IMPLEMENTAÇÃO DE UM MÉTODO DE PONTOS INTERIORES PARA PROGRAMAÇÃO LINEAR

Aurélio Ribeiro L. de Oliveira *
CPFL - C.P. 1808
13085 - Campinas - S.P.

Christiano Lyra Filho
DENSIS - Faculdade de Engenharia Elétrica
C.P. 6101 - UNICAMP
13081 - Campinas - S.P.
E-mail: christiano@bruc.bitnet

Resumo - Este trabalho discute o desenvolvimento de algoritmos de pontos interiores para programação linear, com ênfase na família denominada afim. Destaca aspectos necessários à implementações computacionais eficientes, incluindo métodos para obtenção de soluções iniciais, consideração de variáveis canalizadas, critérios de convergência, representação esparsa de matrizes, decomposição e técnicas de reordenamento. Um código dual-afim é comparado como o MINOS na solução de uma grande quantidade de problemas clássicos.

Abstract - This paper discusses the development of interior point algorithms for linear programming, with special attention to the affine scaling family. The main aspects necessary to achieve efficient implementations are discussed, including methods to obtain initial solutions, variables with upper bounds, stopping rules, sparse matrices representation, decomposition methods and reordering heuristics. A dual-affine scaling code is compared with MINOS in the solution of a large number of classical problems.

1. INTRODUÇÃO

Em 1984 Karmarkar (1984) divulgou resultados formais sobre um novo algoritmo para resolver problemas de programação linear, cuja filosofia divergia radicalmente do simplex (Dantzig, 1963), consagrado mundialmente por 35 anos de aplicações. O grande atrativo teórico do método de Karmarkar (1984) foi apresentar, sob análise de pior caso, complexidade polinomial menor do que o método dos elipsóides (Khachiyan, 1979), o primeiro método polinomial para programação linear (mal sucedido computacionalmente).

Ao divulgar seu algoritmo, Karmarkar (1984) mencionou resultados de uma implementação sigilosa onde teria sido comprovada uma eficiência da ordem de 50 vezes melhor do que o simplex, para problemas de grande porte. Esta afirmação foi alvo de controvérsias, sobretudo porque seu autor não fornecia detalhes, invocando impedimentos contratuais em relação a seu empregador. O método apresentado deixava muitos pontos em aberto e implementações executadas por grupos independentes não conduziram de imediato a resultados conclusivos sobre sua suposta superioridade.

O método simplex busca uma solução ótima através de deslocamentos entre os pontos extremos da fronteira da região de factibilidade (vértices), usando a informação de que um desses pontos é solução ótima do problema. Ao contrário do simplex, o algoritmo proposto por Karmarkar (1984) baseia-se em pontos interiores à região factível e em sucessivas transformações projetivas que levam a um bom comportamento do processo.

A partir da idéia teórica de passos interiores e transformações, pode-se construir uma larga gama de algoritmos diferenciados, por exemplo, quanto ao tipo de transformação (projetiva, afim, etc), ao tratamento das restrições e às direções de busca (Gonzaga 1987; Oliveira 1989). Além das diferenças conceituais, existem ainda inúmeras alternativas quanto ao perfil tecnológico da implementação, envolvendo aspectos como tipo de decomposição para solução de sistemas lineares (Cholesky, LU, QR, etc) e estrutura de dados para tratamento da esparsidade. Assim, parece natural a existência de divergências quanto ao desdobramento das idéias de Karmarkar (1984).

Gill e outros (1986) apresentaram as primeiras experiências computacionais indicativas de que a nova abordagem poderia levar a algoritmos tão ou mais eficientes do que o simplex. Logo em seguida, Adler, Karmarkar,

* Atualmente no programa de Doutorado do Dept. of Mathematical Sciences, Rice University, P.O.Box 1892, Houston, TX, 77251, USA, E-mail:aoliv00@ricevm1.bitnet

Resende e Veiga (1989a) divulgaram o algoritmo dual afim, comparando-o com o MINOS 4.0 (uma conceituada implementação do simplex) em vários problemas clássicos. O método duas afim obteve melhor desempenho na maioria dos problemas, confirmando a competitividade dos métodos de pontos interiores.

Este trabalho descreve aspectos conceituais e computacionais envolvidos no desenvolvimento de códigos primal e dual afim. Discute os algoritmos básicos, o tratamento de variáveis canalizadas, formas para a obtenção de um ponto interior factível, critérios de convergência e aspectos importantes para a implementação. Estuda-se o comportamento do código em uma grande quantidade de problemas, comparando-o com o MINOS 2.5. Discute-se as situações onde sua aplicação parece mais indicada.

2. ALGORITMOS AFINS DE PONTOS INTERIORES

O método primal afim foi proposto inicialmente por Dikin (1967) e desenvolvido independentemente por vários outros autores (Barnes, 1986; Chandru & Kochar, 1986; Vanderbei, Meketon & Freedman 1986). A dedução a seguir é fundamentada na minimização do desvio da condição das folgas complementares (Dikin, 1967). Considere o seguinte problema de programação linear na forma padrão:

$$\begin{array}{lll} (P) & \text{Min } \mathbf{cx} & \text{e seu dual (D)} \quad \text{Max } \mathbf{by} \\ & \text{sa} \quad \mathbf{Ax = b} & \text{sa} \quad \mathbf{A^t y \le c} \\ & \mathbf{x \ge 0} & \mathbf{y} \quad \text{livre} \end{array}$$

Pelo teorema das folgas complementares, se existirem dois pontos x^* e y^*, respectivamente primal e dual factíveis, tais que

$$x^* (c - A^t y^*) = 0$$

então x^* é ótimo do primal e y^* é ótimo do dual.

Seja x um ponto factível do primal e $\gamma(y)$ o vetor dos desvios da condição de folgas complementares, tem-se:

$$\gamma(y) = D(c - A^t y), \quad \text{com} \quad D = \text{Diag}(x, \ldots, x)$$

Pelo teorema das folgas complementares, em uma solução ótima,

$$x_i > 0 \Rightarrow A_i^t y = c_i \quad \text{e} \quad x_i = 0 \Leftarrow A_i^t y < c_i$$

assim, $\gamma(y) = 0$ na otimalidade.

