obtido da mesma forma que em (5.10)

Para os demais status é utilizada uma função de mérito ϕ, definida por

$$\phi(x,s,z,w) = \frac{||(r_p, r_a)||}{\max(1, ||(b,u)||)} + \frac{||(r_d)||}{\max 1, ||c||} + \frac{|c^tx - (b^ty - \sum_{i\in NB} u_is_i)|}{\max(1, ||(b,u)||, ||c||)}.$$

Podemos observar que, pontos (x,s,y,z,w), onde $(x,s,z,w) \geq 0$ e $\phi = 0$ são soluções primal-duais para (5.2) e (5.3), e vice-versa. Para problemas factíveis, isto é, que possuem soluções, após as primeiras iterações, a função ϕ converge para zero, enquanto que, para problemas infactíveis, que não possuem solução, a função ϕ aumenta de valor rapidamente.

Para os demais testes de parada, o método utiliza o vetor ϕ_{min}. Esse vetor possui como k-ésimo elemento o menor valor de ϕ encontrado até a $k-$ésima iteração, ou seja,

$$\phi_{min}[k] = min_{l=0,1,\ldots,k}\phi(x_l, s_l, y_l, z_l, w_l).$$

O código pára na iteração k com status infactível se os teste de solução ótima não são satisfeitos e o seguinte teste é satisfeito,

$$\phi(x_k, s_k, y_k, z_k, w_k) \geq \max(10^{-8}, 10^5\phi_{min}[k]).$$

Agora, caso seja detectada uma convergência muito lenta, ou seja,

$$\phi_{min}[k-30] \geq \frac{1}{2}\phi_{min}[k], \qquad \text{e} \qquad k \geq 30,$$

o método termina com o *status* de desconhecido.

Finalmente, o método termina com o status de sub-ótimo caso o limite de iterações seja excedido sem que nenhuma das outras condições sejam satisfeitas.

6.1.3 Pré-processamento

Freqüentemente, problemas de programação linear contém informações redundantes, bem como informações e estruturas que permitam que algumas componentes da solução sejam determinadas sem que se recorra a um método sofisticado. As rotinas de pré-processamento têm a finalidade de detectar essas informações redundantes do problema original e, produzir um problema menor e mais fácil de ser resolvido do que o original, mas que tenha a mesma solução ótima. Essas rotinas melhoram significativamente a eficiência e estabilidade numérica das implementações dos métodos de pontos-interiores.

O pré-processador utiliza as técnicas descritas por Andersen e Andersen em [3] verificando se os dados do problema possuem as seguintes características:

- **Infactível.** Verifica se as coordenadas do vetor u são maiores que zero, ou seja, se todos os limites superiores das coordenadas limitadas são maiores que zero. Além disso, verifica se uma linha nula de A corresponde a uma coordenada nula do vetor b.

- **Linhas Nulas.** Se a matriz A possui uma linha nula e a coordenada correspondente no vetor b é nula, então podemos remover essa linha de A e essa coordenada de b.

- **Linhas Duplicadas.** Quando uma linha de A, e a coordenada correspondente em b, é múltipla de outra linha, podemos eliminá-la sem afetar a solução.

- **Colunas Duplicadas.** Quando uma coluna de A é múltipla de outra coluna, e se essas duas colunas se referem a duas coordenadas de x que não sejam limitadas, então essas duas colunas podem ser combinadas.

- **Colunas nulas.** A coordenada correspondente x_i pode ser fixada como sendo seu limite superior, ou inferior, isto depende do sinal da variável de custo c_i. Se o limite superior não existe, então o problema primal é ilimitado.

- **Variáveis fixas.** Se uma variável possui o número zero como limite superior e inferior, então podemos considerar essa variável como zero e eliminá-la do problema.

- **Linhas com um único elemento.** Se a $i-$ésima linha de A possui um único elemento não-nulo, A_{ij}, então $x_j = \frac{b_i}{A_{ij}}$ e, podemos eliminar essa variável do problema. Portanto, podemos eliminar a $i-$ésima linha de A, e sua coordenada correspondente de y.

- **Colunas com um único elemento.** Quando A_{ij} é o único elemento não nulo na coluna j de A, e x_j é uma variável livre, podemos expressá-la em termos de outras variáveis representadas na linha i de A e, consequentemente, eliminá-la do problema. Caso x_j não seja livre, podemos eliminá-la se seus limites são menores que os das outras variáveis representadas na $i-$ésima linha de A.

- **Linhas forçadas.** Em alguns casos, a restrição linear representada pela $i-$ésima linha de A força suas variáveis a assumirem o valor de seus limites superior, ou inferior. Por exemplo, a restrição linear,

  $$10x_3 - 4x_{10} + x_{12} = -4$$

  sujeito a

  $$x_3 \in [0, +\infty), \quad x_{10} \in [0, 1], \quad x_{12} \in [0, +\infty)$$

  claramente implica em $x_3 = 0$, $x_{10} = 1$ e $x_{12} = 0$. Assim, esses três valores, e a correspondente linha de A, podem ser eliminados.

O pré-processador procura essas características nos dados do problema. Quando encontra uma delas, o pré-processador começa novamente a procura, pois ao eliminar uma dessas redundâncias do problema, geralmente, surgem outras. O processo termina quando é realizada uma procura sem detectar nenhuma dessas características no problema. Cada eliminação que o

pré-processador realiza é armazenada e depois utilizada para transformar a solução do problema reduzido em solução do problema original.

Embora essa parte do código aparenta ser complexa, o pré-processamento requer pouco tempo computacional em comparação com uma única iteração dos métodos de pontos interiores.

6.2 Experimentos Numéricos

Os resultados numéricos das implementações que utilizam o sistema estável foram implementadas seguindo um mesmo raciocínio. Nas primeiras iterações utilizamos a abordagem original, resolvendo o sistema de equações normais, via decomposição de Cholesky, ou via método dos Gradientes Conjugados pré-condicionado, método híbrido apresentado no capítulo 3. Quando μ atinge um valor menor que 10^{-10}, então a abordagem sistema linear estável é ativado e, consequentemente, utilizado para resolver os sistemas lineares das demais iterações. A abordagem sistema linear estável foi implementado seguindo o método 5.1.

A ativação do sistema estável quando $\mu < 10^{-10}$, foi motivado segundo nossos experimentos numéricos. Também, realizamos testes, para a ativação do sistema estável, quando μ é menor que 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8} e 10^{-9}. No entanto, o melhor desempenho, com base no número de problemas resolvidos, é quando consideramos $\mu < 10^{-10}$.

O principal objetivo da implementação do sistema estável em conjunto com métodos diretos e iterativos consiste em se obter uma maior precisão na solução de problemas de programação linear. Assim, reduzimos o erro no teste de parada e realizamos uma comparação entre os tempos obtidos com cada precisão.

Utilizamos nos experimentos, um computador com processador Intel Pentium Dual Core 64 bits, com 2.50 GHz e com 4 Gb de memória RAM. Compilamos os arquivos escritos em linguagem C no compilador GCC versão 4.3.3. Nesse computador encontrava-se instalado o sistema operacional Linux Ubuntu versão 9.04, com a versão *Linux version 2.6.28-18-generic* do kernell.

Os problemas utilizados nos experimentos numéricos pertencem à coleção NETLIB de problemas de programação linear. Foram utilizados um total de 90 problemas lineares, todos eles com variados tipos de restrições, do tipo não-negativas, canalizadas e fixas. A Tabela 6.1 mostra o número de linhas, o número de colunas, o número de elementos não nulos na matriz de restrições e a quantidade de variáveis canalizadas de cada problema. O símbolo (-) indica que o

problema não possui variáveis canalizadas. A última coluna indica se o problema necessitou ser perturbado, ou não. O símbolo (-) indica que o problema não foi perturbado. Podemos encontrar mais sobre esses problemas em [6].

