

Experiências computacionais com modelos de fluxo para múltiplos produtos com funções de custo não lineares e não separáveis

Luis Ernesto Torres Guardia †
João Carlos C. B. Soares de Mello †

† Departamento de Engenharia de Produção – Universidade Federal Fluminense
Brasil
tepletg@vm.uff.br
jcsmello@pesquisador.cnpq.br
<http://www.uff.br/decisao>

Abstract

In this paper we present the study and the numerical implementation of the primal-dual interior-point method for the solution of the convex nonlinear multicommodity network flow problem. At each iteration of the interior-point method, we solve the corresponding linear system, expressed by the augmented-indefinite system, using an indefinite preconditioned conjugate gradient algorithm combined with the AINV algorithm. We conduct some numerical experiments for networks of different dimensions and number of products and for some nonlinear costs. The computational results show the effectiveness of the interior-point method for this class of network problem.

Resumo

Neste trabalho são apresentados o estudo e a implementação numérica do método de pontos interiores primal-dual para o problema não linear convexo de fluxo em rede com múltiplos produtos. Em cada iteração do método de pontos interiores, resolve-se o correspondente sistema linear, expresso na forma de aumentado indefinido usando o algoritmo do gradiente conjugado com um pré-condicionador indefinido apropriado combinado com o algoritmo AINV. Foram realizados alguns testes numéricos para redes de várias dimensões e vários produtos, com funções de custos não lineares. Os resultados computacionais mostram a eficiência do método de pontos interiores para o caso de fluxo em rede para múltiplos produtos.

Keywords: Nonlinear programming. Interior-point method. Network flow problem

Title: Computational experiments with flow models for multicommodity using non linear and non separable cost functions.

1 Introdução

O presente trabalho trata da solução numérica de um problema de fluxo em rede, com vários produtos em circulação pelo mesmo arco. Considera-se que, de forma geral, o custo associado a cada arco é uma função não linear e não separável. Há ainda restrições de capacidade para os arcos.

Este problema pode ser representado em um grafo orientado $G = (N, E)$ em que N é o conjunto de nós e E o conjunto de arcos. O grafo representa uma rede no qual K produtos diferentes podem ser enviados a partir de certas origens até determinados destinos. Será usada a seguinte nomenclatura: \mathbf{b}_k é o vetor de suprimento/procura para cada produto k , onde $b_{ki} > 0$ representa o suprimento do produto k no nó i , e $b_{ki} < 0$ representa a procura do produto k no nó i . A matriz $\mathbf{A} \in \mathbb{R}^{m \times n}$ é a matriz de incidência nó-arco do grafo G . O vetor $\mathbf{x}_k = (x_{ka})$, $a \in E$ representa o fluxo do produto k nos arcos da rede, e $\mathbf{x}_v \in \mathbb{R}^n$ é o vetor da variável de folga sendo $\mathbf{x} = (\mathbf{x}_k)_{k=1, \dots, K}$. Considera-se que a matriz \mathbf{A} é de posto completo. Caso isso não aconteça, retira-se qualquer linha dessa matriz de forma que ela fique de posto completo. Supõe-se ainda que a função f é de classe C^2 (possui segunda derivada contínua).

O problema em estudo é apresentado em seguida:

$$\text{Minimizar } f(\mathbf{x}) = \sum_{k=1}^K f_k(x) \quad (1)$$

$$\text{sujeito a : } \mathbf{A} \mathbf{x}_k = \mathbf{b}_k, \quad k = 1, \dots, K \quad (2)$$

$$\sum_{k=1}^K \mathbf{x}_k + \mathbf{x}_v = \mathbf{b}_{mc}, \quad (3)$$

$$\mathbf{x}_v \geq \mathbf{0}, \mathbf{x}_k \geq \mathbf{0}, \quad k = 1, \dots, K. \quad (4)$$

A relação (1) é a função objectivo, a minimizar. A igualdade (2) é a restrição de conservação de fluxo na rede enquanto a (3) é conhecida como a restrição de capacidade sendo $\mathbf{b}_{mc} \in \mathbb{R}^n$. A restrição (4) expressa o facto de os fluxos serem não negativos.

Existem poucos trabalhos na literatura relacionados a este tipo de problema. Entre eles destaca-se o trabalho de Lawphongpanich [2000] que usa uma decomposição de Dantzig-Wolfe. A maioria dos trabalhos neste assunto trata com funções não lineares, mas que sejam separáveis. Entre eles pode-se citar os trabalhos de Goffin, *et al.* [1996] e Ouorou, *et al.* [2000], onde a optimização é realizada em função dos fluxos nos caminhos. O presente trabalho difere dos citados por fazer a optimização em função dos fluxos nos arcos.

Outros trabalhos que tratam de problemas relacionados a este foram apresentados por Babonneau e Vial [2005] que apresentam um método que combina plano de corte com a ideia do centro analítico. No que diz respeito às aplicações, Nagurney [2006] apresenta diversos modelos de equilíbrio para redes de transporte, de telecomunicações e de economia e finanças, formulados como um problema de inequações variacionais, em que no caso geral, os custos no arco da rede não dependem exclusivamente do fluxo nesse arco. Outras aplicações podem ser encontradas no trabalho de Nagurney [2006].

Quando o problema de optimização correspondente é formulado em função de fluxo nos arcos, o método tradicional é usar o algoritmo de Frank-Wolfe, ou seu caso melhorado. Também existe o método alternativo simplicial e de geração de colunas, dentre outros. Para maiores detalhes destes métodos, consultar o trabalho de Migdalas [2006].

Este trabalho é organizado da seguinte forma. Na secção 2 descreve-se brevemente o método de pontos interiores primal-dual. Na secção 3 desenvolve-se uma especialização do método de pontos interiores para o problema não linear de fluxo e a formação do respectivo sistema aumentado e sua solução. Na secção 4 apresentam-se os resultados computacionais para redes de diferentes dimensões e um número variados de produtos para algumas funções de custo não linear e não separáveis. Finalmente na última secção são apresentadas as conclusões.

