Proceeding Series of the Brazilian Society of Computational and Applied Mathematics, v. 11, n. 1, 2025.

Trabalho apresentado no XLIII CNMAC, Centro de Convenções do Armação Resort - Porto de Galinhas - PE, 2024

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Uma Nova Abordagem para o Cálculo do Precondicionador Separador

Caroline M. S. Saba[1], Aurelio R. L. Oliveira[2]
Unicamp, Campinas, SP
Julianna P. S. Porto[3]
UFRB, Cruz das Almas, BA

Resumo. Os métodos de pontos interiores são ferramentas eficientes para a resolução de problemas de programação linear de grande porte. Como este método pode ter um alto custo computacional, métodos iterativos precondicionados são usados para resolver os sistemas lineares oriundos do método de pontos interiores. O precondicionador separador é utilizado nas iterações finais do método, onde os sistemas lineares se tornam muito mal condicionados. Mas pode ser caro calcular esse precondicionador, cuja implementação depende, dentre outros aspectos, de uma escolha adequada de colunas linearmente independentes da matriz original para construir a base do mesmo. Por isso, neste trabalho, é proposta uma nova abordagem para o cálculo da base do precondicionador separador, com o objetivo de reduzir o custo computacional do método. Além disso, a abordagem proposta é comparada com a abordagem já existente, a fim de mostrar que a base obtida é a mesma que foi encontrada pelos métodos anteriores.

Palavras-chave. Precondicionador, Método de Pontos Interiores, Programação Linear

1 Introdução

Existem vários estudos sobre os métodos de pontos interiores (MPI) nos quais são desenvolvidas implementações sofisticadas para resolver problemas de programação linear (PL), conforme [1] e [8]. A característica principal do MPI é encontrar uma solução para o PL em poucas iterações, entretanto este método pode ter um alto custo computacional [6], pois cada iteração envolve a resolução de um ou mais sistemas lineares, para encontrar as direções de busca. Para diminuir o custo, são utilizados métodos iterativos. No entanto, os métodos iterativos não tem um bom desempenho para resolver sistemas cuja matriz é muito mal-condicionada, o que acontece com frequência em métodos de pontos interiores. Por isso, são utilizados métodos iterativos precondicionados, pois os precondicionadores melhoram o número de condição da matriz.

Os sistemas lineares tornam-se mal condicionados nas iterações finais do MPI, mas não apresentam esse comportamento nas iterações iniciais. Por isso, torna-se necessário a utilização de precondicionadores híbridos [1]. O precondicionador Fatoração Controlada de Cholesky (FCC) [2] é utilizado nas iterações iniciais do método e o precondicionador separador (PS) [8] é utilizado quando o sistema se aproxima de uma solução do PL.

Neste trabalho é proposta uma nova abordagem para a escolha das colunas linearmente independentes (LI) que formam a base do PS, com o objetivo de reduzir o custo computacional do método. Além disso, a base obtida pela nova abordagem é comparada, por meio de experimentos

[1]carolinemartins@ime.unicamp.br
[2]aurelio@ime.unicamp.br
[3]julianna.pinele@ufrb.edu.br

numéricos, com a base obtida em [1] e [8], com o objetivo de mostrar que as colunas selecionadas para formar a base são as mesmas em ambas as abordagens.

O texto está organizado como se segue. Na Seção 2 são definidos os problemas de programação linear, é apresentado o MPI preditor-corretor e como calcular as direções de busca. Na Seção 3, o PS é descrito e a nova estratégia para escolher o conjunto de colunas que forma a base do precondicionador é apresentada. Na Seção 4 são descritos os experimentos numéricos realizados e os resultados obtidos e, na Seção 5 estão as considerações finais.

2 Problemas de Programação Linear

Considere o PL na forma padrão

$$\begin{aligned} & \min c^t x \\ \text{(P)} \quad & \text{s.a. } Ax = b \ , \\ & x \geq 0 \end{aligned} \tag{1}$$

onde $A \in \mathbb{M}_{m\times n}(\mathbb{R})$ é uma matriz de posto completo, $c, x \in \mathbb{R}^n$ e $b \in \mathbb{R}^m$.

O dual do PL (1) é definido da seguinte forma

$$\begin{aligned} & \max b^t y \\ \text{(D)} \quad & \text{s.a. } A^t y + z = c \ , \\ & z \geq 0 \end{aligned} \tag{2}$$

onde $y \in \mathbb{R}^m$ é um vetor de variáveis livres e $z \in \mathbb{R}^n$ é um vetor de variáveis de folga duais.

As condições de otimalidade, conhecidas também como condições de Karush-Kuhn-Tucker, que podem ser encontradas em [9], são dadas por

$$\begin{aligned} & Ax = b \\ & A^t y + z = c \quad (x, z) \geq 0 \ , \\ & XZe = 0 \end{aligned} \tag{3}$$

onde $X = diag(x)$, $Z = diag(z)$ e o vetor e é unitário.

