UM MÉTODO ITERATIVO PARALELO PARA PROBLEMAS MINIMAX

José Marcos LOPES[1]

- RESUMO: Apresentamos neste trabalho um novo método iterativo para o problema de estimação na norma ℓ_∞. O algoritmo é a versão paralela de um proposto por Dax, é do tipo relaxação por linha e conveniente quando o sistema de equações lineares a ser resolvido é inconsistente de grande porte, esparso e não possui uma estrutura detectável.

- PALAVRAS-CHAVE: Sistemas Lineares; matriz esparsa; norma mínima; métodos iterativos paralelos; métodos tipo relaxação linha.

1 Introdução

Estamos interessados aqui em obter a solução do seguinte problema minimax

$$\text{minimize} \quad \left\| \mathbf{A}\,\mathbf{x} - \mathbf{b} \right\|_\infty \tag{1.1}$$

onde $\mathbf{A}$ é uma matriz real $m \times n$, $\mathbf{b}=(b_1, \ldots, b_m)^t \in \Re^m$ e $\mathbf{x}=(x_1, \ldots, x_n)^t \in \Re^n$ denota o vetor de incógnitas ; "t" representa o transposto do vetor.

O problema (1.1) é equivalente a um Problema de Programação Linear. Quando a matriz $\mathbf{A}$ é de grande porte, métodos que utilizam modificações de $\mathbf{A}$, como é, por exemplo, o caso do método Simplex, não são convenientes ou mesmo inaplicáveis para a resolução de (1.1).

Existe na literatura um grande número de métodos iterativos para a solução de um sistema de equações lineares. Saad e Vorst (2000), apresentam os principais desenvolvimentos nesta área durante o século XX.

Hadjidimos (1978), desenvolveu o algoritmo seqüencial "Accelerated Overrelaxation Method"- (AOR), o qual pode ser visto como a generalização a dois parâmetros do bem conhecido "Sucessive Over-relaxation" – (SOR). A partir desse trabalho, vários métodos iterativos paralelos, bem como propriedades de convergência foram propostos; como exemplos: Wang (1991), Bai e Su (1977), Bai (1998), Cvetkovic e Obrovski (2000) e Cvetkovic (2002). A convergência desses métodos é obtida quando as matrizes de coeficientes dos sistemas de equações lineares são respectivamente *L*-matrizes, *H*-matrizes e matrizes definida positivas.

Frommer e Mayer (1989), apresentaram um método iterativo paralelo através do "splitting" da matriz $\mathbf{A}$. A convergência do método é obtida se o parâmetro de relaxação pertence ao intervalo $(0,w_0)$ com $w_0 > 1$ e $\mathbf{A}$ é uma *H*-matriz. Outras variantes paralelas

[1]Departamento de Matemática, Faculdade de Engenharia de Ilha Solteira, FEIS/UNESP, CEP: 15385-000, Ilha Solteira, SP, Brasil.

utilizando o "splitting"da matriz **A** são fornecidas em Bai (1995), Bai, Wang e Evans (1995) e Neumann e Plemmons (1987).

Propomos neste trabalho, um método iterativo paralelo do tipo ação por linha, onde as linhas (variáveis) podem ser processadas em paralelo. Os métodos do tipo ação por linha são convenientes quando **A** é de grande porte, esparsa e sem uma estrutura detectável. Tais métodos tem se mostrado eficientes ,por exemplo, para resolver grandes sistemas lineares que ocorrem no campo da reconstrução de imagens por projeção (Censor, 1981), (Censor e Zenios, 1993). Nesse caso não é necessário o armazenamento da matriz *A*, as entradas não nulas da i-ésima linha são geradas de dados experimentais em cada iteração.

Consideremos a seguinte regularização para o problema (1.1)

$$\text{minimize } \frac{1}{2}\,\varepsilon\left(\|\mathbf{x}\|_2^2 + \|\mathbf{A}\mathbf{x}-\mathbf{b}\|_\infty^2\right) + \|\mathbf{A}\mathbf{x}-\mathbf{b}\|_\infty \tag{1.2}$$

onde ε é um número real positivo. O interesse no estudo do problema (1.2) é que através da solução deste problema, podemos obter uma aproximação da solução do problema (1.1).

Segundo Dax (1991), o dual de (1.2) é definido por

$$\text{minimize } \frac{1}{2}\left\|\mathbf{A}^t\mathbf{y}\right\|_2^2 - \varepsilon\,\mathbf{b}^t\mathbf{y} + \frac{1}{2}\left(\max\{0\,,\|\mathbf{y}\|_1 - 1\}\right)^2 \tag{1.3}$$

e se $\hat{\mathbf{y}}=(y_1, y_2, \ldots, y_m)^t \in \Re^m$ é solução de (1.3) então $\hat{\mathbf{x}}=\left(\mathbf{A}^t\hat{\mathbf{y}}-\mathbf{c}\right)/\varepsilon$ é solução de (1.2).

A idéia da regularização foi introduzida como uma forma de melhorar a estabilidade de problemas mal condicionados (Tikhnov e Arsenin, 1977). No caso aqui considerado, o objetivo da regularização é obter um problema dual mais simples.

Uma regularização natural para o problema (1.1) seria

$$\text{minimize } \frac{1}{2}\,\varepsilon\|\mathbf{x}\|_2^2 + \|\mathbf{A}\mathbf{x}-\mathbf{b}\|_\infty \tag{1.4}$$

entretanto, segundo Dax (1991), o dual de (1.4) tem a forma

$$\begin{aligned} &\text{minimize } \frac{1}{2}\left\|\mathbf{A}^t\mathbf{y}\right\|_2^2 - \varepsilon\mathbf{b}^t\mathbf{y} \\ &\text{sujeito a } \|\mathbf{y}\|_1 \le 1 \end{aligned} \tag{1.5}$$

as restrições de (1.5) introduzem uma certa dificuldade na implementação de métodos do tipo ação por linha.

Em Lopes e De Pierro (1992) é apresentado um algoritmo paralelo para estimação de mínimo valor absoluto, em que a regularização utilizada é como em (1.4). Mangassariam (1981), foi o primeiro a utilizar esse tipo de regularização para propor um algoritmo iterativo em Programação Linear.

Da notação utilizada ,

$$\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_\infty = \max_{1\le i\le m} \left|\mathbf{a}_i^t\mathbf{x} - b_i\right|$$

onde $\mathbf{a}_i^t$ denota a i-ésima linha de **A**, e

$$\| \mathbf{x} \|_p = \left(\sum_{i=1}^{n} | \mathbf{x}_i |^p \right)^{1/p}, \quad \text{para} \quad 1 \leq p < \infty \quad \text{e qualquer} \quad \mathbf{x} \in \Re^n.$$

Para quaisquer $\mathbf{x}, \mathbf{y} \in \Re^n$, o produto escalar euclidiano será denotado por $\mathbf{x}^t\mathbf{y}$ ou $\langle \mathbf{x}, \mathbf{y} \rangle$.

O artigo está organizado da seguinte forma: na seção 2 apresentamos uma conveniente regularização para o problema minimax; na seção 3 o novo algoritmo é definido e provas de convergência são fornecidas. Na seção 4 apresentamos os resultados de algumas experiências computacionais e a seção 5 refere-se às conclusões finais.