Problemas	Linhas	Colunas	Elem. Não Nulos	Var. Canalizadas	Perturbação
25FV47	788	1843	10538	-	sim
80BAU3B	2140	11066	21631	2968	sim
ADLITTLE	55	137	417	-	-
AFIRO	27	51	102	-	sim
AGG	390	477	2055	-	sim
AGG2	514	750	4558	-	sim
AGG3	514	750	4574	-	-
BANDM	240	395	1894	-	sim
BEACONFD	86	171	1316	-	sim
BLEND	71	111	477	-	sim
BNL1	610	1491	5256	-	sim
BNL2	1964	4008	14037	-	sim
BOEING1	331	697	3104	243	-
BOEING2	126	265	979	73	sim
BORE3D	81	138	549	7	-
BRANDY	133	238	1911	-	sim
CAPRI	241	436	1528	131	sim
CYCLE	1420	2773	15004	77	sim
CZPROB	671	2779	5531	-	-
D2Q06C	2132	5728	31965	-	sim
DEGEN2	442	757	4167	-	sim
DEGEN3	1501	2604	25425	-	sim
E226	198	429	2515	-	sim
ETAMACRO	334	669	1995	131	sim
FFFFF800	233	826	5164	-	sim
FINNIS	438	935	2332	33	sim
FIT1D	24	1049	13427	1026	-

FIT1P	627	1677	9868	399	-
FIT2D	25	10524	129042	10500	sim
FIT2P	3000	13525	50284	7500	sim
FORPLAN	121	447	4415	22	sim
GANGES	1113	1510	6537	397	sim
GFRDPNC	590	1134	2393	258	sim
GREENBEA	1933	4164	23765	264	-
GREENBEB	1932	4254	23672	268	-
GROW15	300	645	5620	600	sim
GROW22	440	946	8252	880	sim
GROW7	140	301	2612	280	-
ISRAEL	174	316	2443	-	-
KB2	43	68	313	9	-
LOTFI	133	346	867	-	sim
MAROS	655	1437	6634	-	sim
NESM	654	2922	13244	1596	sim
PEROLD	593	1374	5636	259	sim
PILOT	1368	4543	41879	1040	sim
PILOT4	396	1022	5001	247	sim
PILOT87	1971	6373	72163	1578	-
PILOTJA	810	1804	11417	331	sim
PILOTNOV	848	2117	11922	332	sim
PILOTWE	701	2814	8841	287	sim
RECIPE	64	123	443	56	-
SC105	104	162	339	-	sim
SC205	203	315	663	-	sim
SC50A	49	77	159	-	sim
SC50B	48	76	146	-	-
SCAGR25	469	669	1715	-	sim
SCAGR7	127	183	455	-	-
SCFXM1	305	568	2457	-	sim

SCFXM2	610	1136	4919	-	-
SCFXM3	915	1704	7381	-	sim
SCORPION	317	412	1247	-	sim
SCRS8	421	1199	3036	-	sim
SCSD1	77	760	2388	-	-
SCSD6	147	1350	4316	-	sim
SCSD8	397	2750	8584	-	-
SCTAP1	284	644	1802	-	sim
SCTAP2	1033	2443	7052	-	-
SCTAP3	1408	3268	9383	-	-
SEBA	448	901	4102	447	-
SHARE1B	112	248	1148	-	-
SHARE2B	96	162	777	-	sim
SHELL	486	1451	2904	117	sim
SHIP04L	292	1905	4290	-	sim
SHIP04S	216	1281	2875	-	sim
SHIP08L	470	3121	7122	-	sim
SHIP08S	276	1604	3644	-	sim
SHIP12L	610	4171	9254	-	sim
SHIP12S	340	1943	4297	-	sim
SIERRA	1202	2705	7771	2016	-
STAIR	356	532	3813	6	sim
STANDATA	314	796	1403	96	-
STANDMPS	422	1192	2831	96	sim
STOCFOR1	102	150	421	-	-
STOCFOR2	1980	2868	8090	-	-
STOCFOR3	15362	22228	62960	-	sim
TRUSS	1000	8806	27836	-	sim
TUFF	257	567	4095	25	sim
VTPBASE	72	111	283	32	-
WOOD1P	171	1718	44575	-	sim

WOODW	708	5364	19809	-	sim

Tabela 6.1: Dimensão dos problemas.

6.3 Tabelas Comparativas

Vimos que nosso principal objetivo é obter uma melhor precisão na resolução dos problemas, por isso, realizamos experimentos em que o teste de parada foi reduzido para 10^{-10}, 10^{-12}, 10^{-14} e 10^{-16}. Identificamos esses valores do teste de parada como sendo a tolerância em que o problema foi resolvido. Também, identificamos precisão de uma problema como sendo a menor tolerância em que esse problema foi resolvido.

Apresentamos na Tabela 6.2 a quantidade de problemas resolvidos com as abordagens, decomposição de Cholesky, na resolução do sistema de equações normais, (PCx) e sistema estável para as últimas iterações do PCx (PCx-SE).

ABORDAGEM	TOLERÂNCIA				
	10^{-8}	10^{-10}	10^{-12}	10^{-14}	10^{-16}
PCx	86	84	73	35	1
PCx-SE	87 (1)	86 (8)	84 (31)	82 (62)	33 (33)

Tabela 6.2: Problemas resolvidos com as abordagens PCx e PCx-SE.

O número entre parênteses, na última linha da tabela, indica o número de problemas resolvidos em que a abordagem do sistema estável foi ativada. Por exemplo, na abordagem PCx-SE com tolerância de 10^{-16}, todos os 33 problemas resolvidos, utilizaram a abordagem do sistema estável nas últimas iterações.

Analisando os números dessa tabela, verificamos um aumento no número de problemas resolvidos em todas as tolerâncias utilizadas. Considerando 10^{-8} como tolerância e utilizando a abordagem PCx-SE, obtemos a resolução para o problema BRANDY, que o PCx não resolve com essa tolerância. Além disso, o PCx-SE resolve esse problema com uma precisão de 10^{-14}.

Agora, quando consideramos a tolerância de 10^{-10}, a abordagem PCx-SE resolve todos os 84 problemas que a abordagem PCx resolve com essa tolerância, exceto o problema CYCLE. Além desses 83 problemas, também obtemos a resolução dos problemas BRANDY, FORPLAN e SCFXM1. Esses três problemas são resolvido pelo PCx com uma precisão de, apenas, 10^{-8}.

O PCx resolve 73 problemas com tolerância de 10^{-12}, porém, quando utilizamos o sistema estável nas últimas iterações do código, esse número aumenta para 84 problemas. Destacamos aqui os problemas FINNIS e KB2. Esses problemas são resolvido pelo PCx com uma precisão de, apenas, 10^{-10}, porém através do PCx-SE foram resolvidos com uma precisão de 10^{-16}.

Quando analisamos a tolerância 10^{-14}, entre as duas abordagens, verificamos a maior diferença entre o número de problemas resolvidos. A abordagem PCx resolve apenas 35 problemas enquanto a abordagem PCx-SE resolve 82 problemas. Ou seja, considerando essa tolerância e, utilizando o método sistema linear estável nas últimas iterações, obtemos mais que o dobro de problemas resolvidos com a abordagem PCx.