2 Método de pontos interiores primal-dual

O método de pontos interiores, iniciado pelo algoritmo de programação linear de Karmarkar [1984] teve grande sucesso computacional por causa de sua complexidade polinomial. Este método foi posteriormente aplicado em programação não linear. As vantagens deste método adequam-no, principalmente, à resolução de problemas de grande porte. Para uma melhor compreensão da teoria de fluxo em rede, pode-se consultar o livro de Ahuja, et al. [1993]. Para a teoria de pontos interiores primal-dual, analisar o livro de Wright [1997].

Apresenta-se a seguir uma breve descrição do método de pontos interiores primal-dual para resolver o seguinte problema de programação não linear com restrições lineares:

$$\begin{aligned} & \text{Minimizar } f(\mathbf{x}) & (5) \\ & \text{sujeito a: } \mathbf{E}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

onde $\mathbf{x} \in \mathbb{R}^q$, $\mathbf{b} \in \mathbb{R}^p$ e \mathbf{E} é uma matriz, $\mathbf{E} \in \mathbb{R}^{p \times q}$, $p < q$ e de posto completo. Assume-se que a função objectivo $f: \mathbb{R}^q \rightarrow \mathbb{R}$ possui segunda derivada contínua e conhecem-se a primeira e segunda derivadas da função objectivo.

A função de Lagrange associada ao problema (5) é:

$$L(\mathbf{w}) = L(\mathbf{x}, \mathbf{y}, \mathbf{z}) = f(\mathbf{x}) + (\mathbf{E}\mathbf{x} - \mathbf{b})^t \mathbf{y} - \mathbf{x}^t \mathbf{z}$$

onde o multiplicador de Lagrange $\mathbf{y} \in \mathbb{R}^p$ corresponde às restrições de igualdades e $\mathbf{z} \in \mathbb{R}^q_+$ corresponde às restrições de desigualdades. As variáveis \mathbf{x} e \mathbf{z} são denominadas de variáveis primais, e a variável \mathbf{y} é denominada de variável dual.

As condições necessárias de primeira ordem de optimalidade de Karush-Kuhn-Tucker (KKT), ver El-Bakry *et al.* (1996), para a solução do problema (5) são dadas por:

$$\mathbf{F}(\mathbf{w}) = \begin{pmatrix} \nabla_x L(\mathbf{w}) \\ \mathbf{E}\mathbf{x} - \mathbf{b} \\ \mathbf{X}\mathbf{z} \end{pmatrix} = \mathbf{0} \quad (6a)$$

$$\mathbf{x} \geq \mathbf{0}, \quad \mathbf{z} \geq \mathbf{0}. \quad (6b)$$

sendo:

$$\begin{aligned} \nabla_x L(\mathbf{w}) &= \nabla_x f(\mathbf{x}) + \mathbf{E}^t \mathbf{y} - \mathbf{z}, \\ \mathbf{X} &= \text{diag}(x_1, x_2, \dots, x_q), \\ \mathbf{Z} &= \text{diag}(z_1, z_2, \dots, z_q), \\ \mathbf{e} &= (1, 1, \dots, 1)^t \in \mathbb{R}^q. \end{aligned}$$

Nesta formulação $\nabla_x L(\mathbf{w})$ é o gradiente de $L(\mathbf{w})$, e \mathbf{X} expressa a matriz diagonal tendo como componentes os elementos do vector \mathbf{x} . Analogamente, a matriz \mathbf{Z} também é diagonal e os componentes são os elementos do vector \mathbf{z} . As equações (6a) são denominadas condições primais, condições duais e condições de complementaridade, respectivamente.

O problema (5) pode ser aproximado introduzindo a função de barreira $\varphi_\mu(\mathbf{x})$, conforme apresentada no problema de optimização (7):

$$\begin{aligned} & \text{minimizar } \varphi_\mu(\mathbf{x}) := f(\mathbf{x}) - \mu \sum_{i=1}^q \ln(x_i) & (7) \\ & \text{sujeito a } \mathbf{E}\mathbf{x} = \mathbf{b}, \end{aligned}$$

onde $\mu > 0$ é denominado o parâmetro de barreira dado.

Sabe-se que se μ é suficientemente pequeno, o problema (7) é uma aproximação do problema original (5), ver Fiacco e McCormick (1968).

Assim, para $\mu > 0$, as condições de optimalidade de barreira de KKT associadas ao problema (7) são dadas por:

$$F_{\mu}(\mathbf{w}) = \begin{pmatrix} \nabla_x L(w) \\ Ex - b \\ XZe - \mu e \end{pmatrix} = \mathbf{0} \quad (8a)$$

$$\mathbf{x} > \mathbf{0}, \quad \mathbf{z} > \mathbf{0}, \quad (8b)$$

Note-se que se $\mu = 0$ e $\mathbf{x} \geq \mathbf{0}$, $\mathbf{z} \geq \mathbf{0}$, então as condições de KKT para o problema (8) coincidem com as condições de KKT do problema original (5). Por isso, a escolha do parâmetro μ tem um papel importante nos métodos de pontos interiores.

O algoritmo primal-dual determina a solução para uma sequência de problemas de barreiras (8) o qual dá lugar a uma sequência de parâmetros de barreira μ de tal modo a convergir para zero. Assim, o problema de barreira é resolvido para um valor fixo dado μ , isto é, resolve-se o sistema (8a) usando o método de Newton. Seja essa direcção de Newton $\mathbf{dw} = (\mathbf{dx}, \mathbf{dy}, \mathbf{dz})^t$, obtida através da linearização do sistema (8a), e definida pela solução do sistema de equações lineares:

$$\begin{pmatrix} \nabla_x^2 L(w) & E^t & -I \\ E & 0 & 0 \\ Z & 0 & X \end{pmatrix} \cdot \begin{pmatrix} dx \\ dy \\ dz \end{pmatrix} = - \begin{pmatrix} \nabla_x L(w) \\ Ex - b \\ XZe - \mu e \end{pmatrix} \quad (9)$$

onde $\nabla_x^2 L(\mathbf{w})$ é a matriz Hessiana de $L(\mathbf{w})$. Neste caso, tem-se $\nabla_x^2 L(\mathbf{w}) = \nabla^2 f(\mathbf{x})$, sendo $\nabla^2 f(\mathbf{x})$ a matriz Hessiana de $f(\mathbf{x})$. Assume-se que a matriz do sistema (9) é não singular. Essa matriz também é conhecida por matriz **KKT**.