2.1 Método de Pontos Interiores

A variante Preditor-Corretor [7] é a mais popular, dentre os vários tipos de métodos de pontos interiores primal-dual. Neste método, obtêm-se as direções de busca pela resolução de dois sistemas lineares, que são obtidos pela aplicação do método de Newton às condições KKT do problema de programação linear na forma padrão, dadas em (3). Primeiro, calcula-se a direção afim resolvendo

$$\begin{bmatrix} 0 & I & A^t \\ Z & X & 0 \\ A & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\tilde{x} \\ \Delta\tilde{z} \\ \Delta\tilde{y} \end{bmatrix} = \begin{bmatrix} r_d \\ r_a \\ r_p \end{bmatrix}, \tag{4}$$

onde os resíduos são dados por

$$\begin{aligned} r_d &= c - A^t y - z, \\ r_p &= b - Ax, \\ r_a &= -XZe. \end{aligned}$$

Em seguida, as direções de busca $(\Delta x, \Delta y, \Delta z)$ são calculadas resolvendo (4) substituindo r_a por $r_m = \mu e - XZe - \Delta\tilde{x}\Delta\tilde{z}e$, onde μ é o parâmetro de centragem.

DOI: 10.5540/03.2025.011.01.0501

2.2 As direções de busca

A resolução dos sistemas lineares é o passo principal de cada iteração do método, em termos de custo computacional. Como ambos os sistemas possuem a mesma matriz, eliminando Δz de (4), obtém-se

$$\begin{bmatrix} -D & A^t \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} r_d - X^{-1} r_a \\ r_p \end{bmatrix}, \tag{5}$$

onde $D = X^{-1}Z$ é uma matriz diagonal com as entradas da diagonal positivas. O sistema (5) é chamado de sistema aumentado. Eliminando Δx de (5), obtemos

$$AD^{-1}A^t \Delta y = r_p + A\left(D^{-1} r_d - Z^{-1} r_a\right), \tag{6}$$

o qual é referido como sistema de equações normais, onde $S = AD^{-1}A^t$ é chamado o complemento de Schur.

Observa-se que a matriz de coeficientes de (6) é simétrica e positiva definida. Assim, o sistema (6) pode ser resolvido utilizando a fatoração de Cholesky. Entretanto, esta fatoração pode ter um alto custo computacional para problemas de grande porte, pois pode gerar muito preenchimento.

Para algumas classes de problema de grande porte, deve-se utilizar um método iterativo precondicionado para resolver os sistemas lineares, como proposto, por exemplo, em [1] e [8]. A abordagem aqui proposta é implementada para determinar as colunas da base do precondicionador separador, o qual é utilizado nas iterações finais do método do gradiente conjugado precondicionado para resolver o sistema de equações normais.

3 Precondicionador Separador

O precondicionador separador, proposto em [8], é calculado da seguinte maneira: Seja $A = [B \;\; N]\, P$, onde $P \in \mathbb{R}^{n\times n}$ é uma matriz de permutação tal que $B \in \mathbb{R}^{m\times m}$ é não singular e $N \in \mathbb{R}^{m\times(n-m)}$, então

$$\begin{aligned} AD^{-1}A^t &= \begin{bmatrix} B & N \end{bmatrix} PD^{-1}P^t \begin{bmatrix} B^t \\ N^t \end{bmatrix} = \begin{bmatrix} B & N \end{bmatrix} \begin{bmatrix} D_B^{-1} & 0 \\ 0 & D_N^{-1} \end{bmatrix} \begin{bmatrix} B^t \\ N^t \end{bmatrix} \\ &= BD_B^{-1}B^t + ND_N^{-1}N^t. \end{aligned}$$

Agora, pré-multiplicando por $D_B^{\frac{1}{2}}B^{-1}$, que é o precondicionador, e pós-multiplicando por sua transposta, temos a matriz M precondicionada, dada por

$$M = D_B^{\frac{1}{2}}B^{-1}\left(AD^{-1}A^t\right)B^{-t}D_B^{\frac{1}{2}} = I_m + WW^t, \tag{7}$$

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

Perto de uma solução, pelo menos $n-m$ entradas de D são muito grandes. Assim, com uma escolha adequada das colunas de B, obtemos entradas da diagonal de D_B e D_N^{-1} que se aproximam de zero. Consequentemente, W se aproxima da matriz nula, M se aproxima da matriz identidade e tanto o número de condição na norma-2 de M, $k_2(M)$, quanto o maior autovalor de M se aproximam de 1, como provado em [8].