Considerando a tolerância 10^{-16}, observamos um aumento de 1 para 33 problemas resolvidos. O problema resolvido pelo PCx com essa tolerância é o problema SCTAP3. Esse problema também é resolvido com essa mesma tolerância quando utilizamos a abordagem PCx-SE. Enquanto o PCx necessita de 21 iterações e $0,11$ segundos para obter a solução, com essa precisão, a abordagem PCx-SE necessita de, apenas, 17 iterações e $0,12$ segundos. Uma redução significativa no número de iterações, que, porém, ocasionou no mesmo tempo computacional, entre as duas abordagens.

Ainda considerando 10^{-16} como tolerância, podemos fazer uma comparação entre as duas abordagens com relação à porcentagem dos 90 problemas da coleção NETLIB, resolvidos com essa tolerância. Enquanto a abordagem PCx resolve apenas $1,11\%$ dos 90 problemas, a abordagem PCx-SE resolve $36,67\%$ dos problemas dessa coleção.

ABORDAGEM	TOLERÂNCIA				
	10^{-8}	10^{-10}	10^{-12}	10^{-14}	10^{-16}
PCH	67	48	18	10	0
PCH-SE	69 (4)	69 (27)	59 (47)	54 (47)	30 (30)

Tabela 6.3: Problemas resolvidos com as abordagens PCH e PCH-SE.

Esse aumento no número de problemas resolvidos com as 5 tolerâncias adotadas, também é observado na Tabela 6.3. Nessa tabela, comparamos o número de problemas resolvidos entre a abordagem pré-condicionador híbrido, PCH, e a abordagem que utiliza o pré-condicionador híbrido nas primeiras iterações e o sistema estável nas últimas iterações, PCH-SE.

Analisando a Tabela 6.3, verificamos um aumento no número de problemas resolvidos, quando comparamos as abordagens PCH e PCH-SE, em todas as tolerâncias utilizadas. Quando consideramos a tolerância de 10^{-8}, a abordagem PCH não resolve os problemas 80BAU3B e SCRS8, enquanto a abordagem PCH-SE o resolve com essa tolerância.

Enquanto o PCH resolve apenas 48 problemas com tolerância 10^{-10}, quando acrescentamos a abordagem do sistema estável nas últimas iterações, esse número aumenta para 69. Destacamos aqui os problemas: ADLITTLE, DEGEN3, FORPLAN, MAROS, SCTAP1, SCTAP3, SHELL, SHIP04L, SHIP04S, SHIP08L, SHIP12S, STANDATA, VTPBASE e WOODW. Esses 14 problemas são resolvidos pelo PCH com uma precisão de 10^{-10}, enquanto o PCH-SE os resolve com uma precisão de 10^{-16}.

Novamente, assim como na comparação da Tabela 6.2, a maior diferença no número de problemas resolvidos, agora entre as abordagens PCH e PCH-SE, ocorre quando consideramos 10^{-14} como tolerância. A abordagem PCH resolve apenas 10 problemas com essa tolerância, enquanto a abordagem PCH-SE resolve 54 problemas, ou seja, uma diferença de 44 problemas resolvidos entre as duas abordagens.

Uma diferença de 41 problemas resolvidos entre as duas abordagens, PCH e PCH-SE, pode ser observada quando consideramos 10^{-12} como tolerância. Agora, a abordagem PCH resolve 18 problemas enquanto a abordagem PCH-SE resolve 59 problemas.

Com a abordagem do sistema estável, 30 problemas passaram a ser resolvidos com uma tolerância de 10^{-16}, enquanto o PCH não resolve nenhum problema com essa tolerância. Podemos destacar os problemas LOTFI, SC205, SCAGR25, SCTAP2, STOCFOR1 e TUFF. Esses problemas são resolvidos pelo PCH-SE com uma precisão de 10^{-16} enquanto o PCH os resolve com uma precisão de, apenas, 10^{-8}.

Realizando uma comparação entre as duas abordagens, PCH e PCH-SE, com relação à porcentagem dos 90 problemas, da coleção NETLIB, resolvidos com tolerância de 10^{-16}, verificamos que enquanto a abordagem PCH não resolve nenhum problema, com essa tolerância, a abordagem PCH-SE resolve $33,33\%$ dos problemas da NETLIB com tolerância de 10^{-16}.

Destacamos agora, o importante papel desempenhado pela perturbação desenvolvida em

nossos estudos. Na tabela abaixo, apresentamos o número de problemas resolvidos pela abordagem PCx-SE, nas 5 tolerâncias, mas agora sem ativar a perturbação.

ABORDAGEM	TOLERÂNCIA				
	10^{-8}	10^{-10}	10^{-12}	10^{-14}	10^{-16}
PCx-SE s/ perturbação	87 (1)	81 (3)	65 (12)	52 (32)	21 (21)

Tabela 6.4: Problemas resolvidos com a abordagem PCx-SE sem perturbação.

Comparando essa tabela com a última linha da Tabela 6.2, verificamos que apenas na tolerância de 10^{-8} obtemos a mesma quantidade de problemas resolvidos, 87. Para as demais tolerâncias verificamos uma redução no número de problemas resolvidos, quando não ativamos a perturbação. A maior redução no número de problemas resolvidos ocorre na tolerância de 10^{-14}, com uma redução de 30 problemas.

Agora, dos 21 problemas resolvidos na tolerância de 10^{-16}, 7 problemas não são resolvidos, com essa tolerância, quando utilizamos a perturbação. Esses 7 problemas são: DEGEN2, DEGEN3, FORPLAN, SCRS8, SCSD1, SEBA e SHIP04L. Destes 7 problemas, 5 são resolvidos pelo sistema estável com uma precisão de 10^{-14}, utilizando a perturbação. Os problemas SCSD1 e SEBA são resolvidos pelo PCx com uma precisão de 10^{-14}, sem ativar a abordagem do sistema estável.

PROBLEMA	PCx		PCx-SE	
	Iter.	Tempo	Iter.	Tempo
LOTFI	14	4×10^{-3}	14	0,01
PILOT	41	3,06	42	6,37
PILOT4	51	0,13	50	0,24
SCSD6	13	0,02	15	0,26
WOODW	31	0,16	31	0,44

Tabela 6.5: Comparação de tempo e iterações entre PCx e PCx-SE, 10^{-10}.

Nossa próxima comparação, consiste no número de iterações e o tempo utilizado pelo sistema estável para que os problemas sejam resolvidos com essas tolerâncias. Um pequeno aumento no tempo de resolução e no número de iterações significa que o sistema estável é uma abordagem eficiente, computacionalmente. Um grande aumento, significa que o sistema estável é uma abordagem muito cara, computacionalmente, para se obter uma melhor precisão nos resultados.

Precisamos então, analisar o número de iterações e o tempo computacional gastos pelas iterações PCx e PCx-SE para resolver cada problema. No entanto, só poderemos realizar uma comparação entre os problemas que foram resolvidos pelas duas abordagens, em uma mesma tolerância.

Assim, apresentamos na Tabela 6.5 o tempo, em segundos, e o número de iterações dos problemas resolvidos pela abordagem PCx e pela abordagem PCx-SE, em que o sistema estável foi ativado, com tolerância de 10^{-10}. Nas tabelas 6.6 e 6.7, apresentamos a mesma comparação da Tabela 6.5, porém com uma tolerância de 10^{-12} e 10^{-14}, respectivamente.

PROBLEMA	PCx			PCx-SE	
	Iter.	Tempo		Iter.	Tempo
25FV47	26	0,17		26	0,24
D2Q06C	37	1,54		30	2,36
DEGEN3	25	1,10		24	3,35
ETAMACRO	32	0,05		33	0,05
GANGES	21	0,10		22	0,48
GROW15	25	0,06		21	0,10
LOTFI	14	8×10^{-3}		14	0,01
NESM	53	0,21		36	0,51
PEROLD	35	0,15		35	0,24
PILOTJA	39	0,41		33	1,02
PILOTWE	52	0,18		51	0,23
SC105	10	4×10^{-3}		10	4×10^{-3}
SC205	11	4×10^{-3}		11	4×10^{-3}
SCSD6	14	0,02		23	0,48
STANDMPS	30	0,03		25	0,03
TUFF	19	0,03		19	0,03
WOODW	35	0,26		32	0,68

Tabela 6.6: Comparação de tempo e iterações entre PCx e PCx-SE, 10^{-12}.