Definindo-se y^k como o vetor que minimiza $\gamma^t(y).\gamma(y)$, pode-se mostrar que:

$$y^k = (AD^2 A^t)^{-1} AD^2 c$$

A partir da estimativa das variáveis duais, y^k, calcula-se o vetor "custo relativo",

$$\hat{c} = c - A^t y$$

e uma direção

$$dx = -D^2 \hat{c}$$

obtida por um reescalamento do vetor custo relativo. É fácil verificar-se que dx é uma direção descedente e factível.

Conhecendo-se dx, calcula-se o novo ponto de maneira análoga à utilizada pela maioria dos métodos para otimização não linear. O tamanho máximo do passo é dado pela primeira componente de x a se anular.

A partir dos conceitos acima, pode-se afirmar o algoritmo primal afim (A1):

Seja x^0, um ponto inicial factível,

e $\beta \in (0,1)$

faça $k \leftarrow 0$

A1.

Repita

$D \leftarrow Diag(x_i)$ A1.1

$y \leftarrow (AD^2 A^t)^{-1} AD^2 c$ A1.2

$\hat{c} \leftarrow c - A^t y$ A1.3

$dx^k \leftarrow -D^2 \hat{c}$ A1.4

$\alpha \leftarrow \beta \, \text{Min}_i \{ -x_i^k / dx_i^k \text{ t.q. } dx_i^k < 0 \}$ A1.5

$x^{k+1} = x^k + \alpha dx$ A1.6

$k \leftarrow k + 1$

até convergir

A obtenção de um ponto interior inicial e o critério de convergência, dois aspectos importantes em relação ao algoritmo, serão discutidos mais adiante.

Chandru & Kochar (1986) mostram que este método, com uma pequena alteração, quando aplicado a uma solução inicial básica, equivale ao método simplex.

O algoritmo primal afim tem uma desvantagem do ponto de vista computacional: o cálculo da direção (**dx**) implica necessariamente em erros de arredondamento, resultando que a restrição $Ax^k = b$ não seja estritamente

respeitada. Isto pode causar problemas de convergência. O método dual afim, proposto por Adler, Karmarkar, Resende e Veiga (1989a), evita estes problemas numéricos.

Considere novamente os problemas (P) e (D). Supondo que ambos tenham solução ótima finita, sabe-se que o valor das soluções ótimas será o mesmo, e que qualquer delas pode ser obtida a partir da outra. Portanto, basta resolver um dos problemas para se obter a solução de ambos. O método dual afim trabalha com o problema na forma (D). Sua dedução pode também ser baseada na minimização do desvio da condição das folgas complementares (Oliveira, 1989).

Introduzindo as variáveis de folga v, em (D) obtemos:

$$\begin{aligned} (D)\quad & \text{MAx } \mathbf{by} \\ & \text{sa } A^t y + v = c \\ & v \geq 0 \\ & y \text{ livre} \end{aligned}$$

O algoritmo dual afim (A2), aplicado ao problema (D) na forma acima, pode ser resumido na seqüência de passos a seguir (Adler e outros, 1989a; Oliveira, 1989; Vanderbei, 1989):

Dados um ponto inicial factível y^0 e $\beta \in (0,1)$

$k \leftarrow 0$

A2.

Repita

$v \leftarrow c - A^t y$ A2.1

$D \leftarrow Diag(1/v_i)$ A2.2

$d_y \leftarrow (AD^2A^t)^{-1} b$ A2.3

$[\, x \leftarrow D^2 A^t d_y \,]$ A2.4

$d_v \leftarrow -A^t d_y$ A2.5

$\alpha \leftarrow \beta \text{ Min } \{ -v_j^k / dv_j^k \text{ t.q. } dv_j^k < 0 \}$ A2.6

$y^{k+1} = y^k + \alpha dy$ A2.7

$k \leftarrow k + 1$

até convergir

Neste algoritmo, o cálculo das variáveis primais (A2.4), entre colchetes, não é necessário a cada iteração (exceto se o valor da função objetivo primal for utilizado como critério de convergência).

3. VARIÁVEIS CANALIZADAS

Uma das grandes vantagens do método simplex é a possibilidade do tratamento das variáveis canalizadas implicitamente, sem aumentar a dimensão da matriz de restrições. Vanderbei (1989) mostra que problemas com variáveis primais canalizadas podem também ser resolvidos através do algoritmo afim de pontos interiores sem aumentar a dimensão das matrizes envolvidas. Qualquer das versões do algoritmo afim, primal ou dual, pode ser utilizada.

Vanderbei (1989) mostra que o cálculo da matriz diagonal D é a única alteração no algoritmo primal afim para a obtenção da direção dx e das variáveis duais y. Este resultado é muito importante, pois simplifica bastante o tratamento de variáveis canalizadas. O cálculo da matriz diagonal para variáveis canalizadas (D^*) fica:

$$(D^*)_i^i = x_i^2 w_i^2 / (x_i^2 + w_i^2)$$

onde $w_i = u_i - x_i$ (u_i é o limite superior).

Naturalmente, deve-se mudar o teste de bloqueio para impedir que as variáveis canalizadas ultrapassem o limite superior. Assim:

$$\alpha = \beta \text{ Min } \{ \text{ Max } (-x_i / dx_i, (u_i - x_i) / dx_i) \}$$

Vanderbei (1989), tomando certos limites em variáveis canalizadas e aplicando a abordagem em diversas formulações de PL, deriva vários outros algoritmos a partir do primal afim.