2 Uma regularização para o problema minimax

Consideremos o problema minimax (1.1) regularizado

$$\text{minimize } \frac{1}{2}\varepsilon\left(\| \mathbf{x} \|_2^2 + \| \mathbf{A}\mathbf{x} - \mathbf{b} \|_\infty^2 \right) + \| \mathbf{A}\mathbf{x} - \mathbf{b} \|_\infty . \tag{2.1}$$

Apresentamos, a seguir, o dual do problema (2.1) e relacionamos as soluções primal-dual. Todos os resultados desta seção são devidos a Dax (1991).

Mangassarian (1981) estudou o Problema de Programação Linear (PPL) regularizado

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}\varepsilon \| \mathbf{x} \|_2^2 + \mathbf{c}^t \mathbf{x} \\ &\text{sujeito a} \quad \mathbf{A}\mathbf{x} \geq \mathbf{b} \end{aligned} \tag{2.2}$$

como uma forma de obter uma solução aproximada para o PPL,

$$\begin{aligned} &\text{minimize } \mathbf{c}^t \mathbf{x} \\ &\text{sujeito a} \quad \mathbf{A}\mathbf{x} \geq \mathbf{b} \end{aligned} \tag{2.3}$$

onde ε, $\mathbf{A}$, $\mathbf{b}$ e $\mathbf{x}$ são como definidos em (1.2) e $\mathbf{c} \in \Re^n$ é o vetor da função objetivo. Mostrou-se, neste caso, que se (2.3) tem solução e $\hat{\mathbf{y}} \in \Re^m$ é a solução do problema

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2} \left\| \mathbf{A}^t\mathbf{y} - \mathbf{c} \right\|_2^2 - \varepsilon \mathbf{b}^t\mathbf{y} \\ &\text{sujeito a} \quad \mathbf{y} \geq 0 \end{aligned} \tag{2.4}$$

então o vetor $\hat{\mathbf{x}} = \left(\mathbf{A}^t\hat{\mathbf{y}} - \mathbf{c} \right) / \varepsilon$ é a solução única de (2.2).

Agora, o problema (1.1) é equivalente ao seguinte PPL (Sposito, 1975)

$$\text{minimize } (1,0, \ldots ,0) \begin{pmatrix} \tau \\ \mathbf{x} \end{pmatrix} \tag{2.5}$$

$$\text{sujeito a } \begin{bmatrix} \mathbf{e} & \mathbf{A} \\ \mathbf{e} & -\mathbf{A} \end{bmatrix} \begin{bmatrix} \tau \\ \mathbf{x} \end{bmatrix} \geq \begin{bmatrix} \mathbf{b} \\ -\mathbf{b} \end{bmatrix}$$

onde $\mathbf{e} = (1, 1, \ldots , 1)^t \in \Re^m$. Assim, utilizando as idéias de Mangassariam (1981), Dax (1991) mostrou que o dual do problema (2.1) tem a forma

$$\text{minimize } F(\mathbf{y}) = \frac{1}{2}\left\| \mathbf{A}^t\mathbf{y} \right\|_2^2 - \varepsilon \mathbf{b}^t\mathbf{y} + \frac{1}{2}\left(\max\left\{0, \sum_{i=1}^{m} |\, y_i \,| - 1\right\}\right)^2 . \tag{2.6}$$

A equivalência entre os problemas (2.1) e (2.6) é estabelecida pelo teorema (2.1).

Teorema 2.1: *O problema (2.6) sempre tem uma solução. Se* $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_m)^t \in \Re^m$ *resolve este problema então o vetor*

$$\hat{\mathbf{x}} = \mathbf{A}^t \hat{\mathbf{y}} / \varepsilon \tag{2.7}$$

é a solução única de (2.1) e

$$\left\| \mathbf{A}\hat{\mathbf{x}} - \mathbf{b} \right\|_\infty = \max\left\{ 0 , \sum_{i=1}^{m} |\, \hat{y}_i \,| - 1 \right\} \Big/ \varepsilon . \tag{2.8}$$

Observações:

2.1. A existência de solução para o problema (2.6) segue diretamente da convexidade da função objetivo $F(\mathbf{y})$.

2.2. Se o sistema linear **Ax = b** é inconsistente então de (2.8) qualquer solução $\hat{\mathbf{y}}$ de (2.6) satisfaz

$$\left\| \hat{\mathbf{y}} \right\|_1 > 1 .$$

Portanto, qualquer ponto de mínimo da função

$$G(\mathbf{y}) = \frac{1}{2}\left\| \mathbf{A}^t\mathbf{y} \right\|_2^2 - \varepsilon \mathbf{b}^t\mathbf{y} + \frac{1}{2}\left(\left\| \mathbf{y} \right\|_1 - 1\right)^2$$

é também um ponto de mínimo de $F(\mathbf{y})$ e vice-versa.

A parte final desta seção é destinada a apresentação de algumas propriedades da função $F(\mathbf{y})$.

Dado $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_m)^t \in \Re^m$, e considerando $F(\mathbf{y})$ como em (2.6) defina a função de uma variável

$$\hat{f}_i(\theta) = F(\hat{\mathbf{y}} + \theta e_i), \qquad i = 1, 2, \ldots, m \tag{2.9}$$

onde $\mathbf{e}_i$ denota a i-ésima coluna da matriz identidade de ordem m x m.

A função $\hat{f}_i(\theta)$ é estritamente convexa e contínua. O teorema (2.2) estabelece a relação entre o problema (2.6) e o ponto de mínimo de $\hat{f}_i(\theta)$.

Teorema 2.2: *O vetor* $\hat{\mathbf{y}} \in \Re^m$ *resolve o problema (2.6) se e somente se para cada i=1, 2, ... , m, o ponto de mínimo de* $\hat{f}_i(\theta)$ *ocorre para* $\theta = 0$.

O lema seguinte estabelece um limitante inferior para a função $F(\mathbf{y})$.

Lema 2.1: *Seja F(* $\mathbf{y}$ *) como definido em (2.6), então*

$$\| \mathbf{y} \|_1 \leq \frac{F(\mathbf{y})}{\varepsilon} + \gamma + \frac{1}{2}\varepsilon^2$$

onde

$$\gamma = 1 + \max_{i=1,\, m} | b_i | . \tag{2.10}$$

3 Definição do algoritmo

Dax (1991) propôs o seguinte método iterativo seqüencial para calcular o ponto de mínimo da função

$$F(\mathbf{y}) = \frac{1}{2} \| \mathbf{A}^t \mathbf{y} \|_2^2 - \varepsilon \mathbf{b}^t \mathbf{y} + \frac{1}{2} \left(\max \left\{ 0 , \sum_{i=1}^{m} | \mathbf{y}_i | - 1 \right\} \right)^2 . \tag{3.1}$$

Cada iteração do algoritmo é composta de m passos onde, no i-ésimo passo, $i = 1, 2, \ldots, m,$ considera-se a *i*-ésima linha de **A**, a qual é denotada por $\mathbf{a}_i^t$.