A solução de um sistema linear deste tipo é necessária em cada iteração de um método de pontos interiores. Este, em termos computacionais, é um passo crucial do método. Esse sistema linear pode ser representado como um sistema simétrico indefinido, cuja solução pode ser obtida usando o algoritmo do gradiente conjugado com um pré-condicionador indefinido.

Seja na iteração k , $\mathbf{dw}_k = (\mathbf{dx}_k, \mathbf{dy}_k, \mathbf{dz}_k)^t$ solução do sistema (9), obtida por algum processo de resolução. Em uma nova iteração, determina-se um novo ponto $\mathbf{w}_{k+1} = (\mathbf{x}_{k+1}, \mathbf{y}_{k+1}, \mathbf{z}_{k+1})^t$ o qual é dado por:

$$(10) \quad \begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{dx}_k \\ \mathbf{y}_{k+1} &= \mathbf{y}_k + \alpha_k \mathbf{dy}_k \\ \mathbf{z}_{k+1} &= \mathbf{z}_k + \alpha_k \mathbf{dz}_k, \end{aligned}$$

sendo α_k o tamanho de passo determinado por um procedimento apropriado de busca de linha. Devido a que o sistema de restrições do problema (5) é linear, usa-se um esquema similar ao usado em programação linear para determinar α_k :

$$\alpha_{kmax} = \min\{ \min_i\{-(x_k)_i / (\Delta x_k)_i / (\Delta x_k)_i < 0\}, \min_i\{-(z_k)_i / (\Delta z_k)_i / (\Delta z_k)_i < 0\} \} \quad (11)$$

$$\alpha_k = \beta \alpha'_k, \quad \alpha'_k = \min\{ \alpha_{kmax}, 1 \} \quad (12)$$

para $\beta \in (0,1)$ e próximo de 1.

Com este novo ponto \mathbf{w}_{k+1} , actualiza-se o parâmetro de barreira μ segundo certas regras, forma-se um novo sistema (9) o qual é resolvido usando qualquer processo de solução, e o processo continua até que um critério de paragem do algoritmo de pontos interiores seja satisfeito.

O método de pontos interiores primal-dual pode ser descrito da seguinte maneira: Sejam dados os vectores $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{z} \in \mathbb{R}^n$ tal que $(\mathbf{x}, \mathbf{z}) > \mathbf{0}$ e um parâmetro de barreira $\mu > 0$. Determina-se a direcção de Newton $(\mathbf{dx}, \mathbf{dy}, \mathbf{dz})$ resolvendo o sistema linear equivalente ao sistema da relação (9). Escolhe-se um passo α , actualiza-se um novo ponto usando a relação (10). Finalmente um novo parâmetro de barreira $\mu > 0$ é determinado e o processo continua até que um certo critério de paragem do algoritmo primal-dual seja satisfeito.

3 Fluxo em rede para múltiplos produtos e o método primal-dual

Nesta secção aplicar-se-á o método de pontos interiores primal-dual apresentado na secção anterior para o caso de múltiplos produtos definido no problema (1)-(4).

Usando o segundo membro do sistema (9), define-se:

$$\begin{aligned}\xi_c &= \nabla_{\mathbf{x}} L(\mathbf{w}) = \nabla f(\mathbf{x}) + \mathbf{E}^t \mathbf{y} - \mathbf{z}, \\ \xi_b &= \mathbf{E}\mathbf{x} - \mathbf{b}, \\ \xi_\mu &= \mathbf{X}\mathbf{Z}\mathbf{e} - \mu\mathbf{e}.\end{aligned}$$

Uma forma de solução do sistema (9) consiste em eliminar a variável \mathbf{dz} desse sistema tal que:

$$\mathbf{dz} = -\mathbf{X}^{-1} (\xi_\mu + \mathbf{Z} \mathbf{dx}), \quad (13)$$

e assim tem-se o seguinte sistema simétrico indefinido, também denominado de sistema aumentado, ver Wright [1997]:

$$\begin{bmatrix} \nabla^2 f(x) + \mathbf{X}^{-1}\mathbf{Z} & \mathbf{E}^t \\ \mathbf{E} & \mathbf{0} \end{bmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix} = - \begin{pmatrix} \xi_c + \mathbf{X}^{-1}\xi_\mu \\ \xi_b \end{pmatrix} \quad (14)$$

Por facilidade de notação, o sistema (14) pode ser escrito da seguinte forma:

$$\begin{bmatrix} \mathbf{D} & \mathbf{E}^t \\ \mathbf{E} & \mathbf{0} \end{bmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \quad (15)$$

onde

$$\begin{aligned}\mathbf{v}_1 &= -(\xi_c + \mathbf{X}^{-1}\xi_\mu), \\ \mathbf{v}_2 &= -\xi_b \\ \mathbf{D} &= \nabla^2 f(\mathbf{x}) + \mathbf{X}^{-1}\mathbf{Z}.\end{aligned}$$

A matriz (15) é simétrica mas indefinida e algoritmos para realizar a decomposição de tal matriz são mais complicados. Por outro lado, este sistema tem vantagens quanto à estabilidade e flexibilidade. Para uma melhor compreensão da teoria de sistemas indefinidos e algoritmos de solução, consultar os livros de Wright [1997] e Nocedal e Wright [1999].

Um dos maiores esforços computacionais requerido no método primal-dual é determinar a solução do sistema linear (9), agora expresso como o sistema simétrico indefinido (14) ou (15).