Analisando a Tabela 6.5, verificamos que o maior aumento no número de iterações ocorreu no problema SCSD6, com um aumento de 2 iterações. Esse aumento ocasionou em um aumento de $0,24$ segundos, no tempo computacional. Para os demais 5 problemas, o número de iterações

e o tempo computacional foram semelhantes entre as abordagens PCx e PCx-SE.

Observando a Tabela 6.6, verificamos que 13, dos 17 problemas, apresentaram o número de iterações e o tempo de resolução semelhantes, entre as abordagens PCx e PCx-SE. Verificamos também, que 3 problemas apresentaram uma redução de no mínimo 6 iterações, com um aumento máximo de $0,80$ segundos, no tempo de resolução. Apenas o problema SCSD6 apresentou um grande aumento no número de iterações, o que ocasionou em um aumento de $0,46$ segundos no tempo de resolução.

PROBLEMA	PCx			PCx-SE	
	Iter.	Tempo		Iter.	Tempo
AFIRO	8	$*$		8	4×10^{-3}
BEACONFD	11	8×10^{-3}		11	8×10^{-3}
BNL2	38	0,78		47	2,59
DEGEN2	12	0,05		12	0,08
ETAMACRO	32	0,05		36	0,07
GROW15	43	0,07		23	0,27
GROW22	38	0,12		23	0,16
SC105	10	$*$		11	4×10^{-3}
SCSD6	14	0,03		28	0,57
SCTAP1	18	0,02		15	0,02
SHELL	21	0,04		23	0,03
TRUSS	26	0,28		21	0,47
TUFF	23	0,04		20	0,04
WOODW	50	0,30		36	0,78

Tabela 6.7: Comparação de tempo e iterações entre PCx e PCx-SE, 10^{-14}.

O símbolo $*$ significa que o problema foi resolvido com um tempo computacional menor que a tolerância exigida no código.

Realizando a comparação com base na tabela acima, observamos que 9 problemas apresentaram número de iterações e tempo computacional semelhantes, entre as abordagens PCx e PCx-SE. Destacamos os problemas: GROW15, GROW22 e WOODW que apresentaram uma redução de 20, 15 e 14 iterações, respectivamente, e um pequeno aumento no tempo computacional. Os problemas BNL2 e SCSD6 apresentaram um aumento de 9 e 14 iterações, respectivamente,

porém um aumento máximo de $1,80s$ no tempo de resolução.

As tabelas 6.8, 6.9 e 6.10 fornecem as mesmas comparações de tempo e iterações, que as tabelas 6.5, 6.6 e 6.7, porém, agora, essa comparação ocorre entre as abordagens PCH e PCH-SE, com tolerância de10^{-10}, 10^{-12} e 10^{-14}, respectivamente.

PROBLEMA	PCH			PCH-SE	
	Iter.	Tempo		Iter.	Tempo
BLEND	11	0,04		11	0,04
RECIPE	13	0,02		31	0,03
SC50B	7	$*$		7	4×10^{-3}
SCTAP3	21	0,33		17	0,35
SIERRA	26	0,26		46	0,36
STANDATA	15	0,03		15	0,03

Tabela 6.8: Comparação de tempo e iterações entre PCH e PCH-SE, 10^{-10}.

PROBLEMA	PCH			PCH-SE	
	Iter.	Tempo		Iter.	Tempo
25FV47	28	2,36		28	2,34
AFIRO	9	10^{-3}		9	10^{-3}
BANDM	18	0,19		19	0,20
BLEND	11	0,03		12	0,02
SC105	11	4×10^{-3}		11	8×10^{-3}
SC50B	7	4×10^{-3}		7	4×10^{-3}

Tabela 6.9: Comparação de tempo e iterações entre PCH e PCH-SE, 10^{-12}.

Analisando a Tabela 6.8, verificamos que 4 problemas apresentaram número de iterações e tempo computacional semelhantes, na comparação entre as abordagens. Os problemas RECIPE e SIERRA apresentaram um aumento de 18 e 20 iterações, respectivamente. Esse aumento significativo no número de iterações não foi refletido no tempo computacional, desses problemas.

A Tabela 6.9 apresenta resultados muito semelhantes entre as abordagens PCH e PCH-SE. Apenas os problemas BANDM e BLEND necessitaram de uma iteração a mais na abordagem PCH-SE, em relação a abordagem PCH. Contudo, todos os 6 problemas apresentaram tempos computacionais muito semelhantes, entre as duas abordagens.

PROBLEMA	PCH			PCH-SE	
	Iter.	Tempo		Iter.	Tempo
AFIRO	9	10^{-3}		9	10^{-3}
FIT1D	20	0,10		20	0,12
SC50B	7	10^{-3}		8	4×10^{-3}

Tabela 6.10: Comparação de tempo e iterações entre PCH e PCH-SE, 10^{-14}.

Observando a Tabela 6.10, verificamos que os três únicos problemas resolvidos pelas abordagens PCH e PCH-SE apresentaram mesmo número de iterações e tempo computacional semelhantes. Apenas o problema SC50B apresentou aumento de uma iteração.

Dessa forma, observando as Tabelas 6.5, 6.6, 6.7, 6.8, 6.9 e 6.10, verificamos que o maior aumento no tempo de resolução foi de $3,30$ segundos. Adicionalmente, observamos que 48 problemas, considerando as 6 últimas tabelas, apresentaram um aumento no tempo computacional menor que 1 segundo.

Podemos concluir então, que o custo computacional para se obter uma melhor precisão na resolução dos problemas, utilizando a abordagem do sistema estável, é muito baixo comparado com os custos computacionais do PCx e PCH. O fato do tempo computacional, tanto do sistema estável quanto do PCH ser maior que o tempo gasto no PCx, se deve ao cálculo da matriz B.

Durante os experimentos, observamos que a perturbação mais utilizada foi a perturbação δ_3, utilizada na maioria dos casos na etapa preditora. A segunda perturbação mais utilizada foi a perturbação δ_1, utilizada na etapa corretora.

Apenas os problemas 80BAU3B e GFRDPNC utilizaram a perturbação δ_2 na etapa preditora e na etapa corretora. Os problemas GROW15 e PILOT utilizaram a perturbação δ_1, também, nas duas etapas. Os problemas CAPRI e PILOTWE utilizaram a perturbação δ_1 na etapa preditora e a perturbação δ_2 na etapa corretora.

Dos 90 problemas da coleção NETLIB, apenas 28 não precisaram ser perturbados.

Realizamos, também, testes com outros pontos iniciais para o método 5.1. Além de utilizarmos $\Delta\tilde{y} = 0$, utilizamos $\Delta x_L = 0$ e $\Delta x_U = 0$; $\Delta\tilde{y} = r1$; $\Delta x_L = r_2$ e $\Delta x_U = r_3$.

A utilização de r_1, ou r_2 e r_3, se mostrou menos eficiente do que a inicialização com o vetor nulo. Com esses pontos iniciais, a implementação do sistema estável necessita de mais iterações para convergir e, em alguns problemas, não ocorre a convergência. Além disso, nenhum dos problemas que não convergiram com $\Delta\tilde{y} = 0$, tiveram convergência com o novo ponto inicial.