Seja o PL com variáveis (primais) canalizadas e livres (PC) e seu dual (DC):

$$\begin{aligned} (PC)\quad & \text{Min } [c_c \; c_l] \begin{bmatrix} x_c \\ x_l \end{bmatrix} \qquad (DC)\quad \text{Max } \mathbf{by} - \mathbf{uz} \\ & \text{sa } [A_c \; A_l] \begin{bmatrix} x_c \\ x_l \end{bmatrix} = b \qquad \text{sa } A_c^t y - z + v_c = c_c \\ & 0 \leq x_c \leq u \qquad A_l^t y + v_l = c_l \\ & x_l \text{ livre} \qquad y \text{ irrestrito; } z, v \geq 0 \end{aligned}$$

O algoritmo dual afim para problemas com variáveis primais canalizadas e livres, resumido na seqüência de passos a seguir, pode ser obtido a partir do primal afim para estes tipos de variáveis, uma das formas estudadas por Vanderbei (sua dedução é encontrada em Vanderbei, 1989, e Oliveira, 1989).

Dado um problema na forma (DC), (y^0, z^0) um ponto inicial factível e $\beta \in (0,1)$, faça:

$k \leftarrow 0$

A3. Repita

$v_c \leftarrow c_c - A_c^t y + z$

$v_1 \leftarrow c_1 - A_1^t y$ A 3.1

$D_c^* \leftarrow \text{diag}(1 / (v_i^2 + z_i^2))$

$D_1^* = \text{diag}(1 / v_i^2)$ A 3.2

$d_y \leftarrow (AD^* A^t)^{-1} (b - A_c D_c^* Z^2 u)$ A3.3

$x_c \leftarrow D_c^* (Z^2 u + A_c^t d_y)$

$x_1 \leftarrow D_1^* A_1^t d_y$ A 3.4

$d_z = Z^2 (x_c - u)$ A3.5

$d_v = -V^2 x$ A3.6

$\alpha \leftarrow \beta \text{ Min}_i \{\text{Min} (-v_i / dv_i, -z_i / dz_i)\}$ A3.7

$y^{k+1} = y^k + \alpha d_y$ A3.8

$z^{k+1} = z^k + \alpha d_z$ A3.9

$k \leftarrow k + 1$

até convergir

onde

$$D^* = \begin{bmatrix} D_c^* & 0 \\ 0 & D_1^* \end{bmatrix} \quad , \quad Z = \text{Diag}(z_i) , \quad V = \text{Diag}(v_i)$$

Assim como acontece no primal afim canalizado, aqui também não há aumento na dimensão do problema.

4. INICIALIZAÇÃO

A idéia por trás do método de inicialização aqui discutido é, num certo sentido, a mesma dos métodos que procuram uma solução básica factível no algoritmo simplex. Ou seja, transforma-se o problema de forma que se possa aplicar o próprio método ao problema modificado para encontrar um ponto interior factível do problema original. Assim, por analogia com o simplex, denominaremos o processo de inicialização da fase I.

A sugestão em Adler e outros (1989a) para o método dual afim tem obtido bons resultados. Ela consiste em chutar uma solução inicial (para a fase I) em função dos dados do problema, da seguinte forma:

$$y^0 = b \, \| c \| / \| A^t b \|$$

Note que $\| A^t y^0 \| = \| c \|$.

Se y^0 não for interior ao politopo dual ($v^0 \leq 0$), é usado um método semelhante ao M grande ("big M") do algoritmo simplex.

O método proposto por Adler e outros (1989a), consiste em acrescentar uma variável artificial dual irrestrita (y_{m+1}) com custo "infinito" ($b_{m+1} = -M$), e a respectiva coluna na matriz transposta ($(A^t)^{m+1} = -e$), onde e é o vetor com todos elementos iguais a 1.

Aplica-se então o algoritmo dual afim até que $y_{m+1} < 0$ (ou seja, $v > 0$), quando o algoritmo passa para o que chamamos de fase II, i.e., para a busca da solução ótima. Se o problema convergir com $y_{m+1} > 0$, o problema dual não tem solução factível.

Existe uma forma de calcular a direção das variáveis duais sem aumentar a dimensão do sistema a ser resolvido. Para isto, trata-se implicitamente a linha da variável artificial. Considere para tanto o seguinte vetor custo para a fase I: $b = [0 \quad -1]$ e a partição da matriz $\begin{bmatrix} A \\ -e^t \end{bmatrix}$. Tem-se o seguinte sistema a ser resolvido:

$$\begin{bmatrix} A \\ -e^t \end{bmatrix} D^* [A^t \; -e] \begin{bmatrix} d_y \\ d_{y_{m+1}} \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

Do primeiro sistema de equações obtemos:

$$AD^* A^t d_y = AD^* e \, d_{y_{m+1}}$$

A componente $d_{y_{m+1}}$ deve ser negativa, pois a função objetivo $[0 \; -1]$ deve ser maximizada. Pode-se, portanto, adotar qualquer valor negativo para $d_{y_{m+1}}$. Por exemplo, -1. Assim, a direção das outras variáveis será determinada pelo seguinte sistema:

$$AD^* A^t d_y = -AD^* e$$

Uma desvantagem de usar $M \rightarrow -\infty$ é que este método de inicialização não leva ao ponto ótimo em problemas que não têm ponto interior na forma dual. Isto pode ser contornado fazendo-se uma "fase complementar" (Oliveira 1989), que consiste em manter a variável artificial no problema, quando for detectado de forma heurística que o mesmo não tem ponto interior.

Neste trabalho, a inicialização de Adler e outros (1989a) foi estendida para o problema dual canalizado, calculando-se y^0 da mesma forma anterior e construindo-se z^0 com componentes z_i,

$$z_i = 1 / u_i$$

Um resultado importante, que pode ser deduzido do algoritmo dual afim canalizado, ocorre quando todas as variáveis primais são canalizadas. Neste caso, é *sempre possível encontrar um ponto dual factível sem necessidade de fase I.* Para tanto, considere as restrições matriciais do problema dual canalizado:

$$A\,y - z + v = c$$

Atribui-se um valor para y^0, por exemplo, através da heurística proposta por Adler e outros, descrita acima.

Quando

$$A_i^t y^0 \geq c_i,$$

faz-se

$$z_i^0 \leftarrow A_i^t\, y^0 - c_i + 1 / u_i\ .$$

Isto garante que $v^0 > 0$. Caso contrário, quando $A_i^t y^0 < c_i$, $z_i^0 \leftarrow 1 / u_i$, como sugerido acima (ou qualquer outro número positivo).