Matrizes da forma (15) frequentemente aparecem na solução de problemas em muitas áreas, por exemplo na formulação de elementos finitos em problemas de equações diferenciais, em dinâmica de fluidos, etc. Para um extensa revisão deste assunto , ver Benzi, *et al.* [2005].

Um método directo para a solução, como o de decomposição triangular esparsa de Bunch e Parlett [1971] pode ter alto custo computacional em termos de tempo de processamento e espaço de armazenamento. Este problema é agravado no caso de problemas de grande porte, mesmo que os coeficientes da matriz sejam esparsos e esta esparsidade seja explorada. Por outro lado, a matriz associada a (15) é indefinida, já que possui autovalores positivos e negativos. Portanto, o sistema (15) não pode ser resolvido usando métodos interactivos, como o do gradiente conjugado, porque pode fracassar, a menos que um bom condicionador seja empregado. Por outro lado, existem diversos métodos interactivos que usam bons condicionadores para determinar a solução do sistema (15). Entre eles, citam-se os trabalhos de Keller *et al.* [2000], de Golub *et al.* [2001], de Bergamaschi *et al.* [2004], Luksan *et al.* [2005], de Bai e Wang [2006], de Dollar e Wathen [2006]. No presente trabalho, usar-se-á o algoritmo do gradiente conjugado pré-condicionado de Durazzi e Ruggiero [2003]. Ele promove a combinação com um pré-condicionador indefinido apropriado de tal forma a acelerar a convergência para a solução (menor tempo computacional) que o problema original (15). O esquema interactivo pode ser dado como segue:

- (1) Sejam \mathbf{dx}^0 , \mathbf{dy}^0 os vectores iniciais dados;
- (2) $\mathbf{r}_1^0 = \mathbf{v}_1 - \mathbf{D} \mathbf{dx}^0 - \mathbf{E}^t \mathbf{dy}^0$; $\mathbf{r}_2^0 = \mathbf{v}_2 - \mathbf{E} \mathbf{dx}^0$;
- (3) $\mathbf{z}_2^0 = \mathbf{R}^{-1} (-\mathbf{r}_2^0 + \mathbf{E} \mathbf{D}'^{-1} \mathbf{r}_1^0)$; $\mathbf{z}_1^0 = \mathbf{D}'^{-1} (\mathbf{r}_1^0 - \mathbf{E}^t \mathbf{z}_2^0)$;
- (4) $\mathbf{p}_1^0 = \mathbf{z}_1^0$; $\mathbf{p}_2^0 = \mathbf{z}_2^0$;
- (5) $\text{num} = (\mathbf{z}_1^0)^T \mathbf{r}_1^0 + (\mathbf{z}_2^0)^T \mathbf{r}_2^0$;
- (6) para $i = 0, 1, \dots$
- (7) $\mathbf{t}_1^i = \mathbf{D} \mathbf{p}_1^i + \mathbf{E}^t \mathbf{p}_2^i$; $\mathbf{t}_2^i = \mathbf{E} \mathbf{p}_1^i$;
- (8) $\text{den} = (\mathbf{p}_1^i)^T \mathbf{t}_1^i + (\mathbf{p}_2^i)^T \mathbf{t}_2^i$;
- (9) $\alpha = \text{num} / \text{den}$;
- (10) $\mathbf{dx}^{i+1} = \mathbf{dx}^i + \alpha \mathbf{p}_1^i$; $\mathbf{dy}^{i+1} = \mathbf{dy}^i + \alpha \mathbf{p}_2^i$;
- (11) $\mathbf{r}_1^{i+1} = \mathbf{r}_1^i - \alpha \mathbf{t}_1^i$; $\mathbf{r}_2^{i+1} = \mathbf{r}_2^i - \alpha \mathbf{t}_2^i$;
- (12) se um critério de paragem é satisfeito, parar;
- (13) $\mathbf{z}_2^{i+1} = \mathbf{R}^{-1} (-\mathbf{r}_2^{i+1} + \mathbf{E} \mathbf{D}'^{-1} \mathbf{r}_1^{i+1})$;
- (14) $\mathbf{z}_1^{i+1} = \mathbf{D}'^{-1} (\mathbf{r}_1^{i+1} - \mathbf{E}^t \mathbf{z}_2^{i+1})$;
- (15) $\text{den} = \text{num}$;
- (16) $\text{num} = (\mathbf{z}_1^{i+1})^T \mathbf{r}_1^{i+1} + (\mathbf{z}_2^{i+1})^T \mathbf{r}_2^{i+1}$;
- (17) $\beta = \text{num} / \text{den}$;
- (18) $\mathbf{p}_1^{i+1} = \mathbf{z}_1^{i+1} + \beta \mathbf{p}_1^i$; $\mathbf{p}_2^{i+1} = \mathbf{z}_2^{i+1} + \beta \mathbf{p}_2^i$;
- (19) final

Para iniciar o processo exposto , apresentam-se as matrizes \mathbf{X} e \mathbf{Z} para o caso de múltiplos produtos, que são as seguintes matrizes diagonais de bloco:

$$\mathbf{X} = \begin{bmatrix} X_1 & & & & & \\ & \cdot & & & & \\ & & \cdot & & & \\ & & & X_K & & \\ & & & & \cdot & \\ & & & & & X_\nu \end{bmatrix}, \quad \mathbf{Z} = \begin{bmatrix} Z_1 & & & & & \\ & \cdot & & & & \\ & & \cdot & & & \\ & & & Z_K & & \\ & & & & \cdot & \\ & & & & & Z_\nu \end{bmatrix},$$

sendo cada sub-matriz \mathbf{X}_k , $k=1, \dots, K$ uma matriz diagonal com componentes x_{ki} , $i=1, \dots, n$ para cada produto k , e \mathbf{X}_v é a matriz diagonal com componentes da variável de folga \mathbf{x}_v . Uma estrutura similar é dada para a matriz \mathbf{Z} .