A característica mais importante do PS é trabalhar bem nas iterações finais do método de pontos interiores, onde os sistemas lineares são muito mal condicionados. Mas o processo para encontrar a matriz B tem um alto custo computacional. Por isso, essa classe de precondicionadores requer uma implementação cuidadosa. Várias estratégias para a implementação de um código eficiente, como colunas chave, colunas simbolicamente dependentes e independentes, componentes fortemente conexas, são descritas em [5].

DOI: 10.5540/03.2025.011.01.0501

3.1 Escolhendo o conjunto de colunas

Em [3], a estratégia utilizada para determinar B é minimizar $||W||$, que é um problema difícil de resolver, mas ele pode ser aproximado por uma heurística barata: selecionar as primeiras m colunas LI de AD^{-1} com menor norma-2. Com essa estratégia, WW^t se aproxima da matriz nula nas iterações finais do método de pontos interiores e M, dada em (7), tende a ser uma matriz melhor condicionada.

Em uma das abordagens propostas neste trabalho, as colunas da matriz B são reordenadas por esparsidade, ou seja, de acordo com o número de elementos não nulos de cada coluna em ordem não decrescente, e então é calculada a fatoração de Cholesky da matriz B^tB. Quando uma coluna linearmente dependente (LD) aparece (considere que a coluna k seja LD), são analisadas as colunas anteriores (à coluna k na matriz B) que possuem elementos não nulos na linha do possível pivô da coluna k: se dentre essas colunas, alguma delas, digamos a coluna j, possuir $D_{jj} > D_{kk}$, a coluna j é eliminada da matriz B. A coluna $m+1$ da matriz A (ordenada pelos valores de D) é colocada na matriz B, respeitando a ordem por esparsidade. As colunas $m+2$ até n da matriz A são movidas uma posição à esquerda, recalcula-se a matriz B^tB e reinicia-se a fatoração dessa matriz. Esse processo é realizado toda vez que for encontrada uma coluna LD na fatoração, até serem encontradas as m colunas LI que formam a matriz B. Essa proposta será chamada de P1.

Na outra abordagem proposta, as colunas da matriz original A são reordenadas de acordo com os valores da diagonal de D, em ordem não decrescente. As primeiras m colunas de A são consideradas como uma candidata à matriz B. Calcula-se então a fatoração de Cholesky da matriz B^tB e quando uma coluna LD aparece (considere que a coluna k seja LD), esta coluna é eliminada. As colunas $k+1$ até n de A são movidas uma posição à esquerda e a fatoração de B^tB é reiniciada com a nova matriz B. Esse processo é realizado até serem encontradas as m colunas LI que formam a matriz B. Essa proposta, que será chamada de P2, já é realizada, usando fatoração LU, em [8].

4 Experimentos Numéricos

Nesta seção, são apresentados os experimentos numéricos realizados com as duas propostas descritas na Seção 3 para obter a matriz B. Estes experimentos foram realizados para comparar a base obtida pelo novo reordenamento proposto com a base obtida em [1] e [8]. Os procedimentos para encontrar a matriz B foram implementados no Octave, em um processador Intel Core i5, 8GB de RAM e sistema operacional Windows. O conjunto de problemas teste é da coleção NETLIB[4] [4].

Na Tabela 1 apresentam-se os problemas de Programação Linear utilizados para os testes, os respectivos número de linhas, número de colunas e o número de elementos não nulos (nnz) da matriz A de restrições desses problemas.

Além dos códigos das propostas P1 e P2, foi implementado um código para comparar as colunas da matriz A selecionadas para formar a matriz B, obtidas nas duas propostas. Na matriz do problema kb2, as colunas encontradas foram as mesmas e o número de elementos não nulos também foi o mesmo em ambas as propostas. Nos demais problemas, nota-se que aparecem poucas colunas diferentes, que mudam a ordem das colunas, mas os índices das colunas encontrados para formar a matriz B foram praticamente os mesmos.

Na Tabela 2 é apresentado o tempo, em segundos, utilizado por cada proposta para encontrar a matriz B. Analisando a Tabela 2, observa-se que o tempo usado para encontrar a matriz B é menor no código P1 para todas as matrizes com número de linhas ou colunas maior do que 330, exceto para a matriz do problema scsd1 (ver Tabela 1). Para os problemas menores, o tempo usado é menor no código P2. Assim, a proposta P1 obteve um melhor desempenho com relação

[4]NETLIB é uma biblioteca de problemas de PL disponibilizada na internet em `http://www.netlib.org/lp/data/`.

DOI: 10.5540/03.2025.011.01.0501

Tabela 1: Dados dos problemas.