Uma outra forma de inicialização para o dual afim não canalizado, aplicável apenas quando $c > 0$, elimina a necessidade da fase I no dual. Neste caso, faz-se $y^0 = 0$, e $v^0 = c$.

5. CRITÉRIO DE CONVERGÊNCIA

Vale a pena lembrar que os métodos de pontos interiores não levam a um ponto na fronteira do politopo de restrições. Como a solução ótima de um PL é um ponto de fronteira, é necessário definir um critério de convergência que determine uma solução suficientemente próxima do ótimo e que detecte esta proximidade sem a necessidade de um número muito grande de iterações nesta região.

Adler e outros (1989a) propõem um critério baseado na variação da função objetivo:

$$| by^k - by^{k-1} | / \text{Max} \{ 1, | by^{k-1} | \} < \varepsilon \qquad (1)$$

e um outro baseado no desvio da condição das folgas complementares:

$$x_j^k / \| x^k \| \geq -\varepsilon_1 ; \qquad | x_j^k v_j^k | / \| x^k \| \| v^k \| \leq \varepsilon_2 \qquad (2)$$

Nos problemas com variáveis canalizadas, deve-se considerar também a variável de folga da canalização, ou seja, $w_j = - u_j - x_j$.

6. IMPLEMENTAÇÃO

6.1 Decomposição

Os métodos de pontos interiores procuram encontrar a solução de problemas lineares em um número razoavelmente pequeno de iterações. No entanto, as iterações de um algoritmo de pontos interiores têm complexidade (requer um esforço computacional) maior que as iterações do simplex, o método tradicional para solução de problemas lineares. Ou seja, na realização de cada iteração os métodos de pontos interiores são mais lentos do que o simplex.

Não entrando em detalhes, cada iteração do simplex envolve uma atualização de "rank 1" de uma matriz (pivoteamento), enquanto que uma iteração de pontos interiores envolve a resolução de um sistema linear.

A passagem mais cara dos métodos de pontos interiores, em termos de esforço computacional, envolve a resolução do seguinte sistema linear:

$$(AD_k^* A^t)\, p^k = q^k$$

O aspecto que leva ao maior ganho no tempo de cada iteração (além da economia em memória) é a consideração da estrutura esparsa das matrizes A e $AD_k^* A^t$. A resolução do sistema através do cálculo explícito da inversa está fora de cogitação, pois, mesmo que $AD_k^* A^t$ seja esparsa, o mesmo não necessariamente ocorre com a sua inversa.

Como a matriz $AD_k^* A^t$ é simétrica positiva definida, o sistema pode ser resolvido com a decomposição de Cholesky (LEL^t ou LL^t), onde L é triangular inferior e E é diagonal. Esta decomposição utiliza aproximadamente metade do esforço computacional da decomposição LU (veja que sendo LEU única $L^t = U$). Mais, ela é intrinsecamente estável quando a matriz decomposta é positiva definida, o que dispensa avaliações numéricas na escolha do pivô (Duff, Erisman & Reid 1986).

A decomposição (LEL^t) é preferível, em princípio. Ao contrário da LL^t, ela não necessita da operação raiz quadrada (que é cara em algumas máquinas) para o cálculo da matriz L.

Obtida a decomposição, tem-se o sistema na seguinte forma:

$$LEL^t\, p = q$$

Este sistema pode ser facilmente resolvido por substituição, em dois passos:

$$Lr = q$$

$$L^t p = E^{-1} r$$

Adler e outros (1989b) e Monma & Morton (1987) sugerem armazenar o produto $A_i^k A_j^k$, que permanece invariante durante o processo iterativo. Desta forma, consegue-se reduzir a metade o número de multiplicações no cálculo de $AD^* A^t$.

6.2 Atualização

A cada iteração, apenas a matriz diagonal do produto $B = AD^* A^t$ se modifica. Karmarkar (1984) demonstrou para o método projetivo que se os elementos de **D** que não modificaram "significativamente" ficarem inalterados, a convergência do método não fica comprometida; além disto, demonstrou que o número de elementos que variam muito é, na média, pequeno ($o(\sqrt{n})$ em m passos). Esta idéia pode ser utilizada para atualizar as matrizes **L** e **E**.

Sendo ϕ o conjunto de elementos que se modificaram significativamente, a nova decomposição é dada por:

$$LEL^t = \underline{L}\underline{E}\underline{L}^t + \sum_{i \in \phi} A^i \left[(D_i^i)^2 - (D_{-i}^i)^2 \right] (A^i)^t$$

Fletcher & Powell (1974) apresentaram um algoritmo de atualização da decomposição para a forma:

$$LEL^t = \underline{L}\underline{E}\underline{L}^t + \sigma z z^t$$

O algoritmo pode ser aplicado $|\phi|$ vezes (onde $|\phi|$ é a cardinalidade do conjunto ϕ) para se obter a nova decomposição. Eles também levam em conta o problema de instabilidade numérica na atualização e o fato da matriz **B** ser definida positiva.

Uma vez que na prática $|\phi|$ varia muito para os métodos primal e dual afim, podendo inclusive ser maior que m por várias iterações, optou-se pela atualização proposta em Irisarri & Sasson (1981) e Housos & Irisarri (1982), que é mais rápida e aplicável a uma matriz genérica, embora não leve em consideração a instabilidade numérica (o que não traz inconvenientes para o problema em estudo).

É possível calcular-se qual o caminho que faz menos operações, uma nova decomposição ou $|\phi|$ atualizações (Oliveira, 1989). O cálculo do número de operações, apesar de depender da estrutura das matrizes A e $AD^* A^t$, não é computacionalmente custoso; por isto, pode ser feito a cada iteração.

6.3 Reordenamento da Matriz

Já observamos que apenas a matriz diagonal D^* se altera a cada iteração. Logo, a estrutura de $AD^* A^t$ fica preservada durante todo o processo. Este fato permite que se faça um pré-processamento simbólico da matriz para identificar quais elementos são estruturalmente nulos, ou seja, são iguais a zero em todas as iterações.