Por outro lado, a matriz de restrições desse problema (1)-(4) é visualizada como segue:

$$\mathbf{E} = \begin{bmatrix} A & O & . & O & O \\ O & A & . & . & O \\ . & . & . & . & . \\ O & . & . & A & O \\ I & I & . & I & I \end{bmatrix} \quad (16)$$

onde cada bloco \mathbf{A} corresponde à matriz de incidência nó-arco para cada produto k , $k = 1, \dots, K$. As matrizes identidades \mathbf{I} correspondem às restrições de capacidade (3).

Finalmente, \mathbf{D} é uma matriz diagonal de blocos dada por:

$$\mathbf{D} = \begin{bmatrix} D_1 & & & & \\ & . & & & \\ & & . & & \\ & & & D_K & \\ & & & & D_v \end{bmatrix} \quad (17)$$

sendo cada sub-matriz \mathbf{D}_k , $k = 1, \dots, K$ é uma matriz dada por

$\mathbf{D}_k = [\nabla^2 f_k(\mathbf{x}) + \mathbf{X}_k^{-1} \mathbf{Z}_k]$ e sendo $\mathbf{D}_v = \mathbf{X}_v^{-1} \mathbf{Z}_v$ uma matriz diagonal associada ao vector de folga \mathbf{x}_v .

Do esquema interactivo anterior, escolhe-se \mathbf{D}' como a matriz diagonal de blocos da matriz \mathbf{D} , isto é, $\mathbf{D}' = \text{diag}(\mathbf{D}_k)$, $k=1, \dots, K$. Por outro lado, a escolha da matriz \mathbf{R} deve ser tal que se aproxime à matriz $\mathbf{E} \mathbf{D}'^{-1} \mathbf{E}^t$ e seja também uma matriz não singular. O formato da matriz $\mathbf{E} \mathbf{D}'^{-1} \mathbf{E}^t$ é dado em (18):

$$\mathbf{E} \mathbf{D}'^{-1} \mathbf{E}^t = \begin{bmatrix} AD_1^{-1} A^t & O & . & O & AD_1^{-1} \\ . & . & . & . & . \\ . & . & . & . & . \\ O & . & . & AD_K^{-1} A^t & AD_K^{-1} \\ D_1^{-1} A^t & . & . & D_K^{-1} A^t & D_v^{-1} + \sum_{k=1}^K D_k^{-1} \end{bmatrix} \quad (18)$$

Este esquema interactivo requer a inversa da matriz \mathbf{R} , isto é, a inversa da matriz $\mathbf{E} \mathbf{D}'^{-1} \mathbf{E}^t$. Como isto poderia ser um elemento complicador para o problema em estudo, usa-se a diagonal de blocos da matriz $\mathbf{E} \mathbf{D}'^{-1} \mathbf{E}^t$, isto é, $\mathbf{R} = \text{diag}(\mathbf{E} \mathbf{D}'^{-1} \mathbf{E}^t)$, ou seja \mathbf{R} é a matriz dada por:

$$\mathbf{R} = \begin{bmatrix} AD_1^{-1} A^t & O & . & O & O \\ . & . & . & . & . \\ . & . & . & . & . \\ O & . & . & AD_K^{-1} A^t & O \\ O & . & . & O & D_v^{-1} + \sum_{k=1}^K D_k^{-1} \end{bmatrix}$$

Assim, a inversa da matriz \mathbf{R} é determinada calculando as matrizes inversas de $\mathbf{A}\mathbf{D}'_k^{-1}\mathbf{A}^t$, para $k=1,\dots,K$. Deve ser lembrado que a última matriz de bloco de \mathbf{R} é uma matriz diagonal, portanto a sua matriz inversa é fácil de ser calculada.

Para calcular as matrizes inversas de $\mathbf{A}(\mathbf{D}'_k)^{-1}\mathbf{A}^t$, usa-se o algoritmo AINV de Benzi *et al.* [2000] tal que:

$$(\mathbf{A}(\mathbf{D}'_k)^{-1}\mathbf{A}^t)^{-1} \cong \mathbf{Z}_k \mathbf{P}_k^{-1} \mathbf{Z}_k^t, \quad k = 1, \dots, K.$$

Esse algoritmo é baseado em um processo de uma $(\mathbf{A}(\mathbf{D}'_k)^{-1}\mathbf{A}^t)$ – ortogonalização aplicado aos vectores unitários base \mathbf{e}_i . O algoritmo AINV de Benzi *et al.* [2000] é apresentado a seguir para obter a matriz inversa da matriz $\mathbf{B}_k = \mathbf{A}(\mathbf{D}'_k)^{-1}\mathbf{A}^t$, de dimensão m , onde \mathbf{e}_i é o i -ésimo vector unitário base, e por facilidade foi eliminado o sub-índice k :

- (1) Seja $z_i^{(0)} = \mathbf{e}_i \quad (1 \leq i \leq m)$
- (2) Para $i = 1, 2, \dots, m$ fazer
- (3) $\mathbf{v}_i = (\mathbf{A}\mathbf{D}'^{-1}\mathbf{A}^t) z_i^{(i-1)}$
- (4) Para $j = i, i+1, \dots, m$ fazer
- (5) $p_j^{(i-1)} = \mathbf{v}_i^T z_j^{(i-1)}$
- (6) final
- (7) se $i = m$ ir a (12)
- (8) Para $j = i+1, \dots, m$ fazer
- (9) $z_j^{(i)} = z_j^{(i-1)} - (p_j^{(i-1)} / p_i^{(i-1)}) z_i^{(i-1)}$
- (10) final
- (11) final
- (12) Seja $z_i = z_i^{(i-1)}$ e $p_i = p_i^{(i-1)}$, para $1 \leq i \leq m$. Retornar
 $\mathbf{Z} = [z_1, z_2, \dots, z_m]$ e $\mathbf{P} = \text{diag}(p_1, p_2, \dots, p_m)$

No passo (3) do algoritmo AINV anterior, pode-se observar que o método realiza o produto matriz-vector $(\mathbf{A}\mathbf{D}'^{-1}\mathbf{A}^t) z_i$. Para o presente problema de rede, não é necessário armazenar, nem a matriz \mathbf{A} , nem a matriz $\mathbf{A}\mathbf{D}'^{-1}\mathbf{A}^t$. Como a matriz \mathbf{D}' é diagonal pode-se aproveitar a estrutura que possuem tais matrizes na aplicação aos problemas de grande porte.