Seja $\mathbf{y} = (y_1, \ldots, y_m)^t$ a estimativa atual da solução para o início do *i*-ésimo passo e defina

$$\mathbf{r} = \mathbf{A}^t \mathbf{y} \tag{3.2}$$

e

$$\rho = \sum_{i=1}^{m} | y_i | - 1 . \tag{3.3}$$

Para cada passo *i*, o valor y_i é substituído por $y_i + \theta^*$ onde θ^* minimiza a função de uma variável

$$f_i(\theta) = F(\mathbf{y} + \theta \mathbf{e}_i) = \frac{1}{2} \| \theta \mathbf{a}_i + \mathbf{r} \|_2^2 - \theta \varepsilon b_i - \varepsilon \mathbf{b}^t \mathbf{y} + \frac{1}{2} \left(\max \left\{ 0 , | y_i + \theta | + \rho - | y_i | \right\} \right)^2 . \tag{3.4}$$

O i-ésimo passo do algoritmo é implementado por:

a) Faça $\theta = - \left(\mathbf{a}_i^t \mathbf{r} - \varepsilon b_i \right) / \mathbf{a}_i^t \mathbf{a}_i$, $\eta = y_i + \theta$ e $\rho := \rho - | y_i |$.

b) Se $\rho + |\eta| \leq 0$ vá para (d).

c) Se $\eta > 0$ faça $\theta := \theta + \max \left\{ -\eta , -(\eta + \rho) / \left(\mathbf{a}_i^t \mathbf{a}_i + 1 \right) \right\}$.

Se $\eta < 0$ faça $\theta := \theta + \min \left\{ -\eta , -(\eta - \rho) / \left(\mathbf{a}_i^t \mathbf{a}_i + 1 \right) \right\}$.

d) Faça $y_i := y_i + \theta$, $\mathbf{r} := \mathbf{r} + \theta \mathbf{a}_i$ e $\rho := \rho + | y_i |$.

O símbolo := denota atribuição aritmética.

Apresentamos agora a definição do novo algoritmo, que é a versão paralela do algoritmo descrito anteriormente e será denominado algoritmo MINIMAXPAR.

Inicialização: *Sejam* $\mathbf{y}^0 \in \Re^m$ *a aproximação inicial,* ε *um número real positivo e* $\lambda_1, \lambda_2, \ldots, \lambda_m$ *números reais positivos tais que*

$$\sum_{i=1}^{m} \lambda_i = 1. \tag{3.5}$$

Faça

$$\mathbf{r}^0 = \mathbf{A}^t \mathbf{y}^0 \tag{3.6}$$

e

$$\rho^0 = \sum_{i=1}^{m} \left| \mathbf{y}_i^0 \right| - 1 \ . \tag{3.7}$$

Iteração Principal: Para $k = 0, 1, 2, \ldots$ faça

$$\mathbf{y}^{k+1} = \mathbf{y}^k + \Lambda \delta^k \tag{3.8}$$

$$\mathbf{r}^{k+1} = \mathbf{r}^k + \sum_{i=1}^{m} \lambda_i \, \delta_i^k \, \mathbf{a}_i \tag{3.9}$$

onde Λ é uma matriz diagonal com elementos diagonais λ_i' s e δ^k é um m-vetor com componentes δ_i^k definidas por: para $i = 1, 2, \ldots, m$ sejam

$$\theta_i^k = -\left(\langle \mathbf{a}_i \, , \mathbf{r}^k \rangle - \varepsilon b_i \right) \Big/ \langle \mathbf{a}_i \, , \mathbf{a}_i \rangle \tag{3.10}$$

$$\eta_i = y_i^k + \theta_i^k \tag{3.11}$$

$$\rho_i = \rho^k - \left| y_i^k \right|. \tag{3.12}$$

Se $\rho_i + \left| \eta_i \right| \leq 0$, faça $\delta_i^k = \theta_i^k$.

Se $\eta_i > 0$, faça

$$\delta_i^k = \max \left\{ \frac{\theta_i^k - \eta_i}{\lambda_i}, \theta_i^k - (\eta_i + \rho_i) \Big/ (\langle \mathbf{a}_i \, , \mathbf{a}_i \rangle + 1) \right\}. \tag{3.13}$$

Se $\eta_i < 0$, faça

$$\delta_i^k = \min \left\{ \frac{\theta_i^k - \eta_i}{\lambda_i} \, , \ \theta_i^k - (\eta_i - \rho_i) \Big/ (\langle \mathbf{a}_i \, , \mathbf{a}_i \rangle + 1) \right\}. \tag{3.14}$$

Para cada iteração k, no cálculo de δ_i^k, usamos apenas a i-ésima linha da matriz $\boldsymbol{A}$. Assim todos os δ_i^k , $i = 1, 2, \ldots, m$ podem ser calculados simultaneamente caso se utilize um computador com processamento paralelo.

No que se segue, $\left\{\mathbf{y}^k\right\}$ denota uma seqüência gerada pelo algoritmo MINIMAXPAR.

Lema 3.1: *Para todo k, k* = 0, 1, 2, ...

$$\mathbf{r}^k = \mathbf{A}^t\,\mathbf{y}^k\,.$$

Demonstração: Por indução finita sobre *k.* Para $k = 0$, o resultado segue de (3.6). Vamos supor que o resultado seja válido para *k*-1 e provar que o mesmo é verdadeiro para k. De (3.9) vem,

$$\begin{aligned}\mathbf{r}^k &= \mathbf{r}^{k\text{-}1} + \sum_{i=1}^{m}\lambda_i\,\delta_i^{k\text{-}1}\,\mathbf{a}_i \\ &= \mathbf{r}^{k\text{-}1} + \sum_{i=1}^{m}\left(y_i^k - y_i^{k\text{-}1}\right)\mathbf{a}_i \\ &= \mathbf{A}^t\,\mathbf{y}^{k\text{-}1} + \mathbf{A}^t\,\mathbf{y}^k - \mathbf{A}^t\,\mathbf{y}^{k\text{-}1} = \mathbf{A}^t\,\mathbf{y}^k\end{aligned}$$

onde a segunda igualdade ocorre de (3.8) e a terceira da hipótese de indução.

Observações:

3.1. O valor θ_i^k como definido em (3.10) é o ponto de mínimo da função de uma variável

$$\begin{aligned}h(\theta) &= \frac{1}{2}\left\|\mathbf{A}^t\left(\mathbf{y}^k + \theta\,\mathbf{e}_i\right)\right\|_2^2 - \varepsilon\mathbf{b}^t\left(\mathbf{y}^k + \theta\,\mathbf{e}_i\right) \\ &= \frac{1}{2}\left\|\theta\mathbf{a}_i + \mathbf{r}^k\right\|_2^2 - \theta\varepsilon\,b_i - \varepsilon\mathbf{b}^t\,\mathbf{y}^k\end{aligned}$$

onde a segunda igualdade segue do lema (3.1) e e_i denota a *i*-ésima coluna da matriz identidade de ordem *m* x *m.*

As expressões (3.13) e (3.14), fazem a correção quando necessário, para que este ponto seja o mínimo da função $f_i(\theta)$ definida por:

$$f_i(\theta) = \frac{1}{2}\left\|\theta\,a_i + \mathbf{r}^k\right\|_2^2 - \theta\,\varepsilon\,b_i - \varepsilon\,\mathbf{b}^t\mathbf{y}^k + \frac{1}{2}\left(\max\left\{0\,,\left|\,y_i^k + \theta\right| + \rho_i - \left|\,y_i^k\,\right|\right\}\right)^2. \tag{3.15}$$

3.2. O valor θ_i^k como definido em (3.10), pode também ser visto como sendo a variação de y_i no *i*-ésimo passo da iteração $k + 1$ do método de Jacobi para o sistema de equações lineares

$$\mathbf{A}\mathbf{A}^t\mathbf{y} = \varepsilon\mathbf{b}\,. \tag{3.16}$$

De fato, o esquema iterativo de Jacobi para (3.16) é definido para cada i por:

$$y_i^{k+1} = \frac{1}{\langle\,\mathbf{a}_i\,,\,\mathbf{a}_i\,\rangle}\left[\varepsilon\,b_i - \sum_{\substack{j=1\\ j\neq i}}^{m}\langle\,\mathbf{a}_i\,,\,\mathbf{a}_j\,\rangle\;y_j^k\right]$$

$$= \frac{1}{\langle \mathbf{a}_i , \mathbf{a}_i \rangle}\left[\varepsilon\, b_i \; - \langle \mathbf{a}_i , \mathbf{A}^t \mathbf{y}^k \rangle + \langle \mathbf{a}_i , \mathbf{a}_i \rangle y_i^k\right]. \qquad (3.17)$$

Assim, do lema (3.1), de (3.10) e (3.17) temos que

$$y_i^{k+1} - y_i^k = \theta_i^k .$$

Logo, o algoritmo MINIMAXPAR pode ser visto como do tipo Jacobi.