É preciso também considerar que o processo de decomposição é capaz de modificar radicalmente a esparsidade estrutural do sistema através da ocorrência de "fill-ins" (enchimentos), que são elementos nulos em $AD^* A^t$ e não nulos em **L**. Neste caso, uma matriz esparsa pode resultar em uma matriz cheia após a decomposição, o que é indesejável.

O número de "fill-ins" é função da ordem em que estão colocadas as linhas e colunas de $AD^* A^t$ (que por sua vez é função da ordem das linhas de A). Logo, se pudermos encontrar a ordenação que garante o mínimo "fill-in", o número de operações por iteração será menor.

Infelizmente, o problema "minimum global fill-in" é NP-completo (Yannakakis, 1981), o que torna sua solução mais difícil do que o problema original. Entretanto, várias heurísticas têm sido usadas para reordenar matrizes de forma a preservar, mesmo que parcialmente, a esparsidade durante a fatoração (Duff, Erisman & Reid 1986). Duas delas, "minimum degree" (utilizado nesta implementação) e "minimum local fill-in" foram testadas com bons resultados para matrizes originadas de problemas de programação linear (deCarvalho 1987).

O método "minimum degree", para matrizes simétricas, foi proposto por Tinney & Walker (1967). Ele recebe esta denominação devido a sua interpretação em teoria de grafos. A estratégia consiste em escolher o nó com menor grau no grafo associado à matriz. Isto corresponde a selecionar a linha (e coluna, pois a matriz neste caso é simétrica) com menor número de elementos não nulos como a próxima linha do pivô. Após a escolha do pivô, repete-se o processo para a matriz resultante, sem atualização da matriz no esquema 1 (Tinney & Walker 1967), e com atualização, incluindo os "fill-ins", no esquema 2. O esquema 1 é mais rápido, porém, em geral, produz resultados piores em termos do número de "fill-ins", o que justifica a utilização do esquema 2.

O ponto crítico do reordenamento "minimum degree" é a atualização do grau dos nós resultantes após uma eliminação, pois é preciso levar em conta não só os "fill-ins" que surgem nesta eliminação, mas também em todas as anteriores. Após a escolha de um pivô (nó), apenas os nós

adjacentes a ele antes da sua escolha têm o grau alterado. Deve-se subtrair uma ligação referente ao nós retirado e acrescentar as ligações referentes aos novos "fill-ins". No grafo, um "fill-in" aparece quando dois nós adjacentes ao pivô não são adjacentes entre si.

Utilizando o conceito de clique (Duff, Erisman & Reid 1986), o reordenamento pode ser feito sem utilizar posições de memória para representar os "fill-ins". A localização dos "fill-ins" pode ser feita em uma fase subscqüente (Eisenstat, Schultz & Sherman 1981).

6.4 Estrutura de Dados

A consideração da esparsidade exige estruturas de dados sofisticadas para o armazenamento das matrizes A, $AD^* A^t$ e LEL^t. Uma vez que as operações realizadas pelas matrizes são diferentes, a forma de armazenamento também pode ser diferente. Por medida de economia de memória, os elementos das matrizes $AD^* A^t$ e LEL^t são armazenados nas mesmas posições.

Está claro que o esquema de armazenamento deve permitir percorrer as matrizes por linha e por coluna. Optou-se pela estrutura de armazenamento proposto por Knuth (1973) que tem estas facilidades. Porém, adotou-se uma inversão de apontadores na matriz L (Correia, Lyra & Oliveira, 1988; Oliveira, 1989) o que permite resolver o sistema triangular superior mais facilmente. É conveniente destacar que esta estrutura pode ser desvantajosa em aplicações que necessitam retirar ou acrescentar elementos à matriz. Não é o caso do algoritmo estudado, onde a estrutura da matriz é a mesma durante toda a resolução do problema.

O armazenamento dos elementos da matriz $AD^* A^t$ têm ainda um campo adicional que contém o número de pares $A_i^k A_j^k$ diferentes de zero para sua formação. Note que, quando o elemento for um "fill-in" de L, este campo conterá o valor zero. Desta forma, se todos os pares $A_i^k A_j^k$ e o índice da coluna k forem armazenados em um vetor, a montagem dos elementos de $AD^* A^t$ pode ser feita com a metade das multiplicações.

7. EXPERIÊNCIAS COMPUTACIONAIS

As experiências a seguir comparam a implementação do método dual afim com o MINOS (versão 2.5), um dos mais respeitados entre os códigos baseados no método simplex. Além disto, são feitas algumas comparações envolvendo o tratamento de variáveis canalizadas.

A maioria dos problemas testados formam o primeiro conjunto de problemas utilizados em Adler e outros (1989a) - **NETLIB**, problemas testes de programação linear (Dongarra & Grosse 1987). Além destes, foram utilizados quatro problemas, que têm a característica de trabalhar com variáveis canalizadas. Os problemas são os seguintes:

1) CSF3 - Problema de planejamento da operação de sistema hidrelétrico aplicado ao sistema do médio São Francisco (Carneiro, 1984; Correia, Lyra & Oliveira 1988);

2) ENER12G - Problema de coordenação da operação de um sistema multisetorial de suprimento de energia (problema de grafo generalizado) (Correia 1989; Correia & Lyra 1988);

3) IEEE24 - Problema de despacho ótimo de sistemas de potência aplicado ao sistema IEEE24 (grafo generalizado com restrições adicionais) (Carvalho 1986; Carvalho & Soares 1988);

4) CANTAR - Problema de operação ótima de um sistema de reservatórios aplicado ao sistema Cantareira (problema de fluxo de custo mínimo) (Andrade & Correia 1988).

Os principais parâmetros usados foram os seguintes:

- Tamanho do passo na fase **I**: 0,99
- Tamanho do passo na fase **II**: 0,95
- Precisão na variação da função objetivo: 10^{-8}

A linguagem "C" foi escolhida para a implementação, devido a sua portabilidade, rapidez e facilidade na utilização de apontadores. Os resultados mostrados a seguir, para o conjunto de problemas NETLIB, foram obtidos em um VAX 11/785, sistema operacional VMS v4.7, utilizando os compiladores VAX C 2.3 para o método de pontos interiores e VAX FORTRAN 4.6 para o MINOS. Em ambos os casos, as opções "default" são utilizadas, incluindo a otimização do código. Os parâmetros do MINOS são os usuais exceto: *rows*= 1500, *columns*= 6000, *elements*= 5000, *iterations*= 7500, *log frequency*= 200 e *solution*= NO. O outro conjunto de problemas foi testado em um microcomputador compatível ao IBM/PC, Cobra XPC v3.02, com coprocessador 8087, 8 Mhz e compilador Turbo C 1.5.