Com as observações anteriores, pode-se aplicar o algoritmo do gradiente conjugado com um pré-condicionador indefinido para o sistema aumentado (15).

Uma outra alternativa, em vez de aplicar o algoritmo AINV, seria determinar a decomposição de Cholesky da matriz $\mathbf{A}\mathbf{D}'^{-1}\mathbf{A}^t$, ver Castro [2000] para o caso da função objectivo ser linear ou (ver passo (13) da solução do sistema aumentado) resolver o sistema de equações:

$$\mathbf{R} \mathbf{z}_2^{i+1} = -\mathbf{r}_2^{i+1} + \mathbf{E}\mathbf{D}'^{-1} \mathbf{r}_1^{i+1},$$

isto é:

$$(\mathbf{A}\mathbf{D}'_k^{-1}\mathbf{A}^t) \mathbf{z}_2^{i+1} = -\mathbf{r}_2^{i+1} + \mathbf{E}\mathbf{D}'^{-1} \mathbf{r}_1^{i+1},$$

e neste caso, usa-se o esquema apresentado em Júdice, *et al.* [2003].

Com a solução do sistema (15), determinou-se \mathbf{dx} e \mathbf{dy} . Usando (13) determina-se então \mathbf{dz} e, novamente, um novo ponto interior é calculado usando a relação (10). O processo repete-se até que um critério de paragem do método primal-dual seja satisfeito.

Para o caso em que a função objectivo seja separável, o problema não linear de múltiplos produtos pode ser resolvido usando o sistema normal de equações, em vez do sistema aumentado, ver Torres [2006].

4 Resultados computacionais

Nesta secção, são apresentados os resultados computacionais para determinar a solução do problema não linear de fluxo em rede, de diferentes dimensões, para diversos produtos e diferentes funções de custos não lineares e não separáveis. As experiências foram realizadas usando o método de pontos interiores primal-dual, cujo correspondente sistema linear foi resolvido usando os diferentes métodos de solução discutidos anteriormente.

Todos os testes computacionais foram realizados em um microcomputador PC Duron, plataforma Windows XP, de 512 MB de memória RAM e 1600 MHz de frequência. Os códigos computacionais foram totalmente escritos em FORTRAN. Foi usada dupla precisão para todos os cálculos computacionais.

Na aplicação do método primal-dual, a sequência do parâmetro de barreira $\{\mu_k\}$ deve convergir para zero tão rápido quanto possível. Várias regras foram estudadas para o decréscimo desse parâmetro μ , ver por exemplo os trabalhos de Waltz *et al.* [2005], Wächter *et al.* [2006], Yamashita e Yabe [2005]. Neste trabalho usou-se a estratégia adoptada por Luksan *et al.* [2005] que segundo os autores desenvolve-se bem na prática. A regra é a seguinte:

$$\begin{aligned} \mu &= \sigma \mathbf{x}^t \mathbf{z} / np \\ &\text{onde} \\ \sigma &= 0,1 \min \{0,05 (1 - \rho) / \rho, 2\}^3 \\ &\text{e} \\ \rho &= \frac{\min(x_i z_i)}{x^t z / np} \end{aligned}$$

sendo np o número de variáveis.

Foram analisadas redes de diversas dimensões, baseadas na rede básica do trabalho de Nagurney [1984], p. 476. Essa rede básica consiste de 20 nós e 28 arcos e é usada para determinar a solução do problema de equilíbrio de tráfego em rede de transporte. Essa rede básica foi depois estendida para formar redes de grande porte. Para isso, foi implementado um programa especial FORTRAN para determinar a dimensão da nova rede, isto é, o número de arcos e de nós e os dois vectores que determinam a origem e o destino que definem cada arco da rede.

Em todos os testes computacionais do método primal-dual é necessário partir-se de um ponto inicial, não necessariamente viável. Para determinar o tamanho de passo α , foi usado o valor de $\beta = 0,99995$. Neste caso, aceita-se o primeiro passo α , sem usar qualquer função de mérito nem qualquer filtro, como é usado em problemas gerais da programação não linear.

O critério de paragem na solução do respectivo sistema linear simétrico indefinido foi fixado para:

$$\mathbf{r} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} - \begin{bmatrix} D & E^t \\ E & 0 \end{bmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix}$$

tal que $\|\mathbf{r}\| < \epsilon_1$, ϵ_1 uma determinada tolerância ou quando um número máximo de iterações é alcançado, digamos 600 iterações. Já o critério de paragem do método de pontos interiores é quando o valor da função objectivo no ponto actual é suficientemente próximo do valor da função objectivo no ponto anterior, isto é, a diferença relativa absoluta desses valores das funções objectivo é menor ou igual a 10^{-8} e o valor do parâmetro μ é próximo a zero. Nesse caso, verifica-se se $\|F(\mathbf{w})\| \leq \epsilon_2$, para ϵ_2 uma certa tolerância, por exemplo $\epsilon_2 = 10^{-3}$ ou quando

$$(|\text{gap}| / (1 + |\text{gap}|)) \leq \epsilon_3$$

sendo ϵ_3 é uma certa tolerância que depende do número de produtos.