3.3. De (3.5) e (3.8) temos que para qualquer k = 0, 1, 2, ... , o iterado y^{k+1} pode ser escrito como:

$$\begin{aligned} \mathbf{y}^{k+1} &= \left(y_1^k + \lambda_1\, \delta_1^k \,, y_2^k + \lambda_2\, \delta_2^k \,, \ldots, y_m^k + \lambda_m\, \delta_m^k\right) \\ &= \sum_{i=1}^{m} \lambda_i\, \mathbf{y}^k + \left(\lambda_1\, \delta_1^k , \lambda_2\, \delta_2^k , \ldots, \lambda_m\, \delta_m^k\right) \\ &= \lambda_1\left(\mathbf{y}^k + \delta_1^k\, \mathbf{e}_1\right) + \lambda_2\left(\mathbf{y}^k + \delta_2^k\, \mathbf{e}_2\right) + \ldots + \lambda_m\left(\mathbf{y}^k + \delta_m^k\, \mathbf{e}_m\right). \qquad (3.18) \end{aligned}$$

Assim, y^{k+1} é definido como sendo a combinação convexa dos vetores $\mathbf{y}^k + \delta_i^k\, \mathbf{e}_i$, i=1, 2, ... m.

Agora, da maneira como o algoritmo MINIMAXPAR foi definido, para qualquer i=1, 2, ..., m temos que

$$F\left(\mathbf{y}^k + \delta_i^k\, \mathbf{e}_i\right) \le F\left(\mathbf{y}^k\right). \qquad (3.19)$$

O lema a seguir, mostra que a função objetivo $F(\mathbf{y})$ definida em (3.1) é decrescente sobre a seqüência de iterados $\left\{\mathbf{y}^k\right\}$ e converge.

Lema 3.2: (i) $F\left(\mathbf{y}^{k+1}\right) < F\left(\mathbf{y}^k\right)$ *para todo* k , $k = 0, 1, 2, \ldots$

(ii) $\lim_{k\to\infty} \left(F\left(\mathbf{y}^{k+1}\right) - F\left(\mathbf{y}^k\right)\right) = 0$.

Demonstração:

De (3.18) temos que

$$F\left(\mathbf{y}^{k+1}\right) = F\left(\sum_{i=1}^{m} \lambda_i \left(\mathbf{y}^k + \delta_i^k\, \mathbf{e}_i\right)\right) < \sum_{i=1}^{m} \lambda_i\, F\left(\mathbf{y}^k + \delta_i^k\, \mathbf{e}_i\right) \le \sum_{i=1}^{m} \lambda_i\, F\left(\mathbf{y}^k\right) = F\left(\mathbf{y}^k\right)$$

onde a primeira desigualdade ocorre da convexidade estrita de $F(\cdot)$, a segunda de (3.19) e a última igualdade de (3.5).

(ii) Do item (i), $\left\{F\left(\mathbf{y}^k\right)\right\}$ é monotônica decrescente, do teorema (2.1) $F(\mathbf{y})$ possui um ponto de mínimo, ou seja, é limitada inferiormente, logo a seqüência $\left\{F\left(\mathbf{y}^k\right)\right\}$ converge e temos o resultado desejado.

A função de uma variável definida a seguir será utilizada na demonstração da convergência do algoritmo MINIMAXPAR.

$$\psi(t) = F\left(\mathbf{y}^k + t\left(\mathbf{y}^{k+1} - \mathbf{y}^k\right)\right) \tag{3.20}$$

onde $\mathbf{y}^k$ e $\mathbf{y}^{k+1}$ são iterados consecutivos gerados pelo algoritmo MINIMAXPAR e F(.) é como definido em (3.1).

Sejam $t_1 = 0$ e $t_2 = 1$, então de (3.20) temos que $\psi(t_1) = F\left(\mathbf{y}^k\right)$ e $\psi(t_2) = F\left(\mathbf{y}^{k+1}\right)$. Agora, do item (i) do lema (3.2), segue que

$$\psi(t_1) > \psi(t_2). \tag{3.21}$$

A função $\psi(t)$ é duas vezes continuamente diferenciável e estritamente convexa (de fato, um "spline" quadrático), então como $\psi(t_1) > \psi(t_2)$ segue que

$$\psi'(t_2) \leq 0. \tag{3.22}$$

Seja $H(\mathbf{y})$ a matriz Hessiana de $F(\mathbf{y})$ para o ponto $\mathbf{y}$. Como $F(\mathbf{y})$ é duas vezes diferenciável, estritamente convexa e quadrática então $H(\mathbf{y})$ é definida positiva.

O lema abaixo estabelece a limitação da seqüência $\left\{\mathbf{y}^k\right\}$.

Lema 3.3: *A seqüência* $\left\{\mathbf{y}^k\right\}$ *é limitada.*

Demonstração: Do item (*i*) do lema (3.2) temos que para qualquer k , $k = 1, 2, 3, ...$

$$F\left(\mathbf{y}^k\right) < F\left(\mathbf{y}^{k-1}\right) < ... < F\left(\mathbf{y}^1\right) < F\left(\mathbf{y}^0\right). \tag{3.23}$$

Agora do lema (2.4) temos que

$$\left\| \mathbf{y}^k \right\|_1 \leq \frac{F\left(\mathbf{y}^k\right)}{\varepsilon} + \gamma + \frac{1}{2}\,\varepsilon\gamma \tag{3.24}$$

onde γ é dado por (2.10).

Assim, de (3.23) e (3.24) segue que

$$\left\| \mathbf{y}^k \right\|_1 \leq \frac{F\left(\mathbf{y}^0\right)}{\varepsilon} + \gamma + \frac{1}{2}\,\varepsilon\gamma \tag{3.25}$$

e temos o resultado desejado.

O lema a seguir é devido a Dax (1985) e será utilizado para a demonstração da convergência do algoritmo MINIMAXPAR.