A tabela 1 resume as principais características do primeiro conjunto de problemas testados.

Â representa o número de elementos não nulos de A. Os dados referem-se aos problemas na forma padrão. O número de "fill-ins" foi obtido utilizando o reordenamento "minimum degree".

O tempo medido, da mesma forma que em Adler e outros (1989a), é o da subrotina DRIVER, que exclui a leitura e interpretação dos dados. Também como em Adler

e outros (1989a), a solução do problema E226 obtida pelo MINOS está muito distante do ótimo. Os tempos do método dual afim incluem o reordenamento.

O teste de convergência baseado nas folgas complementares não é realizado nos problemas BrandY e Scrs8, pois, nestes casos, ocorre "overflow". Por isso, o valor obtido para o problema BrandY está distante do ótimo. O número máximo de iterações em cada fase foi limitado em 300.

A tabela 4 mostra os dados e resultados obtidos com o segundo subconjunto de problemas considerando as restrições de canalização como parte da matriz A. A tabela 5 mostra os dados quando se considera implicitamente a canalização, na forma apresentada no item 3.

8. CONCLUSÕES E COMENTÁRIOS

Este trabalho discutiu o desenvolvimento de algoritmos de pontos interiores para programação linear, enfatizando a família denominada afim. Foram considerados aspectos importantes para implementações computacionais eficientes, incluindo conceitos desenvolvidos no âmbito do presente trabalho. Implementou-se um código dual afim, incorporando o conjunto de aspectos teóricos e tecnológicos apresentados. O código foi comparado com o MINOS 2.5 em

PROBLEMA	LINHAS	COLUNAS	Â	L̂	FILL-IN
Afiro	27	51	102	107	17
ADLittle	56	138	424	411	27
Scagr 7	129	185	465	767	138
Sc205	205	317	665	1172	516
Share2b	96	162	777	1040	169
Share1b	117	253	1179	1383	382
Scorpion	388	466	1534	2559	458
Scagr25	471	671	1725	2981	588
ScTap1	300	660	1872	2620	934
BrandY	220	303	2202	3450	689
Scsd1	77	760	2388	1392	259
Israel	174	316	2443	11599	372
BandM	305	472	2494	4664	940
Scfxm1	330	600	2732	4715	1482
E226	223	472	2768	3687	864
Scrs8	490	1275	3288	6646	4448
Beaconfd	173	295	3408	2901	59
Scsd6	147	1350	4316	2545	446
Ship04s	402	1506	4400	3654	340
Scfxm2	660	1200	5469	9647	3161
Ship041	402	2166	6380	4830	200
Ship08s	778	2467	7194	6238	732
ScTap2	1090	2500	7334	14805	8210
Scfxm3	990	1800	8206	14567	4828
Ship12s	1151	2869	8284	7576	1080
Scsd8	397	2750	8584	5879	1599
ScTap3	1480	3340	9734	18843	9977
CzProb	929	3562	10708	8391	390
25FV47	821	1876	10705	34789	22892
Ship081	778	4363	12882	9714	424
Ship121	1151	5533	16276	12328	504

Tabela 1. NETLIB.

PROBLEMA	FASE I ITERAÇÕES	TOTAL DE ITERAÇÕES	TEMPO SEGS.	FUNÇÃO OBJETIVO
Afiro	7	9	0,20	-4,6475315e02
ADLittle	34	89	4,23	2,2549495e05
Scagr 7	57	134	9,21	-2,3313897e06
Sc205	114	208	21,39	-5,2202057e01
Share2b	77	114	8,25	-4,1573227e02
Share1b	120	262	22,72	-7,6589298e04
Scorpion	91	148	18,45	1,8781247e03
Scagr25	250	680	144,35	-1,4753432e07
ScTap1	188	269	34,17	1,4122500e03
BrandY	234	388	50,93	1,5185098e03
Scsd1	110	242	21,59	8,6666666e00
Israel	102	331	47,41	-8,9664482e05
BandM	258	555	106,35	-1,5862801e02
Scfxm1	239	429	56,07	1,8416758e04
E226 *	127	599	87,18	-1,1638929e01
Scrs8	182	563	125,68	9,0429693e02
Beaconfd	15	48	6,49	3,3592485e04
Scsd6	183	605	96,63	5,0500000e01
Ship04s	97	240	43,11	1,7987146e06
Scfxm2	587	967	249,83	3,6660261e04
Ship041	58	345	77,92	1,7933245e06
Ship08s	71	327	98,73	1,9200982e06
ScTap2	459	899	357,06	1,7248071e03
Scfxm3	874	1400	519,01	5,4901253e04
Ship12s	207	604	242,79	1,4892361e06
Scsd8	763	1604	622,85	9,0500000e02
ScTap3	479	1107	589,99	1,4240000e03
CzProb	809	1650	627,55	2,1851967e06
25FV47	1052	5445	4338,39	5,5018460e03
Ship081	164	561	220,41	1,9090552e06
Ship121	577	1367	720,90	1,4701879e06

Tabela 2. MINOS 2.5.

uma grande quantidade de problemas. Os resultados obtidos mostram que os métodos de pontos interiores concorrem com o método simplex e até o superam em várias situações. Isto vale particularmente para problemas esparsos de maior porte.

A versão 2.5 do MINOS foi utilizada por ser a que se encontrava disponível para os testes. É possível que uma versão mais atualizada melhore o desempenho do simplex. Por outro lado, deve-se lembrar que as pesquisas sobre métodos de pontos interiores são muito recentes, havendo grandes perspectivas de ganhos adicionais no desempenho desses métodos.