Aqui, “gap” é a diferença da função primal, do problema (5), $f(\mathbf{x})$, e a função dual $d(\mathbf{x}, \mathbf{y}, \mathbf{z})$ dada por:

$$d(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \mathbf{b}^t \mathbf{y} - (\mathbf{x}^t \nabla f(\mathbf{x}) - f(\mathbf{x})).$$

Foram feitas análises para $K= 20, 50, 100, 500$ e 1000 produtos. Como cada variável de decisão \mathbf{x}_k , $k=1, \dots, K$ tem dimensão igual ao número de arcos, então a dimensão total de todas essas variáveis de decisão $\mathbf{x} = (\mathbf{x}_k)_{k=1, \dots, K}$ é igual ao número de arcos da rede multiplicado pelo número de produtos. Não foram levados em conta o número de componentes da variável de folga \mathbf{z} . Assim, por exemplo, o número total de variáveis para a rede de 390 arcos e 210 nós e 1000 produtos, como é um dos casos estudados, é de 390.000 variáveis.

No presente trabalho foram usados dois tipos de funções de custo não linear não separável, encontrados na literatura, indicados a seguir:

$$i) f(\mathbf{x}_k) = \sum x_{ki}^2 + \sum_{i < j} x_{ki} x_{kj}, \text{ para } \mathbf{x}_k = (x_{ki}) \text{ para produto } k$$

$$ii) f(\mathbf{x}_k) = \sum x_{ki}^4 + 0,5 \sum x_{ki}^2 + \sum_{i < j} x_{ki} x_{kj}, \text{ para } \mathbf{x}_k = (x_{ki}) \text{ para produto } k.$$

Shi [2004] usou funções deste tipo, mas trabalhou com o caso menos geral de programação convexa linearmente restringido e com um único produto.

A seguir são apresentados os resultados computacionais para a primeira função não linear.

$$\text{Função de custo } f(\mathbf{x}) = \sum x_i^2 + \sum_{i < j} x_i x_j$$

Tabela 4.1 – Rede de 105 arcos e 60 nós

Número de produtos	μ	gap	Número de iterações	Tempo (segundos)
20	10^{-12}	10^{-15}	17	3,24
50	10^{-12}	10^{-15}	15	13,47
100	10^{-12}	10^{-15}	16	17,24
500	10^{-12}	$0,572201 \times 10^{-5}$	9	19,73
1000	10^{-12}	10^{-15}	14	167,92

Tabela 4.2 – Rede de 390 arcos e 210 nós

Número produtos	μ	gap	Número iterações	Tempo (segundos)
20	10^{-12}	10^{-15}	17	42,97
50	10^{-12}	10^{-15}	18	111,81
100	10^{-12}	10^{-15}	15	203,65
500	10^{-12}	10^{-15}	15	1047,56
1000	10^{-12}	$0,174788 \times 10^{-2}$	15	2066,87

A seguir são apresentados os resultados computacionais para a segunda função.

$$\text{Função de custo } f(x) = \sum x_i^4 + 0,5 \sum x_i^2 + \sum_{i < j} x_i x_j$$

Tabela 4.3 – Rede de 105 arcos e 60 nós

Número produtos	μ	gap	Número iterações	Tempo (segundos)
20	10^{-12}	$0,619884 \times 10^{-5}$	9	0,63
50	10^{-12}	$0,762934 \times 10^{-5}$	9	1,50
100	10^{-12}	$0,114439 \times 10^{-4}$	9	2,94
500	10^{-12}	$0,596457 \times 10^{-1}$	9	14,92
1000	10^{-12}	$0,366077 \times 10^{-3}$	8	26,81

Tabela 4.2 – Rede de 390 arcos e 210 nós

Número produtos	μ	gap	Número iterações	Tempo (segundos)
20	10^{-12}	$0,343285 \times 10^{-2}$	25	37,97
50	10^{-12}	$0,555181 \times 10^{-2}$	25	94,63
100	10^{-12}	$0,532985 \times 10^{-1}$	24	182,24
500	10^{-12}	0,211540081	23	881,55
1000	10^{-12}	0,229695103	24	1845,61

Como pode ser observado, o valor do parâmetro de barreira μ é bastante pequeno em todos os problemas de teste, e o número de iterações necessárias para alcançar uma solução aceitável é razoável.

O algoritmo AINV comportou-se muito bem e é robusto para o caso de determinar a inversa da matriz $\mathbf{B}_k = \mathbf{A} (\mathbf{D}'_k)^{-1} \mathbf{A}^t$, $k=1, \dots, K$. Este algoritmo é um dos passos computacionalmente caro no método de pontos interiores primal – dual. Uma outra alternativa seria determinar a decomposição dessas matrizes usando o algoritmo de Cholesky, ver Castro [2000] para o caso do custo da função objectivo ser linear.

Pode-se observar também, que o método de pontos interiores para a maioria dos casos usando a primeira função, o *gap* é nulo. Já para a segunda função, esse *gap* é próximo a zero.

O tempo computacional, como era de esperar-se, aumenta quando o número de produtos também aumenta, mas que é bastante aceitável. O algoritmo de pontos interiores aplicado para fluxo em rede poderia ainda ser melhorado com refinamentos adicionais.

5 Conclusões

Neste artigo, o método de pontos interiores primal-dual foi desenvolvido para resolver problemas de fluxo em rede para múltiplos produtos. O algoritmo apresentado explora a estrutura da matriz de restrições de tal forma a não precisar armazenar nem a matriz de restrições nem o produto das matrizes envolvidas, o que pode ser útil para problemas de grande porte.

Foram construídas uma série de problemas não lineares não separáveis, para diferentes número de produtos (até 1000 produtos) e diferentes dimensões da rede.

A solução do sistema simétrico indefinido foi determinada usando o algoritmo do gradiente conjugado pré-condicionado através da diagonalização de cada sub-matriz

$\mathbf{D}_k = [\nabla^2 f_k(\mathbf{x}) + \mathbf{X}_k^{-1} \mathbf{Z}_k]$, $k=1, \dots, K$, obtendo-se a matriz diagonal \mathbf{D}' . Com isto foi possível aplicar o algoritmo AINV para determinar, aproximadamente, a inversa da matriz

$\mathbf{A} (\mathbf{D}'_k)^{-1} \mathbf{A}^t$. Esse algoritmo AINV mostrou-se robusto e comportou-se bem na prática.