Adicionalmente, experimentamos uma outra forma de escolher a matriz B. Vimos na Seção 3.3.1 que a matriz B era escolhida como sendo as primeiras m colunas linearmente independentes de A, ordenadas de acordo com os valores da norma das colunas de AD^{-1}, em ordem crescente. Utilizando essa ordenação segundo os valores da norma das colunas de D^{-1} não se mostrou muito eficiente. Novamente alguns problemas deixaram de convergir e não obtivemos convergência para nenhum novo problema.

Ainda em busca de uma nova forma para escolhermos a matriz B, utilizamos o reordenamento das colunas da matriz A de acordo com os valores da norma das colunas de X, X^{-1}, Z e Z^{-1}, mas, novamente, nenhuma dessas abordagens apresentaram melhores resultados.

Conforme vimos na seção 4.4.3, caso o sistema estável não seja resolvido em 100 iterações, multiplicamos o valor da perturbação por π e iniciamos o método novamente. Isso acontece até que tenhamos a solução do sistema estável ou até que a perturbação seja igual a 1. Os resultados apresentados nesta seção foram obtidos considerando $\pi = 10$.

Na tentativa de obter melhores resultados, para os problemas degenerados, realizamos testes alterando a perturbação utilizada em nossos experimentos. Primeiramente, alteramos o valor de π para 5, ou seja, caso o problema não seja resolvido em 100 iterações, multiplicamos o valor da perturbação por 5. Em seguida alteramos o valor de π para 2.

Nestes dois testes, os resultados obtidos, com relação ao número de problemas resolvidos em cada iteração, foram muito semelhantes aos resultados obtidos quando consideramos $\pi = 10$. Porém, com relação ao tempo computacional, os melhores resultados foram obtidos quando consideramos $\pi = 10$.

Em nosso último experimento, reduzimos o número máximo de iterações, para se obter a solução do sistema estável, de 100 para 50. Com essa alteração, obtivemos a resolução dos mesmos problemas, quando consideramos o número máximo de iterações igual a 100. Exceto na tolerância de 10^{-16}. Nessa tolerância, quando consideramos o número máximo de iterações igual a 100, o problema ETAMACRO é resolvido, enquanto o problema GROW15 não é resolvido. Porém, quando consideramos o número máximo de iterações igual a 50, o problema ETAMACRO não é resolvido, enquanto o problema GROW15 é resolvido.

Capítulo 7

Conclusões e trabalhos futuros

Apresentamos uma abordagem com alta precisão, na resolução dos problemas, e eficiente para a solução dos sistemas lineares que surgem ao se determinar as direções de busca, nos métodos de pontos interiores. Utilizamos um pré-processamento, que consiste em uma eliminação de blocos na matriz do sistema linear. Em seguida, aplicamos nosso método para resolver a abordagem do sistema linear resultante, o sistema linear estável.

Reordenando as colunas da matriz A, de acordo com a ordem crescente da norma das colunas da matriz AD^{-1}, identificamos as primeiras m colunas linearmente independentes de A, segundo o ordenamento acima. Então, particionamos os vetores das variáveis de forma a se aplicar o método iterativo para a obtenção da solução do sistema linear estável.

O método funciona muito bem, quando estamos próximos de uma solução do problema. Um fato muito importante, pois as abordagens mais utilizadas tendem a se tornar mal-condicionadas quando próximas de uma solução.

Observamos que nosso método, para a resolução do sistema linear estável, também apresenta um bom desempenho para problemas com variáveis canalizadas. Por esse motivo, podemos dizer que o método pode ser aplicado à grande maioria dos problemas lineares que surgem de situações reais.

Através de nossos experimentos numéricos, mostramos que o método sistema linear estável é uma abordagem que apresenta uma alta precisão na solução de problemas lineares. Nossa abordagem apresenta uma precisão maior que a abordagem do sistema de equações normais, via decomposição de Cholesky e, também, via abordagem pré-condicionador híbrido. Isso se deve ao fato de a matriz da abordagem do sistema linear estável ser melhor condicionada do que a matriz do sistema de equações normais, quando próxima de uma solução do problema.

Constatamos que o método do sistema linear estável apresenta um melhor desempenho quando implementado nas últimas iterações de um método direto do que de um método iterativo, para resolução do sistema de equações normais. Esse melhor desempenho é reflexo do número de problemas resolvidos quando utilizamos nossa abordagem nas últimas iterações.

Podemos concluir, dessa forma, que os dois métodos, direto e iterativo, para a solução do sistema de equações normais, quando adicionado o método do sistema linear estável nas últimas iterações, passaram a ser mais robustos com relação à tolerância exigida.

Mostramos também que o custo computacional para se obter essa alta precisão foi um pequeno aumento no número de iterações, o que ocasionou em um pequeno aumento no tempo computacional. Isso se deve ao fato de o tempo de uma iteração no método do sistema linear estável ser muito menor do que o tempo de uma iteração no sistema de equações normais, quando próximo de uma solução do problema e desconsiderando o tempo transcorrido no cálculo da matriz B.

7.1 Trabalhos futuros

Para os problemas degenerados, verificamos que é necessária uma perturbação na matriz para a resolução do sistema linear estável. Destacamos aqui, nossa primeira proposta de pesquisas futuras, obter uma perturbação com melhores resultados.

Verificamos também, que ativar a perturbação para determinados problemas pode ser prejudicial para sua resolução. Assim, um estudo sobre qual o melhor momento de se ativar, ou não, uma perturbação, pode fazer com que o método sistema linear estável tenha um melhor desempenho.

Vimos no capítulo de experimentos numéricos que alguns problemas foram resolvidos com precisão de 10^{-12}, outros com precisão de 10^{-14} e alguns com precisão de 10^{-16}. Propomos aqui, uma alteração no código PCx, utilizando o método do sistema linear estável, que calcule a precisão com que um problema é resolvido.

O desempenho do sistema estável pode ser ainda melhor, se encontrarmos um momento ótimo para se ativar o método sistema linear estável. O momento da ativação do método, apresentado nesse trabalho, foi tomado com base nos experimentos numéricos.

Outra possibilidade de pesquisa, seria desenvolver uma versão do método sistema linear estável para ser ativado nas primeiras iterações sem comprometer a convergência do problema.

Essa versão poderia ser mais robusta e, com certeza, mais eficiente, com relação ao tempo de solução.

Finalmente, desenvolver um teste mais preciso na decisão de manter a decomposição LU atual. Uma decisão mais precisa de manter a matriz básica B que estamos trabalhando pode significar uma maior rapidez na solução de um problema. Em nossos experimentos, utilizamos o cálculo da matriz B no passo preditor de todas as iterações, quando o método é ativado.

Apêndice A

Método sistema linear estável - versão MAX

Apresentamos neste apêndice uma segunda versão do método para a resolução do sistema linear estável com variáveis canalizadas. Esse método se aplica a um sistema semelhante ao sistema (5.13) como veremos.

O método versão MAX também surge do sistema linear,

$$\begin{pmatrix} 0 & 0 & A^t & I & -J^t \\ A & 0 & 0 & 0 & 0 \\ J & I & 0 & 0 & 0 \\ Z & 0 & 0 & X & 0 \\ -WJ & 0 & 0 & 0 & S \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta y \\ \Delta z \\ \Delta w \end{pmatrix} = - \begin{pmatrix} r_d \\ r_p \\ r_u \\ r_a \\ r_c \end{pmatrix}, \tag{A.1}$$

onde $r_c = r_b - W r_u$.