Dois fatores parecem ser importantes para a boa performance do método implementado:

- o pequeno número de iterações para a convergência e seu crescimento lento com a dimensão do problema;
- a facilidade de se encontrar uma solução factível inicial (muitas vezes o processo de inicialização é o que toma a maior parte do tempo utilizado pelo simplex).

Por outro lado, em alguns tipos de problemas o método simplex ainda é o mais indicado. Por exemplo, problemas com a presença de colunas densas, como é o caso do problema ISRAEL. A utilização de métodos iterativos para a resolução do sistema linear melhora em muito a eficiência do método de pontos interiores para problemas com colunas densas (Adler e outros 1989a). Porém, isto não parece ser suficiente para superar o método simplex nestas situações.

PROBLEMA	FASE I ITERAÇÕES	TOTAL DE ITERAÇÕES	TEMPO SEGS.	FUNÇÃO OBJETIVO
Afiro	1	20	0,43	-4,6475316e02
ADLittle	1	24	2,20	2,2549495e05
Scagr 7	3	26	3,64	-2,3313897e06
Sc205	5	30	6,27	-5,2202060e01
Share2b	4	28	6,65	-4,1573227e02
Share1b	8	38	12,95	-7,6589299e04
Scorpion	5	44	22,04	1,8781477e03
Scagr25	3	29	17,60	-1,4753432e07
ScTap1	6	36	22,89	1,4122499e03
BrandY *	6	233	308,74	1,3424202e03
Scsd1	0	300	143,88	8,6666660e00
Israel	9	39	423,43	-8,9664482e05
BandM	7	32	45,00	-1,5862801e02
Scfxm1	10	39	60,17	1,8416758e04
E226	12	51	66,37	-1,8751931e01
Scrs8 *	14	43	114,80	9,0429596e02
Beaconfd	9	31	42,42	3,3592485e04
Scsd6	0	20	18,94	5,0499998e01
Ship04s	5	30	31,99	1,7987146e06
Scfxm2	10	43	142,06	3,6659516e04
Ship041	5	30	44,46	1,7933243e06
Ship08s	5	33	59,98	1,9200982e06
ScTap2	7	34	272,80	1,7248071e03
Scfxm3	10	42	214,46	5,4897951e04
Ship12s	5	40	84,89	1,4892361e06
Scsd8	0	21	39,66	9,0499999e02
ScTap3	7	36	341,32	1,4239999e03
CzProb	3	53	142,29	2,1825283e06
25FV47	11	59	1906,30	5,5018419e03
Ship081	5	35	104,85	1,9090552e06
Ship121	5	31	118,37	1,4701879e06

Tabela 3. Método Dual Afim.

Os problemas que não têm ponto interior na forma dual (BrandY, Scfxm1, E226, Scrs8, Beaconfd. Scfxm2, Scfxm3 e 25FV47) tiveram seus tempos aumentados devido a uma "fase complementar" para a determinação da solução ótima. Isto não ocorre nas implementações que utilizam a fase I como sugerida em Adler e outros (1989a). Por outro lado, na inicialização adotada, o tempo das iterações na fase I é menor, por não aumentar a dimensão do sistema linear a ser resolvido. Isto leva a ganhos nos problemas que têm ponto interior.

O método dual afim obteve resultados ruins para os problemas Scsd1 e BrandY. A realização de um passo com 90% do valor máximo consegue resultados muito melhores para o problema Scsd1. O mau comportamento do método para o problema BrandY também está ligado a existência de linhas redundantes.

A utilização de variáveis canalizadas implicitamente resultou em um ganho efetivo no número de operações por iteração, confirmado pelos tempos obtidos nos casos teste. Em alguns casos, a versão canalizada obteve tempos inferiores, mesmo quando convergiu em um número maior de iterações. A diferença do número de iterações nestes casos (canalização implícita ou explícita) se deve a formas distintas para obtenção do ponto inicial).

Uma observação interessante é que as restrições de canalização colocadas na matriz A não geram novos "fill-ins" em AD^2A^t, se for utilizado o reordenamento

PROBLEMA	LINHAS	COLUNAS	Â	L̂	FILL-IN
CSF3	168	294	566	779	205
Ener12G	460	684	1337	1792	403
IEEE24	91	114	257	364	30
Cantar	178	249	498	647	120

PROBLEMA	FASE I ITERAÇÕES	TOTAL DE ITERAÇÕES	TEMPO SEGS.	FUNÇÃO OBJETIVO
CSF3	1	24	29,93	-4.3000000e02
Ener12G	2	25	66,62	-0.0000100e00
IEEE24	1	20	11,81	-1.9620000e03
Cantar	1	24	23,62	-2.0930501e02

Tabela 4. Canalização na Matriz.

PROBLEMA	LINHAS	COLUNAS	Â	L̂	FILL-IN
CSF3	63	189	356	414	205
Ener12G	155	379	727	906	403
IEEE24	34	57	143	164	30
Cantar	57	128	256	284	120

PROBLEMA	FASE I ITERAÇÕES	TOTAL DE ITERAÇÕES	TEMPO SEGS.	FUNÇÃO OBJETIVO
CSF3	1	37	29,22	-4.3000002e02
Ener12G	2	21	38,83	-0.0000120e00
IEEE24	0	22	7,85	-1.9620001e03
Cantar	1	23	14,23	-2.0930501e02

Tabela 5. Canalização fora da Matriz.

"minimum degree". Isto porque, no grafo associado à matriz $AD^{*} A^{t}$, o nó que representa uma restrição de canalização forma um subgrafo completo com seus nós adjacentes. Neste subgrafo, ou o nó dessa restrição é escolhido primeiro (e não gera "fill-in" pois pertence a um grafo completo) ou algum outro nó que pertence a este grafo completo é escolhido primeiro. Neste caso, o grau deste nó é igual ao do nó que representa a restrição de canalização, ou seja, eles estão ligados aos mesmos nós e a escolha de qualquer deles não gera "fill-ins". Finalmente, a escolha de um nó que não pertence ao subgrafo não gera "fill-ins" adicionais com a linha de restrição de canalização.