Lema 3.4: *Seja* $\psi(t)$ *uma função duas vezes continuamente diferenciável e sejam dois pontos* $t_1 < t_2$ *satisfazendo as seguintes condições:*

1) $\psi(t_1) > \psi(t_2)$;

2) $\psi'(t_2) \leq 0$ e

3) $\psi''(t) \geq a$ *para todo* $t_1 \leq t \leq t_2$ *onde* a *é uma constante positiva. Então*

$$t_2 - t_1 \leq 2\left(\frac{\psi(t_1)-\psi(t_2)}{a}\right)^{1/2}. \tag{3.26}$$

Teorema 3.1: *Seja* $L = \left\{\mathbf{y} \,\middle|\, F(\mathbf{y}) \leq F(\mathbf{y}^0)\ ,\ \textit{onde}\ \ \|\mathbf{y}\|_2 \leq c,\ c: \text{constante}\right\}$ *um conjunto de nivel. Cada ponto de acumulação da seqüência* $\left\{\mathbf{y}^k\right\}$ *é um ponto de mínimo de* $F(\mathbf{y})$ *sobre L.*

Demonstração: Fazendo $c = \dfrac{F\left(\mathbf{y}^0\right)}{\varepsilon} + \gamma + \dfrac{1}{2}\,\varepsilon\gamma$ como em (3.25), então de (3.23) e do lema (3.3) temos que para qualquer k, k = 1,2, ..., o ponto $\mathbf{y}^k$ gerado pelo algoritmo MINIMAX é tal que $\mathbf{y}^k \in$ L.

Desde que *L* é um conjunto compacto e $H(\mathbf{y})$ é definida positiva, então existe uma constante $a > 0$ tal que $H(\mathbf{y}) \geq a\mathbf{I}$ para todo $y \in L$ ($\mathbf{I}$: matriz identidade).

Seja $\mathbf{u} = \mathbf{y}^{k+1} - \mathbf{y}^k$ e de (3.20), $\psi(t) = F\left(\mathbf{y}^k + t\mathbf{u}\right)$. Assim, assumindo-se $\mathbf{u} \neq \mathbf{0}$, temos que

$$\psi''(t) \geq a\,\|\mathbf{u}\|_2^2. \tag{3.27}$$

Logo, considerando $t_1 = 0$, $t_2 = 1$ e de (3.21), (3.22), (3.27) e do lema (3.3) segue que

$$1 \leq 2\left(\frac{F\left(\mathbf{y}^k\right) - F\left(\mathbf{y}^{k+1}\right)}{a\,\|\mathbf{u}\|_2^2}\right)^{1/2}$$

ou

$$\left\|\mathbf{y}^{k+1} - \mathbf{y}^k\right\|_2 \leq 2\left(\frac{F\left(\mathbf{y}^k\right) - F\left(\mathbf{y}^{k+1}\right)}{a}\right)^{1/2}. \tag{3.28}$$

Tomando o limite quando $k \to \infty$ na expressão (3.28) e do item (ii) do lema (3.2) obtemos

$$\lim_{k\to\infty} \left\|\mathbf{y}^{k+1} - \mathbf{y}^k\right\|_2 = 0. \tag{3.29}$$

Do lema (3.3), a seqüência $\left\{\mathbf{y}^k\right\}$ é limitada, e isto assegura a existência de pelo menos um ponto de acumulação. Seja $\hat{\mathbf{y}}$ um ponto de acumulação de $\left\{\mathbf{y}^k\right\}$, então existe uma subseqüência $\left\{\mathbf{y}^{i_k}\right\}$ de $\left\{\mathbf{y}^k\right\}$ tal que

$$\lim_{k\to\infty} \mathbf{y}^{i_k} = \hat{\mathbf{y}}\ . \tag{3.30}$$

Para concluirmos a prova, devemos mostrar que para cada *i*, i = 1,2, ... *m,* o ponto de mínimo da função de uma variável

$$\hat{f}_i(\theta) = F(\hat{\mathbf{y}} + \boldsymbol{\theta} + \mathbf{e}_i) \tag{3.31}$$

ocorre para $\theta = 0$.

De (3.30), (3.31) e da continuidade da função *F* vem,

$$\begin{aligned} \lim_{k\to\infty} F\left(\mathbf{y}^{i_k} + \theta\, \boldsymbol{e}_i\right) &= F\left(\lim_{k\to\infty} \mathbf{y}^{i_k} + \theta\, \boldsymbol{e}_i\right) \\ &= F\left(\hat{\mathbf{y}} + \theta\, \boldsymbol{e}_i\right) \\ &= \hat{f}_i(\theta). \end{aligned}$$

Ou seja, quando y^{i_k} aproxima-se de $\hat{y}$, o ponto de mínimo da função de uma variável

$$f_i(\theta) = F\left(\mathbf{y}^{i_k} + \theta \mathbf{e}_i\right)$$

aproxima-se daquele de $\hat{f}_i(\theta)$. Portanto, se para algum índice *i*, zero não é um ponto de mínimo de $\hat{f}_i(\theta)$ então o limite definido em (3.29) seria violado.

Teorema 3.2: *A seqüência* $\left\{\mathbf{A}^t\, \mathbf{y}^k / \varepsilon\right\}$ *converge para um ponto* $\hat{\mathbf{x}}$ *tal que* $\hat{\mathbf{x}}$ *é a solução única do problema (2.1).*

Demonstração: A seqüência $\left\{\mathbf{A}^t\, \mathbf{y}^k / \varepsilon\right\}$ é limitada desde que $\left\{\mathbf{y}^k\right\}$ é limitada. Seja $\hat{\mathbf{x}}$ um ponto de acumulação de $\left\{\mathbf{A}^t\, \mathbf{y}^k / \varepsilon\right\}$ e seja $\left\{\mathbf{y}^{i_k}\right\}$ uma subseqüência de $\left\{\mathbf{y}^k\right\}$ tal que

$$\lim_{k\to\infty} \mathbf{A}^t\, \mathbf{y}^{i_k} / \varepsilon = \hat{\mathbf{x}}. \tag{3.32}$$

Agora, $\left\{\mathbf{y}^{i_k}\right\}$ possui um ponto de acumulação $\hat{\mathbf{y}}$, pois a mesma é limitada. Assim existe uma subseqüência $\left\{\mathbf{y}^{j_k}\right\}$ de $\left\{\mathbf{y}^{i_k}\right\}$ tal que

$$\lim_{k\to\infty} \mathbf{y}^{j_k} = \hat{\mathbf{y}} \tag{3.33}$$

e de (3.32) vem

$$\lim_{k\to\infty} A^t\, y^{j_k} / \varepsilon = \hat{x}. \tag{3.34}$$

Assim, de (3.33) e (3.34) temos que

$$\hat{\mathbf{x}} = \mathbf{A}^t\, \hat{\mathbf{y}} / \varepsilon \tag{3.35}$$

onde $\hat{\mathbf{y}}$ é um ponto de acumulação de $\left\{\mathbf{y}^k\right\}$.

Portanto, pelo teorema (3.1) $\hat{\mathbf{y}}$ é um ponto de mínimo de $F(\mathbf{y})$ e pelo teorema (2.1), temos que $\hat{\mathbf{x}}$ é a solução única do problema (2.1).

4 Resultados numéricos

Apresentamos a seguir, algumas experiências computacionais para o algoritmo paralelo MINIMAXPAR e o algoritmo seqüencial de Dax. Os algoritmos foram implementados em FORTRAN visual Workbench da Microsoft e os testes foram realizados em um microcomputador pessoal.

Para os dois algoritmos e em todos os exemplos utilizamos a solução inicial $\mathbf{y}^0 = (0, 0, \cdots, 0) \in \Re^m$. Para o algoritmo MINIMAXPAR usamos em todos os casos, $\lambda_i = 1/m$ para $i = 1, 2, \cdots, m$, onde m denota o número de linhas da matriz **A**. Uma escolha mais refinada para os pesos λ_i poderá produzir melhores resultados de convergência para o algoritmo MINIMAXPAR.