Agora, assumimos que a matriz A no sistema acima possui a forma $A = (B_1 \quad B_2 \quad L \quad U)$, ou seja, assumimos que a matriz B pode ser particionada em $(B_1 \quad B_2)$. A divisão da matriz B depende das coordenadas de X_B. Uma coordenada B_i está em B_1, se $x_{B_i} \geq s_{B_j}$, ou se s_{B_j} não existe, onde a coordenada s_{B_j} corresponde a coordenada x_{B_i}. Caso $x_{B_i} < s_{B_j}$, então a coordenada B_i está em B_2.

Assumindo que a matriz A possui a estrutura acima, então a matriz J pode ser escrita da

seguinte forma,

$$J = \begin{pmatrix} J_{\tilde{B}_1} & & & \\ & J_{\tilde{B}_2} & & \\ & & J_{\tilde{L}} & \\ & & & J_{\tilde{U}} \end{pmatrix}.$$

Com um raciocínio análogo ao apresentado na seção 5.3, podemos concluir que os blocos $J_{\tilde{B}_2}$ e $J_{\tilde{U}}$ são matrizes identidade com dimensões B_2 e U, respectivamente.

Com a finalidade de evitar matrizes muito extensas em nosso trabalho, nos próximos passos mostraremos apenas os últimos 2 conjuntos de equações do sistema linear $(A.1)$. Os demais conjuntos de equações da matriz do sistema não sofrem alteração, por esse motivo, podem ser omitidos. Bem como as colunas nulas que venham a surgir.

Assim, o sistema linear obtido dos últimos 2 conjuntos de equações do sistema linear anterior é igual ao sistema formado pela matriz A_2 e pelo vetor d_2, onde a matriz A_2 é dada por

$$\begin{pmatrix} Z_{B_1} & & & & X_{B_1} & & & & & \\ & Z_{B_2} & & & & X_{B_2} & & & & \\ & & Z_L & & & & X_L & & & \\ & & & Z_U & & & & X_U & & \\ -W_{\tilde{B}_1} J_{\tilde{B}_1} & & & & & & & & S_{\tilde{B}_1} & \\ & -W_{B_2} I_{B_2} & & & & & & & & S_{B_2} \\ & & -W_{\tilde{L}} J_{\tilde{L}} & & & & & & & \\ & & & -W_U I_U & & & & & & \end{pmatrix}$$

e o vetor d_2 é dado por

$$d_2 = -\left(r_{a_{B_1}}, r_{a_{B_2}}, r_{a_L}, r_{a_U}, (r_c)_{\tilde{B}_1}, r_{c_{B_2}}, (r_c)_{\tilde{L}}, r_{c_U}\right)^t.$$

Como desejamos que X_{B_1} e S_{B_2} fiquem na diagonal principal do sistema estável, vamos mover X_{B_1}, S_{B_2}, X_L e S_U. Para realizar esse passo, utilizamos o primeiro conjunto de equações de $(A.1)$. Assim, o sistema linear formado pela matriz A_2 e pelo vetor d_2 é semelhante ao sistema

linear formado pela matriz A_3 e pelo vetor d_3, onde A_3 é a matriz

$$\begin{pmatrix}
Z_{B_1} & & & & -X_{B_1}B_1^t & & & X_{B_1}J_{\tilde{B}_1}^t & \\
& Z_{B_2} & & & & X_{B_2} & & & \\
& & Z_L & & -X_L L^t & & & & X_L J_{\tilde{L}}^t \\
& & & Z_U & & & X_U & & \\
-W_{\tilde{B}_1}J_{\tilde{B}_1} & & & & & & & S_{\tilde{B}_1} & \\
& -W_{B_2} & & & S_{B_2}B_2^t & S_{B_2} & & & \\
& & -W_{\tilde{L}}J_{\tilde{L}} & & & & & & S_{\tilde{L}} \\
& & & -W_U I_U & S_U U^t & & S_U & &
\end{pmatrix}$$

e

$$d_3 = -\left(-X_{B_1}r_{d_{B_1}} + r_{a_{B_1}}, r_{a_{B_2}}, -X_L r_{d_L} + r_{a_L}, r_{a_U}, (r_c)_{\tilde{B}_1}, S_{B_2}r_{d_{B_2}} + r_{c_{B_2}}, (r_c)_{\tilde{L}}, S_U r_{d_U} + r_{c_U}\right)^t.$$

Agora, utilizando o segundo conjunto de equações do sistema $(A.1)$, vamos eliminar os blocos Z_{B_1}, Z_{B_2}, $-W_{\tilde{B}_1}J_{\tilde{B}_1}$ e $-W_{B_2}$. Para isso, utilizamos as matrizes $(B^{-1})_1$ e $(B^{-1})_2$, onde $(B^{-1})_1$ são as primeiras B_1 linhas da matriz B^{-1} e $(B^{-1})_2$ são as linhas restantes de B^{-1}. Podemos verificar que

$$(B^{-1})_1 B_1 = I = (B^{-1})_2 B_2 \qquad \text{e} \qquad (B^{-1})_1 B_2 = (B^{-1})_2 B_1 = \mathbf{0},$$

onde $\mathbf{0}$ é a matriz nula.

Juntamente com esse passo, vamos eliminar os blocos $X_{B_1}J_{\tilde{B}_1}^t$, $X_L J_{\tilde{L}}^t$, S_{B_2} e S_U. Para essa eliminação, utilizamos os blocos $S_{\tilde{B}_1}$, $S_{\tilde{L}}$, X_{B_2} e X_U, da matriz A_3, respectivamente.

Dessa forma obtemos o sistema linear semelhante, formado pela matriz A_4 e pelo vetor d_4, dados por

$$A_4 = \begin{pmatrix}
K_1(B^{-1})_1 L & K_1(B^{-1})_1 U & -X_{B_1}B_1^t & & & & \\
-Z_{B_2}(B^{-1})_2 L & -Z_{B_2}(B^{-1})_2 U & & X_{B_2} & & & \\
Z_L + X_L J_L^t S_L^{-1} W_L J_L & & -X_L L^t & & & & \\
& Z_U & & & X_U & & \\
W_{\tilde{B}_1}J_{\tilde{B}_1}(B^{-1})_1 L & W_{\tilde{B}_1}J_{\tilde{B}_1}(B^{-1})_1 U & & & & S_{\tilde{B}_1} & \\
K_2(B^{-1})_2 L & K_2(B^{-1})_2 U & S_{B_2}B_2^t & & & & \\
-W_{\tilde{L}}J_{\tilde{L}} & & & & & & S_{\tilde{L}} \\
& -(W_U + S_U X_U^{-1} Z_U) & S_U U^t & & & &
\end{pmatrix},$$

$$d_4 = -\Big(r_1, \quad -Z_{B_2}(B^{-1})_2 r_p + r_{a_{B_2}}, \quad r_3, \quad r_{a_U}, \quad W_{\tilde{B}_1} J_{\tilde{B}_1}(B^{-1})_1 r_p + (r_c)_{\tilde{B}_1}, \quad r_2, \quad (r_c)_{\tilde{L}}, \quad r_4\Big)^t,$$

onde

$$\begin{aligned}
K_1 &= (-Z_{B_1} - X_{B_1} J^t_{\tilde{B}_1} S^{-1}_{\tilde{B}_1} W_{\tilde{B}_1} J_{\tilde{B}_1}) \\
K_2 &= (W_{B_2} + S_{B_2} X^{-1}_{B_2} Z_{B_2}) \\
r_1 &= X_{B_1} J^t_{\tilde{B}_1} S^{-1}_{\tilde{B}_1} [W_{\tilde{B}_1} J_{\tilde{B}_1}(B^{-1})_1 r_p + (r_c)_{\tilde{B}_1}] + Z_{B_1}(B^{-1})_1 r_p + X_{B_1} r_{d_{B_1}} - r_{a_{B_1}} \\
r_2 &= S_{B_2} X^{-1}_{B_2} [r_{a_{B_2}} - Z_{B_2}(B^{-1})_2 r_p] - W_{B_2}(B^{-1})_2 r_p - S_{B_2} r_{d_{B_2}} - r_{c_{B_2}} \\
r_3 &= X_L (J^t_{\tilde{L}} S^{-1}_{\tilde{L}} W_{\tilde{L}} J_{\tilde{L}} (r_c)_{\tilde{L}} + r_{d_L}) - r_{a_L} \\
r_4 &= S_U (X^{-1}_U Z_U r_{a_U} - r_{d_U}) - r_{c_U}.
\end{aligned}$$