Problema	Linhas	Colunas	nnz
adlittle	56	138	424
agg	488	615	2862
kb2	43	68	313
sc50a	50	78	160
sc205	205	317	665
scagr25	471	671	1725
scfxm1	330	600	2732
scfxm2	660	1200	5469
scfxm3	990	1800	8206
scrs8	490	1275	3288
scsd1	77	760	2388
scsd6	147	1350	4316
scsd8	397	2750	8584
sctap2	1090	2500	7334
sctap3	1480	3340	9734
seba	515	1036	4360
shell	536	1777	3558
stair	356	620	4021
truss	1000	8806	27836

Tabela 2: Tempo de processamento para encontrar a matriz B.

Problema	Tempo - P1	Tempo - P2
adlittle	0,011266	0,032322
agg	1,5492	3,5989
kb2	0,0022120	0,0011630
sc50a	0,028517	0,0030620
sc205	0,35188	0,021634
scagr25	0,56891	5,1679
scfxm1	6,8771	5,9727
scfxm2	21,034	30,987
scfxm3	79,097	107,53
scrs8	1,9963	11,248
scsd1	0,031232	0,0043449
scsd6	0,10606	0,23507
scsd8	2,9947	4,0594
sctap2	74,709	132,77
sctap3	156,26	394,08
seba	2,2347	12,511
shell	87,114	93,178
stair	1,5527	2,3591
truss	26,554	792,59

ao tempo de processamento em problemas maiores, porque nessa proposta as colunas da matriz B são reordenadas por esparsidade, logo, as decomposições que estão envolvidas nessa proposta são muito mais esparsas e, por isso, geram menos preenchimento e realizam menos operações do que a

outra proposta. Nota-se que no problema truss, que foi o problema testado com maior dimensão, a redução no tempo é muito significativa.

5 Considerações Finais

Os testes realizados tiveram como objetivo mostrar que, qualquer que seja o reordenamento, a nova proposta obtém a mesma base que foi encontrada pelos métodos anteriores. Com relação às colunas encontradas, considerando que os resultados obtidos dependem de erros de arredondamento e de parâmetros, além do fato de estar sendo usada a fatoração de Cholesky, pode-se considerar que os resultados obtidos pelos testes foram satisfatórios. Com relação ao tempo, a proposta P1 obteve um bom desempenho processando os problemas que possuem dimensões maiores. Sabe-se que no Octave existe uma limitação para trabalhar com problemas de grande porte, maiores do que os problemas testados neste trabalho, que são as dimensões com as quais se trabalha normalmente. Futuramente, pretende-se melhorar o cálculo do PS, implementando a ideia proposta neste trabalho usando a fatoração LU retangular na linguagem C, que é mais eficiente para encontrar a matriz B do que a fatoração de Cholesky, incorporada ao código PCx, veja [8].

Agradecimentos

Ao CNPq e à Universidade Federal do Recôncavo da Bahia pelo apoio financeiro.

Referências

[1] S. Bocanegra, F. F. Campos e A. R. L. Oliveira. "Using a Hybrid Preconditioner for Solving Large-Scale Linear Systems arising from Interior Point Methods". Em: **Computational Optimization and Applications** 36.1–2 (2007), pp. 149–164.

[2] F.F. Campos. "Analysis of Conjugate Gradients - type methods for solving linear equations". Tese de doutorado. Oxford University Computing Laboratory, 1995.

[3] C.O. Castro, M.R. Heredia e A. R. L. Oliveira. "Recycling basic columns of the splitting preconditioner in interior point methods". Em: **Computational Optimization and Applications** 86 (2023), pp. 49–78. DOI: https://doi.org/10.1007/s10589-023-00492-13.

[4] D.M. Gay. "Electronic Mail Distribution of Linear Programming Test Problems". Em: **Mathematical Programming Society COAL Newsletter** 13 (1985), pp. 10–12. DOI: https://doi.org/10.1007/s10589-023-00492-13.

[5] C.T.L.S. Ghidini, A. R. L. Oliveira e D. C. Sorensen. "Computing a hybrid preconditioner approach to solve the linear systems arising from interior point methods for linear programming using the conjugate gradient method". Em: **Annals of Management Science** 3.1 (2014), pp. 45–66. DOI: 10.24048/ams3.no1.2014-43.

[6] J. Gondzio. "Interior Point Methods 25 Years Later". Em: **European Journal of Operational Research** 218 (2012), pp. 587–601.

[7] S. Mehrotra. "On the implementation of a primal-dual interior point method". Em: **SIAM Journal on Optimization** 2.4 (1992), pp. 575–601. DOI: 10.1137/0802028.

[8] A. R. L. Oliveira e D. C. Sorensen. "A new class of preconditioners for large-scale linear systems from interior point methods for linear programming". Em: **Linear Algebra and Its Applications** 394 (2005), pp. 1–24. DOI: https://doi.org/10.1016/j.laa.2004.08.019.

[9] S. J. Wright. **Primal-Dual Interior-Point Methods**. Philadelphia: Society for Industrial e Applied Mathematics (SIAM), 1997. ISBN: 9781611971453.