O critério de parada considerado foi

$$\left\| \mathbf{y}^{k+1} - \mathbf{y}^k \right\|_\infty \leq TOL$$

ou o número de iterações excede 2.000 (TOL é um número real positivo e pequeno pré-fixado). Os símbolos "*" nas Tabelas 1,2,6,7,8 e 9, indicam que o algoritmo não convergiu após 2.000 iterações.

O valor ótimo é indicado por *VO* e calculado como

$$VO = \left\| \mathbf{A}\mathbf{x}^* - \mathbf{b} \right\|_\infty$$

onde $\mathbf{x}^* = \mathbf{A}^t \mathbf{y}^* / \varepsilon$ e $\mathbf{y}^*$ é o valor fornecido pelo algoritmo.

Para os testes utilizamos diferentes valores de ε e a comparação entre os algoritmos foi feita através do número de iterações necessárias, o qual está indicado por ITER, pelo valor da função objetivo *VO* e pelo tempo de CPU, o qual é indicado por TEMPO e representado como

$$min: sec.\ 10^{-2}\ sec.$$

Para problemas em que a dimensão da matriz **A** é pequena, não foi possível medir o tempo de CPU, pois a menor fração de tempo medida é de centésimos de segundo. Os casos mais interessantes são aqueles onde **A** é de médio para grande porte, e assim o tempo de CPU pode ser medido sem qualquer problema.

Consideramos dois tipos de problemas para os testes numéricos. No primeiro caso, **A** é uma matriz de pequeno a médio porte e a solução do sistema linear **Ax** = **b** é conhecida e está indicada por x^* (Exemplos 1, 2, 3 e 4). Utilizamos estes exemplos, como uma forma de verificar a exatidão da solução fornecida pelos algoritmos. No segundo caso, consideramos problemas gerados aleatoriamente. Para o exemplo 6, a seguir, a matriz **A** é esparsa, sem uma estrutura detectável e com $m >> n$, ou seja, o sistema linear **Ax** = **b** é inconsistente. Assim, este é o principal exemplo desta seção, tendo em vista que os algoritmos apresentados são do tipo ação por linha e convenientes para este tipo de sistema linear.

Exemplo 1: (Cheney, 1982, p.44)

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ 1 & 2 \\ 2 & 4 \\ 2 & 1 \\ 3 & 1 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 3 \\ 1 \\ 7 \\ 11{,}1 \\ 6{,}9 \\ 7{,}2 \end{bmatrix} \qquad onde \quad \mathbf{x}^* = (2\ ,\ 2)^t .$$

Exemplo 2:

$$\mathbf{A} = \begin{bmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

onde

$$\mathbf{x}^* = \left(\frac{36}{99}, \frac{45}{99}, \frac{45}{99}, \frac{36}{99} \right)^t .$$

Exemplo 3: (Murty, 1988, p.19)
A é uma matriz n x n, onde

$$a_{ij} = \begin{cases} 1 & \text{se } i = j \\ 0 & \text{se } i < j \\ 2 & \text{se } i > j \end{cases} \qquad e \qquad b_i = \sum_{j=1}^{n} a_{ij}$$

para

$$i = 1, 2, \cdots, m .$$

Assim **A** é uma matriz semidefinida positiva, com det **A** = 1, e o sistema linear **Ax** = **b** possui a solução única $\mathbf{x}^* = (1, 1, \cdots, 1)^t$.

Exemplo 4: (Ruggiero e Lopes, 1988, p.94)

$$A = \begin{bmatrix} -\alpha & 0 & 0 & 1 & \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\alpha & 0 & -1 & 0 & -\alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\alpha & -1 & 0 & 0 & \alpha & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \alpha & 0 & 1 & 0 & \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -\alpha & 0 & 0 & 1 & \alpha & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\alpha & 0 & -1 & 0 & -\alpha & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & \alpha \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -\alpha \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\alpha & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\alpha & -1 \end{bmatrix}$$

$$b = \begin{bmatrix} 0 & 0 & 0 & 10 & 0 & 0 & 0 & 15 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 0 \end{bmatrix}^t$$

com $\alpha = \text{sen } 45^\circ = \sqrt{2}/2 \cong 0{,}7071$.

Seguindo Dax e Berkowitz (1990) e Bartels *et al*. (1989) usamos, também para os testes numéricos, os seguintes sistemas lineares gerados aleatoriamente.

Exemplo 5: **A** é uma matriz m x n e esparsa, onde cada linha $\mathbf{a}_i^t$ de **A** possui η_i elementos não nulos, η_i é um número aleatório do conjunto {1, 2, 3, 4, 5}, a localização destes elementos não nulos de $\mathbf{a}_i^t$ são números aleatórios do conjunto $\{1, 2, \cdots, n\}$ (**A** possui n colunas) e a_{ij} e b_i são números aleatórios do intervalo [-1 , 1].

Todos os números aleatórios foram gerados através da rotina RANDOM (ranval), onde ranval é o número aleatório fornecido e pertencente ao intervalo [0 , 1). A rotina SEED (seedval) é utilizada para variar o ponto inicial dos números pseudo-aleatórios gerados. Se seedval = -1 , então a seqüência de valores de RANDOM será sempre diferente.

Para este exemplo, o algoritmo de Dax não convergiu após 2.000 iterações com tolerância TOL = 10^{-4} e $\varepsilon = 1,\ 0{,}1,\ 0{,}01$ e $0{,}001$. Se **A** é de ordem 1.000 x 100, o tempo gasto pelo método de Dax nas 2.000 iterações foi de aproximadamente 6,5 minutos.

Os resultados fornecidos pelo algoritmo MINIMAXPAR estão dispostos na Tabela 10.

Exemplo 6: A matriz $\mathbf{A} = 1/n\ \mathbf{G}.\mathbf{G}^t$ onde **G** é uma matriz quadrada de ordem n com g_{ij} , $i,j = 1,\ 2,\ \cdots,\ n$, gerado aleatoriamente tal que $g_{ij} \in [\ 0\ ,\ 1\ )$. Assim **A** é uma matriz quadrada, de ordem n, simétrica e semidefinida positiva com $a_{ij} \in [\ 0\ ,\ 1\ )$.

O vetor b é gerado também de maneira aleatória com $b_i \in [\ -1\ ,\ 1\ )$ para $i = 1,\ 2,\ \cdots,\ n$.

Para este exemplo, o algoritmo de Dax não convergiu após 2.000 iterações com tolerância $TOL = 10^{-4}$ e $\varepsilon = 1\ ,\ 0{,}1$ e $0{,}01$.

Os resultados apresentados pelo algoritmo MINIMAXPAR estão dispostos na Tabela 11.

As tabelas a seguir apresentam os resultados das experiências computacionais efetuadas.