Igualmente pode-se concluir a eficiência do método de pontos interiores para problemas de grande porte, apesar da solução, usando o *gap*, estar próximo à solução exacta. Pode-se aceitar que são erros satisfatórios para problemas de grande porte.

Futuros trabalhos incluirão o desenvolvimento de novos pré-condicionadores mais eficientes para este tipo de aplicação. Também poderão ser incluídos o uso de algoritmos numéricos em paralelo e/ou a comparação com o método de Cholesky.

6 Referências

- Ahuja, A.; Magnanti, T. & Orlin, J. (1993). *Network Flows: Theory, Algorithms and Applications*, Prentice-Hall, Inc. New Jersey.
- Babonneau, F. & Vial, J. (2005). ACCPM with a nonlinear constraint and an active set strategy to solve nonlinear multicommodity flow problems. http://www.optimization-online.org/DB_HTML/2005/06/1148.html. p. 1-29.
- Bai, Z. & Wang, Z. (2006). Restrictive preconditioners for conjugate gradient methods for symmetric positive definite linear systems. *Journal of Computational and Applied Mathematics*, 187, 202-226.
- Benzi, M.; Cullum, J. & Tuma, M. (2000). Robust approximate inverse preconditioning for the conjugate gradient method. *SIAM Journal on Scientific Computing*, 22, 4, 1318-1332.
- Benzi, M.; Golub, G. & Liesen, J. (2005). Numerical solution of saddle point problems. *Acta Numerica*, 14, 1-137.
- Bergamaschi, L.; Gondzio, J. & Zilli, G. (2004). Preconditioning indefinite systems in interior point methods for optimization. *Computational Optimization and Applications*, 28, 149-171.
- Bunch, J. & Parlett, B. (1971). Direct methods for solving symmetric indefinite systems of linear equations. *SIAM Journal of Numerical Analysis*, 8, 639-655.
- Castro, J. (2000). A specialized interior – point algorithm for multicommodity network flows. *SIAM Journal on Optimization*, 10, 3, 852 -877.
- Dollar, H. & Wathen, A. (2006). Approximate factorization constraint preconditioners for saddle-point matrices. *SIAM Journal on Scientific Computing*, 27, 5, 1555-1572.
- Durazzi, C. & Ruggiero, V. (2003). Indefinitely preconditioned conjugate gradient method for large sparse equality and inequality constrained quadratic problems. *Numerical Linear Algebra with Applications*, 10, 673-688.
- El-Bakry, A. ; Tapia, R. ; Tsuchiya, T. & Zhang, Y. (1996). On the formulation and theory of Newton interior-point method for nonlinear programming. *Journal of Optimization Theory and Applications*, 89, 507-541.
- Fiacco, A. e McCormick, G. (1968). *Nonlinear programming: Seqüencial unconstrained minimization technique*. John Wiley and Sons.
- Goffin, J.; Gondzio, R.; Sarkissian, R. & Vial, J. (1996). Solving nonlinear multicommodity flow problems by the analytic center cutting plane method. *Mathematical Programming*, 76, 131–154.
- Golub, G.; Wu, X. & Yuan, J. (2001). SOR-like methods for augmented systems. *BIT*, 41, 1, 71-85.

- Júdice, J., Patricio, J., Portugal, L., Resende, M. E Veiga, G. (2003). A study of the preconditioners of network interior point methods. *Computational Optimization and Applications*, 24, 5 – 35.
- Karmarkar, N. (1984). A new polynomial time algorithm for linear programming. *Combinatorica*, 4, 373-395.
- Keller, C.; Gould, N. & Wathen, A. (2000). Constraint preconditioning for indefinite linear systems. *SIAM Journal Matrix Analysis and Applications*, 21, 4, 1300-1317.
- Lawphongpanich, S. (2000). Simplicial with truncated Dantzig-Wolfe decomposition for nonlinear multicommodity network flow problems with side constraints. *Operations Research Letters*, 26, 33-41.
- Luksan, L.; Matonoba, C. & Vlcek, J. (2005). Interior point methods for large-scale nonlinear programming, *Optimization Methods and Software*, 20, 4-5, 569-582.
- Migdalas, A. (2006) . Nonlinear Programming in Telecommunications. In: *Handbook of Optimization in Telecommunications* [edited by M. Resende and P. Pardalos], Springer, 27-66.
- Nagurney, A. (1984). Comparative tests of multimodal traffic equilibrium methods, *Transportation Research*, 18B, 469-485.
- Nagurney, A. (2006). Supernetworks. In: *Handbook of Optimization in Telecommunications* [edited by M. Resende and P. Pardalos], Springer, 1073-1119.
- Nocedal, J. & Wright, S. (1999). *Numerical Optimization*, Springer-Verlag, New York.
- Ouorou, A.; Mahey, P. & Vial, J. (2000). A survey of algorithms for convex multicommodity flow problems. *Management Science*, 46, 1, 126-147.
- Shi, Y. (2004). A projected-steepest-descent potential-reduction algorithm for convex programming problems. *Numerical Linear Algebra with Applications*, 11, 883-893.
- Torres, L. (2006). Modelo não linear de fluxo em rede para multi-produtos. CD-ROM. XIII CLAIO – Congreso Latino-Iberoamericano de Investigación Operativa, 27 – 30 de Noviembre de 2006, Montevideo, Uruguay.
- Wächter A. & Biegler, L. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, *Mathematical Programming, Serie A*, 106, 1, 25-57.
- Waltz, R.; Morales, J.; Nocedal, J. & Orban, D. (2005). An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical Programming, Serie A*, 107, 3, 391-408.
- Wright, S. (1997). *Primal-Dual Interior-Point Methods*, SIAM, Philadelphia, Pa.
- Yamashita, H. & Yabe, H. (2005). Quadratic convergence of a primal -dual interior point method for degenerate nonlinear optimization problems, *Computational Optimization and Applications*, 31, 123-143.