9. AGRADECIMENTO

Os autores agradecem a Geraldo Veiga, por fornecer referências essenciais ao desenvolvimento deste trabalho.

A pesquisa foi parcialmente financiada pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq)

10. REFERÊNCIAS

ADLER, I.; KARMARKAR, N.; RESENDE M. G. C. & VEIGA, G. (1989a), An Implementation of Karmarkar's Algorithm for Linear Programming, *Mathematical Programming 44*, 297-335.

ADLER, I.; KARMARKAR, N.; RESENDE, M. G. C. & VEIGA G. (1989b), Data Structures and Programming Techniques for the Implementation of Karmarkar's Algorithm, *ORSA Journal on Computing 1*, 84-106.

ANDRADE, M. G. & CORREIA, P. B. (1988), Operação Ótima de um Sistema de Reservatórios com Algoritmos em Rede, *Anais do 7º Congresso Brasileiro de Automática* 2, 872-879. ITA, S. J. Campos SP.

BARNES, E. R. (1986), A Variation on Karmarkar's Algorithm for Solving Linear Programming Problems, *Mathematical Programming* 36, 174-182.

CARNEIRO, M. S. (1984), Modelo de Otimização para a Operação Hidroelétrica da Cascata de S. Francisco, *Tese de Mestrado*, FEE, UNICAMP.

DECARVALHO, M. (1987), On the Minimization of Work Needed to Factor a Symmetric Positive Definite Matrix, *University of California, Berkeley.*

CARVALHO, M. F. H. (1986), Modelos de Fluxo em Redes Aplicados a Sistemas de Energia Elétrica, *Tese de Doutorado*, FEE, UNICAMP.

CARVALHO, M. F. H.; SOARES, S. & OHISI, T. (1988), Optimal Active Power Dispatch by Network Flow Approach, *IEEE Transactions on Power Systems* 3, 1640-1647.

CHANDRU, V. & KOCHAR, B. S. (1986), A Class of Algorithms for Linear Programming. *Purdue University.*

CORREIA, P. B. (1989), Um Modelo Multisetorial para Otimização do Suprimento de Energia: Eletricidade, Gás Natural e Cogeração com Biomassa, *Tese de Doutorado*, FEE, UNICAMP.

CORREA, P. B. & LYRA, C. (1988), Um Algoritmo em Rede de Fluxo não-conservativa para Otimização do Suprimento de Energia, *Anais do III Congresso Latino-Americano de Automática 1*, 210-215. Viña del Mar, Chile.

CORREIA, P. B.; LYRA, C. & OLIVEIRA, A. R. L. (1988). Uma implementação Computacional de Algoritmo Polinomial de Programação Linear: Aplicação ao Planejamento da Operação de Sistemas Hidrotérmicos, *Anais do 7º Congresso Brasileiro de Automática 2*, 924-929. ITA, S. J. Campos SP.

DIKIN, I. I. (1967), Iterative Solution of Problems of Linear and Quadratic Programming, *Soviet Mathematics Doklady 8*, 674-675.

DONGARRA, J. J. & GROSSE, E. (1987), Distribuition of Mathematical Software via Eletronic Mail, *Communications of the ACM*, 403-414.

DUFF, I. S.; ERISMAN, A. M. & REID, J. K. (1986), Direct Methods for Sparse Matrices, *Clarendon Press*, Oxford.

EISENSTAT, S. C.; SCHULTZ, M. H. & SHERMAN, A. H.(1981), Algorithms and Data Structures for Sparse Symmetric Gaussian Elimination, *SIAM Journal Science & Stat. Comp. 2*, 225-237.

FLETCHER, R. & POWELL, M. J. D. (1974), On the Modification of LDL^t Fatorizations, *Mathematics of Computation 28*, 1067-1087.

GILL, P. E.; MURRAY, W.; SAUNDERS, M. A.; TOMLIN, J. A. & WRIGHT, M. H. (1986), On the Projected Newton Barrier Methods for Linear Programming and an Equivalence to Karmarkar's Projective Method, *Mathematical Programming 36*, 183-209.

GONZAGA, C. C. (1987), Search Directions for Interior Linear Programming Methods, *University of California, Berkeley.*

HOUSOS, E. C. & IRISARRI, G. D. (1982), A Sparse Variable Metric Optimization Method Applied to the Solution of Power System Problems, *IEEE Transactions on Power Apparatus and Systems 101*, 195-202.

IRISARRI, G. D. & SASSON, A. M. (1981), An Automatic Contigency Selection Method for On-Line Security Analysis, *IEEE Transactions on Power Apparatus Systems 100*, 1838-1844.

KACHIYAN, L.G. (1979), A Polynomial Algorithm in Linear Programming, *Soviet Mathematics Doklady 20*, 191-194.

KARMARKAR, N. (1984), A New Polynomial-Time Algorithm for Linear Programming, *Combinatorica 4*, 373-395.

KNUTH, D. E. (1973), The Art of Computer Programming vol. 1 Fundamental Algorithms, *Addison Wesley Publishing Company.*

MONMA, C. L. & MORTON, A. J. (1987), Computational Experience With a Dual Affine Variant of Karmarkar's Method For Linear Programming, *Operations Research Letters 6*, 261-267.

OLIVEIRA, A. R. L. (1989), Métodos de Ponto Interior em Programação Linear - Estudo e Implementação, *Tese de Mestrado, FEE*, UNICAMP.

TINNEY, W. F. & WALKER, J. W. (1967), Direct Solutions of Sparse Network Equations by Optimally Ordered Triangular Factorization, *Proceedings of the IEEE 55*, 1801-1809.

VANDERBEI, R. J. (1989), Karmarkar's Algorithm and Problems with Free Variables, *Mathematical Programming 43*, 31-44.

VANDERBEI, R. J.; MEKETON, M. S. & FREEDMAN, B. A. (1986), A Modification of Karmarkar's Linear Programming Algorithm, *Algorithmica 1*, 395-407.

YANNAKAKIS M. (1981), Computing the Minimum Fill-in is NP-Complete, *SIAM Journal of Algorithms and Discrete Methods 2*, 77-79.