Nosso sistema estável versão MAX, se encontra no primeiro, terceiro, sexto e oitavo conjunto de equações, do sistema acima. Podemos reescrevê-lo da seguinte forma:

$$\begin{pmatrix}
\begin{pmatrix} -X_{B_1} & \\ & S_{B_2} \end{pmatrix} & \begin{pmatrix} K_1 & \\ & K_2 \end{pmatrix} (B^{-1})L & \begin{pmatrix} K_1 & \\ & K_2 \end{pmatrix} (B^{-1})U \\
-X_L L^t B^{-t} & Z_L + X_L J^t_{\tilde{L}} S^{-1}_{\tilde{L}} W_{\tilde{L}} J_{\tilde{L}} & \\
S_U U^t B^{-t} & & -(W_U + S_U X^{-1}_U Z_U)
\end{pmatrix}
\begin{pmatrix} \Delta\tilde{y} \\ \Delta x_L \\ \Delta x_U \end{pmatrix} =$$

$$= \begin{pmatrix} \begin{pmatrix} r_1 \\ r_2 \end{pmatrix} \\ r_3 \\ r_4 \end{pmatrix}, \qquad \text{(A.2)}$$

onde $\Delta\tilde{y} = B^t \Delta y$.

Assim como o sistema estável em (5.13), o sistema linear acima possui os três blocos da diagonal principal invertíveis, enquanto os demais quatro blocos fora da diagonal principal convergem para zero, quando o sistema está próximo de uma solução.

Adicionalmente, o sistema $(A.2)$ também é um sistema por blocos, logo podemos utilizar o método bloco iterativo para encontrar sua solução. Após encontrarmos a solução do sistema

$(A.2)$, podemos encontrar as demais direções $(\Delta' s)$ da seguinte forma:

$$\begin{pmatrix}
\Delta w_{\tilde{B}_1} = \left(-S_{\tilde{B}_1}\right)^{-1}(W_{\tilde{B}_1} J_{\tilde{B}_1}(B^{-1})_1[r_p + L\Delta x_L + U\Delta x_U] + (r_c)_{\tilde{B}_1}) \\
\Delta z_{B_2} = X_{B_2}^{-1}(Z_{B_2}(B^{-1})_2[r_p + L\Delta x_L + U\Delta x_U] - r_{a_{B_2}}) \\
\Delta z_{B_1} = -rd_{B_1} - B_1^t \Delta y + J_{\tilde{B}_1}^t \Delta w_{\tilde{B}_1} \\
\Delta w_{B_2} = rd_{B_2} + B_2^t \Delta y + \Delta z_{B_2} \\
\Delta w_{\tilde{L}} = S_{\tilde{L}}^{-1}(W_{\tilde{L}} J_{\tilde{L}} \Delta x_L - (r_c)_{\tilde{L}}) \\
\Delta z_L = -r_{d_L} - L^t \Delta y + J_L^t \Delta w_L \\
\Delta z_U = X_U^{-1}(-r_{a_U} - Z_U \Delta x_U) \\
\Delta w_U = r_{d_u} + \Delta z_u + U^t \Delta y \\
\Delta x_B = B^{-1}(-r_p - L\Delta x_L - U\Delta x_U) \\
\Delta s_{\tilde{B}_1} = -(r_u)_{\tilde{B}_1} - J_{\tilde{B}_1} \Delta x_{B_1} \\
\Delta s_{B_2} = -(r_u)_{B_2} - \Delta x_{B_2} \\
\Delta s_{\tilde{L}} = -(r_u)_{\tilde{L}} - J_{\tilde{L}} \Delta x_L \\
\Delta s_U = -r_{u_U} - \Delta x_U
\end{pmatrix},$$

onde $\Delta x_B = \begin{pmatrix} \Delta x_{B_1} \\ \Delta x_{B_2} \end{pmatrix}$.

O método MAX foi implementado seguindo os mesmos critérios da implementação do método 5.1, porém, os resultados dos experimentos numéricos desse método foram inferiores aos resultados apresentados no capítulo 6.

Referências Bibliográficas

[1] I. Adler, N. Karmarkar, M G. C. Resende, and G. Veiga. Data structures and programming techniques for the implementation of Karmarkar's algorithm. *ORSA Journal on Computing*, 1:84–106, 1989.

[2] I. Adler, M. Resende, G. Veiga, and N. Karmarkar. An implementation of Karmarkar's algorithm for linear programming. *Mathematical Programming*, 44:297–335, 1989.

[3] E. D. Andersen and K. D. Andersen. Presolving in linear programming. *Mathematical Programming*, 71:221–245, 1995.

[4] K. D. Andersen. A modified shcur-complement method for handling dense solumns in interior-point methods for linear programming. *ACM Transactions on Mathematical Software*, 22(3):348–356.

[5] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programmin: Theory and Algorithms*. John Wiley & Sons, 1979.

[6] R. E. Bixby. Implementing the Simplex Method: the Initial Basis. *ORSA Journal on Computing*, 4:267–284, 1992.

[7] S. Bocanegra, F. F. Campos, and A. R. L. Oliveira. Using a hybrid preconditioner for solving large-scale linear systems arising from interior point methods. *Computational Optimization and Applications*, 36:149–164, 2007.

[8] J. L. Boldrini, S. I. R. Costa, V. L. Figueiredo, and H. G. Wetzler. *Àlgebra Linear*. Harbra, São Paulo, 1986.

[9] J. R. Bunch and B. N. Parlett. Direct methods for solving symmetric indefinite systems of linear equations. *SIAM Journal Numerical Analysis*, 8:639–655, 1971.

[10] F. F. Campos. *Analysis of Conjugate Gradients - type methods for solving linear equations.* PhD thesis, Oxford University Computing Laboratory, Oxford, 1995.

[11] F. F. Campos and N. R. C. Birkett. An efficient solver for multi-right hand side linear systems based on the CCCG(η) method with applications to implicit time-dependent partial differential equations. *SIAM J. Sci. Comput.*, 19(1):126–138, 1998.

[12] V. Chvátal. *Linear Programming.* W. H. Freeman, New York, 1983.

[13] J. Czyzyk, S. Mehrotra, M. Wagner, and S. J. Wright. PCx an interior point code for linear programming. *Optimization Methods & Software*, 11-2(1-4):397–430, 1999.

[14] G. B. Dantzig. Linear programming and extensions. *Princeton University Press, Princeton, NJ*, 1963.

[15] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* SIAM, Philadelphia, PA, 1996.

[16] I. I. Dikin. Iterative solution of problems of linear and quadratic programming. *Soviets Math. Doklady*, 8:674–675, 1967.

[17] J. Dominguez and M. González-Lima. A primal-dual interior-point algorithm for quadratic programming. *Numerical Algorithms*, 42:1–30, 2006.

[18] J. Dongarra, G. H. Golub, E. G., C. Moler, and K. Moore. Netlib and na-net: Building a scientific computing community. *IEEE Annals of the History of Computing*, 30:30–41, 2008.