Tabela 1 - Algoritmo de Dax para o exemplo 1 com $TOL = 10^{-7}$ e diferentes valores de ε

ε	ITER	*VO*	SOLUÇÃO
10^{0}	33	0,10E + 01	(0,20E + 01 , 0,20E + 01)
10^{-1}	296	0,10E + 01	(0,20E + 01 , 0,20E + 01)
10^{-2}	*	-	-

Tabela 2 - Algoritmo MINIMAXPAR para o exemplo 1 com $TOL = 10^{-7}$ e diferentes valores de ε

ε	ITER	*VO*	SOLUÇÃO
10^{0}	534	0,10E + 01	(0,199998E + 01 , 0,20E + 01)
10^{-1}	1021	0,10E + 01	(0,20E + 01 , 0,20E + 01)
10^{-2}	*	-	-

Tabela 3 - Algoritmo de Dax para o exemplo 2 com $TOL = 10^{-7}$ e diferentes valores de ε

ε	ITER	*VO*	SOLUÇÃO
10^{0}	20	0,95E - 06	(0,3636361 , 0,4545454 , 0,4545453 , 0,3636362)
10^{-1}	17	0,51E - 05	(0,3636347 , 0,4545440 , 0,4545449 , 0,3636361)
10^{-2}	14	0,51E - 04	(0,3636199) , 0,4545313 , 0,4545380 , 0,3636346)
10^{-3}	11	0,55E - 03	(0,3634610 , 0,4543945 , 0,4544683 , 0,3636170)
10^{-4}	8	0,58E - 02	(0,3617733 , 0,4529407 , 0,4537250 , 0,3634313)

Tabela 4 - Algoritmo MINIMAXPAR para o exemplo 2 com $TOL = 10^{-7}$ e diferentes valores de ε

ε	ITER	*VO*	SOLUÇÃO
10^{0}	146	0,64E - 05	(0,3636350 , 0,4545429 , 0,4545428 , 0,3636348)
10^{-1}	118	0,64E - 04	(0,3636207 , 0,4545191 , 0,4545189 , 0,3636208)
10^{-2}	90	0,65E - 03	(0,3634747 , 0,4542726 , 0,4542726 , 0,3634746)
10^{-3}	63	0,62E - 02	(0,3620975 , 0,4519546 , 0,4519548 , 0,3620974)
10^{-4}	35	0,64E - 01	(0,3477381 , 0,4277822 , 0,4277822 , 0,3477381)

Tabela 5 - Algoritmos de Dax e MINIMAXPAR para o exemplo 3 com n=20, $TOL = 10^{-4}$ e diferentes valores de ε

ε	DAX			MINIMAXPAR		
	ITER	*VO*	TEMPO	ITER	*VO*	TEMPO
10^{1}	43	0,4013E + 00	0:00.05	1.043	0,4156E + 00	0:00.27
10^{0}	231	0,6299E - 01	0:00.06	1.997	0,1459E + 00	0:00.44
10^{-1}	308	0,3704E - 01	0:00.06	94	0,1522E + 01	0:00.06
10^{-2}	39	0,4141E + 00	-	6	0,7487E + 01	-

Tabela 6 - Algoritmos de Dax e MINIMAXPAR para o exemplo 3 com n=20, $TOL = 10^{-6}$ e diferentes valores de ε

ε	DAX			MINIMAXPAR		
	ITER	*VO*	TEMPO	ITER	*VO*	TEMPO
10^{1}	43	0,4013E + 00	0:00.05	1.129	0,4015E + 00	0:00.28
10^{0}	309	0,5561E - 01	0:00.05	*	-	-
10^{-1}	*	-	-	*	-	-
10^{-2}	*	-	-	1.959	0,1459E + 00	0:00.44
10^{-3}	308	0,3686E - 01	0:00.06	94	0,1522E + 01	0:00.06

Tabela 7 - Algoritmos de Dax e MINIMAXPAR para o exemplo 3 com n = 100, TOL= 10^{-4} e diferentes valores de ε

ε	DAX			MINIMAXPAR		
	ITER	*VO*	TEMPO	ITER	*VO*	TEMPO
10^{1}	209	0,4000E + 00	0:01.04	*	-	-
10^{0}	1.148	0,7215E - 01	0:05.72	540	0,3963E + 01	0:03.02
10^{-1}	214	0,3930E + 00	0:01.04	9	0,3512E + 02	0:00.05
10^{-2}	65	0,2151E + 01	0:00.39	5	0,9849E + 02	-

Tabela 8 - Algoritmos de Dax e MINIMAXPAR para o exemplo 3 com n = 100, TOL= 10^{-6} e diferentes valores de ε

ε	DAX			MINIMAXPAR		
	ITER	VO	TEMPO	ITER	VO	TEMPO
10^{1}	214	0,4000E + 00	0:01.10	*	-	-
10^{0}	1.484	0,5599E - 01	0:07.42	*	-	-
10^{-1}	*	-	-	*	-	-
10^{-2}	*	-	-	540	0,3963E + 01	0:03.02
10^{-3}	214	0,3935E + 00	0:01.10	9	0,3512E + 02	0:00.05

Tabela 9 - Algoritmos de Dax e MINIMAXPAR para o exemplo 4 com TOL= 10^{-4} e diferentes valores de ε

ε	DAX			MINIMAXPAR		
	ITER	*VO*	TEMPO	ITER	*VO*	TEMPO
10^{3}	*	-	-	343	0,7631E + 01	0:00.22
10^{2}	77	0,768E + 01	0:00.06	286	0,7628E + 01	0:00.06
10^{1}	67	0,7699E + 01	0:00.05	227	0,7600E + 01	0:00.05
10^{0}	58	0,7315E + 01	0:00.05	165	0,7320E + 01	0:00.05
10^{-1}	*	-	-	145	0,5546E + 01	0:00.05
10^{-2}	*	-	-	285	0,2911E + 01	0:00.06
10^{-3}	*	-	-	207	0,3235E + 01	0:00.06

Tabela 10 - Algoritmo MINIMAXPAR para o exemplo 5 com TOL = 10^{-4}

ε	m = 200 e n = 100			m = 1000 e n = 100		
	ITER	VO	TEMPO	ITER	VO	TEMPO
10^{0}	1.015	0,9826E + 00	0:01.55	139	0,9981E + 00	0:18.56
10^{-1}	217	0,9942E + 00	0:00.33	11	0,1011E + 01	0:01.59
10^{-2}	45	0,1046E + 01	0:00.06	34	0,1221E + 01	0:04.67
10^{-3}	305	0,1184E + 01	0:00.44	107	0,2589E + 01	0:14.28

Tabela 11 - Algoritmo MINIMAXPAR para o exemplo 6 com $TOL=10^{-4}$

ε	$n = 20$			$n = 100$		
	ITER	*VO*	TEMPO	ITER	*VO*	TEMPO
10^{0}	622	0,8433E + 00	0:00.11	699	0,9784E + 00	0:04.17
10^{-1}	738	0,8104E + 00	0:00.16	123	0,1022E + 01	0:00.77
10^{-2}	312	0,7416E + 00	0:00.06	4	0,1041E + 01	-

Conclusões

De maneira geral, algoritmos seqüenciais (tipo Gauss-Seidel), como é o caso do algoritmo de Dax (1991), tem convergência mais rápida do que algoritmos simultâneos (tipo Jacobi), como é o caso do algoritmo MINIMAXPAR. Entretanto, os métodos paralelos (simultâneos) são convenientes para a utilização em computadores com processamento paralelo. Se um método tipo Jacobi gasta em um computador convencional (1 processador central), um tempo de CPU $t = t_0$ para um determinado problema, com uma matriz **A** possuindo m linhas, então o tempo gasto para esse mesmo problema em um computador com m processadores paralelos é da ordem de $0(t) = t_0/m$, uma vez que há um tempo de sincronização/comunicação entre os processadores.

Com base em nossas experiências computacionais para os algoritmos de Dax e MINIMAXPAR e para os exemplos considerados, podemos concluir que:

1. Para aqueles sistemas lineares onde a solução $\boldsymbol{x}^*$ era conhecida, exemplos 1 e 2 tanto o algoritmo de Dax como o MINIMAXPAR forneceram excelentes aproximações para x^* para $\varepsilon = 1$, $\varepsilon = 10^{-1}$ ou $\varepsilon = 10^{-2}$. Quanto menor é o valor de ε, maior é o valor de VO, ou seja, a aproximação tende a ficar pior com a diminuição de ε.