[19] I. S. Duff. The solution of large-scale least-square problems on supercomputers. *Annals Oper. Res.*, 22:241–252, 1990.

[20] G. E. Forsythe and E. G. Straus. On best conditioned matrices. In *Proceedings of the Amer. Math. Soc.*, volume 6, pages 340–345, 1955.

[21] R. Fourer and S. Mehrotra. Performance on an augmented system approach for solving least-squares problems in an interior-point method for linear programming. *Mathematical Programming Society COAL Newsletter*, 19:26–31, 1992.

[22] A. George and E. Ng. An implementation of Gaussian elimination with partial pivoting for sparse systems. *SIAM J. Sci. Statist. Comput.*, 6:390–409, 1985.

[23] P. E. Gill, W. Murray, D. B. Ponceleón, and M. A. Saunders. Preconditioners for indefinite systems arising in optimization. *SIAM J. Matrix Anal. and Applications*, 13:292–311, 1992.

[24] G. H. Golub and Charles F. Van Loan. *Matrix Computations Third Edition.* The Johns Hopkins University Press, Baltimore, Maryland, 1996.

[25] J. Gondzio. HOPDM (Version 2.12) – A fast LP solver based on a primal–dual interior point method. *European Journal of Operational Research*, 85:221–225, 1995.

[26] J. Gondzio. Multiple centrality corrections in a primal-dual method for linear programming. *Computational Optimization and Applications*, 6:137–156, 1996.

[27] C. C. Gonzaga. Path-following methods for linear programming. *SIAM Review*, 34:167–224, 1992.

[28] M. González-Lima, H. Wei, and H. Wolkowicz. A stable primal-dual approach for linear programming under nondegeneracy assumptions. *Computational Optimization and Applications*, 44(2):213–247, 2009.

[29] E. Ng J. R. Gilbert and B. w. Peyton. An efficient algorithm to compute row and column counts for sparse cholesky factorization. *SIAM Journal on Matriz Analysis and Aplications*, 15:1075–1091, 1994.

[30] M. T. Jones and P. E. Plassmann. An improved incomplete Cholesky factorization. *ACM Trans. Math. Software*, 21:5–17, 1995.

[31] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.

[32] L. G. Khachian. A polynomial algorithm in linear programming. *Soviet Mathematics Doklady*, pages 20:191–194, 1979.

[33] V. Klee and G. J. Minty. How good is the simplex algorithm? In *O. Shisha, Editor, Inequalities-III*, pages 159–175, 1972.

[34] M. Kokima, S. Mizuno, and A. Yoshise. A primal-dual interior-point method for linear programming. progress in mathematical programming, interior-point and related methods, Spring-Verlag, new york. pages 29–47, 1989.

[35] J. W.-H. Liu. Modification of the minimum degree algorithm by multiple elimination. *ACM Transactions on Mathematical Software*, 11:141–153, 1985.

[36] J. W.-H. Liu. The role of elimination trees in sparse factorization. *SIAM Journal on Matrix Analysis and Applications*, 11:134–172, 1990.

[37] D. C. Luenberger. Hyperbolic pairs in the method of conjugate gradients. *SIAM Journal on Applied Mathematics*, 17:1263–1267, 1969.

[38] D. C. Luenberger. The conjugate gradient method for constrained minimization problemas. *SIAM Journal Numerical Analysis*, 7:390–398, 1970.

[39] D. G. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley, Reading, 1984.

[40] I. J. Lustig, R. E. Marsten, and D. F. Shanno. Computational experience with a primal-dual interior-point method for linear programming. *Linear Algebra Appl.*, 152:191–222, 1991.

[41] I. J. Lustig, R. E. Marsten, and D. F. Shanno. On implementing Mehrotra's predictor-corrector interior point method for linear programming. *SIAM Journal on Optimization*, 2:435–449, 1992.

[42] T. A. Manteuffel. An incomplete factorization technique for positive definite linear systems. *Math. Comp.*, 34(150):473–497, 1980.

[43] K. A. McShane, C. L. Monma, and D. F. Shanno. An implementation of a primal-dual interior-point method for linear programming. *ORSA Journal on Computing*, 1:70–83, 1989.

[44] D. Megiddo. Pathways to the optimal set in linear programming. Progress in mathematical programming, interior-point and related methods, Spring-Verlag. pages 131–158, 1989.

[45] S. Mehrotra. Implementations of affine scaling methods: Approximate solutions of systems of linear equations using preconditioned conjugate gradient methods. *ORSA Journal on Computing*, 4:103–118, 1992.

[46] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575–601, 1992.

[47] R. C. Monteiro and I. Adler. Interior path-following primal-dual algorithms, part 1: Linear programming. *Mathematical Programming*, 44:27–41, 1989.

[48] NETLIB LP repository: NETLIB collection LP test sets. http://www.netlib.org/lp/data.

[49] E. Ng and B. P. Peyton. Block sparse Cholesky algorithms on advanced uniprocessors computers. *SIAM Journal on Scientific Stat. Computing*, 14:1034–1056, 1993.

[50] A. R. L. Oliveira. A new class of preconditioners for large-scale linear systems from interior point methods for linear programming. Technical report, PhD Thesis, TR97-11, Department of Computational and Applied Mathematics, Rice University, Houston TX, 1997.

[51] A. R. L. Oliveira and D. C. Sorensen. A new class of preconditioners for large-scale linear systems from interior point methods for linear programming. *Linear Algebra and Its Applications*, 394:1–24, 2005.

[52] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM Journal Numerical Analysis*, 12:617–629, 1975.

[53] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. of Mathematical Software*, 8(1):43–71, 1982.

[54] L. F. Portugal, M.G. C. Resende, G. Veiga, and J. J. Júdice. A truncated primal-infeasible dual-feasible network interior point method. *Networks*, 35:91–108, 2000.

[55] M. G. C. Resende and G. Veiga. An implementation of the dual affine scalling algorithm for minimum cost flow on bipartite uncapaciated networks. *SIAM J. Optim.*, 3:516–537, 1993.

[56] M. A. G. Ruggiero and V. L. R. Lopes. *Cálculo Numérico - Aspectos Teóricos e Computacionais*. Pearson-Makron Books, São Paulo, 1997.

[57] R. A. Tapia and Yin Zhang. Superlinear and quadratic convergence of primal-dual interior point methods for linear programming revisited. *Journal of Optimization Theory and Applications*, 73:229–242, 1992.

[58] R. J. Vanderbei. Splitting dense columns in sparse linear systems. *Linear Algebra and its Applications*, 152:107–117, 1991.

[59] R. J. Vanderbei. *Linear Programming – Foundations and Extensions*. Kluwer Academics Publishers, Boston, USA, 1996.

[60] R. J. Vanderbei and T. J. Carpenter. Indefinite systems for interior point methods. *Mathematical Programming*, 58:1–32, 1993.

[61] M. I. Velazco, A. R. L. Oliveira, and F. F. Campos. A note on hybrid preconditions for large scale normal equations arising from interior-point methods. *Optimization Methods and Software*, 25(2):321–332, 2010.

[62] W. Wang and D. P. O'Leary. Adaptive use of iterative methods in predictor-corrector interior point methods for linear programming. *Numerical Algorithms*, 25(1–4):387–406, 2000.

[63] S. J. Wright. Modified cholesky factorization in interior-point algorithms for linear programming. *SIAM Journal on Optimization*, 9:1159–1191.

[64] S. J. Wright. *Primal–Dual Interior–Point Methods*. SIAM Publications, SIAM, Philadelphia, PA, USA, 1996.

[65] Y. Zhang. User's guide do LIPSOL - linear-programming interior point solvers. *Optimization Methods and Software*, pages 385–396, 1999. Special Issue on Interior Point Methods.