2. Para o exemplo 3, o valor ótimo da função objetivo, mostrou-se apenas razoável. Embora não tabelado, observamos dos testes que as primeiras componentes do vetor solução apresentado pelos algoritmos eram bastante próximas de 1, enquanto para as últimas componentes isso não ocorreu. A consideração de uma menor tolerância não trouxe ganhos significativos para os dois métodos pois, neste caso, o número de iterações aumentou significativamente, enquanto o valor ótimo *VO* permaneceu com a mesma ordem de grandeza.

Considerando $n = 20$ e *VO* da ordem de 10^0, o algoritmo de Dax gastou um tempo de CPU igual a $5x10^{-2}$ sec e o MINIMAXPAR $28x10^{-2}$ sec. Para nenhum valor de ε, o algoritmo MINIMAXPAR obteve *VO* da ordem 10^{-1} (Tabelas 5 e 6). Para o caso onde $n = 100$ o algoritmo seqüencial de Dax teve um desempenho superior ao do algoritmo MINIMAXPAR (Tabelas 7 e 8).

3. Para o exemplo 4, o desempenho do algoritmo MINIMAXPAR foi superior ao de Dax. O algoritmo de Dax apresentou o melhor resultado para ε = 1 com um tempo de CPU igual a $5x10^{-2}$ sec em 58 iterações e *VO* da ordem de 10^1, enquanto que o método MINIMAXPAR obteve para $\varepsilon = 10^{-1}$ em 145 iterações o mesmo tempo de CPU e a mesma ordem de grandeza de *VO* do que o método de Dax.

4. Para os exemplos 5 e 6, o algoritmo de Dax não convergiu após 2.000 iterações para os valores de ε considerados. A utilização de uma menor tolerância neste caso, não traz ganhos significativos para a ordem de grandeza de *VO*, para o algoritmo MINIMAXPAR.

Do exposto, e de nossas experiências computacionais, consideramos que o algoritmo MINIMAXPAR é conveniente para ser utilizado quando da disponibilidade de um computador com arquitetura paralela e o sistema linear $Ax = b$ a ser resolvido é inconsistente, com **A** de grande porte, esparsa e sem uma estrutura detectável.

Agradecimentos: À FAPESP pelo financiamento parcial deste trabalho (proc. nº 95/1008-5). Aos Professores Drs. Adhemar Sanches e Euclides Braga Malheiros pelas sugestões apresentadas.

LOPES, J. M. An iterative parallel method for minimax problems. *Rev. Mat. Estat.,* São Paulo, v.21, n.3, p.21-39. 2003.

- *ABSTRACT: This study presents a new parallel iterative method to solve the estimation problem in an ℓ_∞ norm. This algorithm, a parallel version of Dax's algorithm, is a row-relaxation type which is a convenient when the system to be solved is inconsistent, large, sparse and unstructured.*
- *KEYWORDS: Linear systems, sparse matrix, minimum norm, parallel iterative methods, row-relaxation methods.*

Referências

BARTELS, R.H.; CONN, A.R., LI, Y. Primal methods are better than dual methods for solving overdetermined linear systems in the l_∞ sense*? Siam J. Numer. Anal.*, Philadelphia, v.26, p.693-726, 1989.

BAI, Z.Z. Parallel matrix multisplitting block relaxation iteration methods. *Math. Numer. Sinica,* New York, v.17, p.238-252, 1995.

BAI, Z.Z.; WANG, D. R.; EVANS, D. J. Models of asynchronous parallel matrix multisplitting relaxed iterations. *Parallel Comput.,* Amsterdam, v.21, p.565-582, 1995.

BAI, Z.Z. A class of parallel decomposition-type relaxation methods for large sparse systems of linear equations. *Linear Algebra Applic.,* New York, v.282, p.1-24, 1998.

BAI, Z.Z., SU, Y. On the convergence of a Class of parallel decomposition-type relaxation methods. *Appl. Math. Comput.,* Orland, v.81, p.1-21, 1997.

CENSOR, Y. Row-action methods for huge and sparse systems and their applications. *Siam Rev.*, Philadelphia, v.23, p.444-464, 1981.

CENSOR, Y.; ZENIOS, S. Introduction to methods of parallel optimization. In: COLÓQUIO BRASILEIRO DE MATEMÁTICA, 19.,1993, Rio de Janeiro. *Anais...*Rio de Janeiro: IMPA-CNPq, 1993.

CVETKOVIC, L.; OBROVSKI, J. Some convergence results of PD relaxation methods. *Appl. Math. Comput.,* Orlando, v.107, p.103-112, 2000.

CVETKOVIC, L. Some convergence conditions for a class of parallel decomposition-type linear relaxation methods. *Appl. Numer. Math.,* Orlando, v.41, p.81-87, 2002.

CHENEY, E.W. *Introduction to aproximation theory.* New York: Chelsea, 1982. 259p.

DAX, A. *A row relaxation method for large minimax problems*. Jerusalem: Hidrological Service, 1991. 32p.

_____. Sucessive refinemnt of large multicell models. *Siam J. Numer. Anal.,* Philadelphia, v.22, p.865-887, 1985.

DAX A.; BERKOWITZ, B. Column relaxation methods for least norm problems. *Siam Sci. Stat. Comput.,* Philadelphia, v.11, p.975-989,1990.

FROMMER, A.; MAYER, G. Convergence of relaxed parallel multisplitting methods. *Linear Algebra Applic.,* New York, v.119, p.141-152, 1989.

HADJIDIMOS, A. Accelerated overrelaxation method. *Math. Comput.,* v.32, n. 141, p.149-157, 1978

LOPES, J.M.; DE PIERRO, A.R. Um método iterativo paralelo para estimação de mínimo valor absoluto. *Pesq. Operacional*, Rio de Janeiro, v.12, p.31-46, 1992.

MANGASARIAN, O.L. Iterative solution of linear programs. *Siam J. Numer. Anal.,* Philadelphia, v.18, p.606-614, 1981.

MURTY, K.G. *Linear complementarity, linear and nonlinear programming.* Berlin: Heldermann Verlag, 1988. 629p.

NEUMANN, M.; PLEMMONS, R. J. Convergence of parallel multisplitting iterative methods for M-matrices. *Linear Algebra Applic.,* New York, v.88, p.559-573, 1987.

RUGGIERO, M.A.G.; LOPES, V.L.R. *Cálculo numérico:* aspectos teóricos e computacionais. São Paulo: McGraw-Hill, 1992. 406p.

SAAD, Y.; VORST, H. V.D. Iterative solution of linear systems in the 20^{th} century. *J. Comput. Appl. Math.,* Amsterdam, v.123, p.1-33, 2000.

SPOSITO, V.A. *Linear and nonlinear programming*. Iowa: The Iowa State University Press, 1975. 269p.

TIKHNOV, A.N.; ARSENIN, V.Y. *Solutions of ill-posed problems*. New York: John Wiley, 1977. 320p.

WANG, D. R. On the convergence of the parallel multisplitting AOR algorithm . *Linear Algebra Applic.,* New York, v.154-156, p.473-486, 1991.

Recebido em 21.11.2002.

Aprovado após revisão em 05.05.2003.