Universidade do Estado do Rio de Janeiro

Centro de Tecnologia e Ciências

Faculdade de Engenharia

Moisés Ceni de Almeida

Precondicionador de Inversa Aproximada por Blocos

Rio de Janeiro

2019

Moisés Ceni de Almeida

Precondicionador de Inversa Aproximada por Blocos

Tese apresentada, como requisito parcial para obtenção do título de Doutor em Ciências, ao Programa de Pós-Graduação em Engenharia Mecânica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Fenômenos de Transporte.

Orientador: Prof. Dr. Luiz Mariano Carvalho

Rio de Janeiro

2019

CATALOGAÇÃO NA FONTE
UERJ / REDE SIRIUS / BIBLIOTECA CTC/B

A447 Almeida, Moisés Ceni de.
Precondicionador de inversa aproximada por blocos / Moisés Ceni de Almeida. – 2019.
95f.

Orientador: Luiz Mariano Paes de Carvalho Filho.
Tese (Doutorado) – Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia.

1. Engenharia mecânica - Teses. 2. Sistemas lineares - Teses. 3. Matriz inversa - Teses. 4. Algorítmos - Teses. 5. Análise matemática – Teses. I. Carvalho Filho, Luiz Mariano Paes de. II. Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia. III. Título.

CDU 517.956

Bibliotecária: Júlia Vieira – CRB7/6022

Assinatura

Data

Moisés Ceni de Almeida

Precondicionador de Inversa Aproximada por Blocos

Tese apresentada, como requisito parcial para obtenção do título de Doutor em Ciências, ao Programa de Pós-Graduação em Engenharia Mecânica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Fenômenos de Transporte.

Aprovada em: 27 de Março de 2019

Banca Examinadora:

Prof. Dr. Luiz Mariano Carvalho (Orientador)
Instituto de Matemática e Estatística da UERJ

Prof. Dr. Norberto Mangiavachi
Faculdade de Engenharia da UERJ

Prof. Dr. Nelson Maculan
COPPE - UFRJ

Prof. Dr. Paulo Goldfeld
Instituto de Matemática da UFRJ

Prof. Dr. Michael Ferreira de Souza
Departamento de Estatística e Matemática Aplicada da UFC

Rio de Janeiro

2019

AGRADECIMENTO

Agradeço primeiro a Deus, por não me deixar sozinho em nenhum instante dessa luta e por não desistir de sonhar comigo.

Aos meus pais, por serem base pra mim.

À minha família, em especial aos meus irmãos, que me apoiaram.

À Viviane, por ser minha motivação.

Aos meus companheiros de jornada Ítalo e Zanardi.

Ao meu orientador e amigo, Dr. Luiz Mariano Paes de Carvalho Filho, por me servir de inspiração.

Aos colaboradores do meu trabalho, os amigos/professores Dr. Michael Souza e Dr. Paulo Goldfeld.

À CAPES por financiar meu doutorado.

RESUMO

ALMEIDA, Moisés Ceni de. *Precondicionador de Inversa Aproximada por Blocos*. 95 f. Tese (Doutorado em Engenharia Mecânica) - Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro (UERJ), Rio de Janeiro, 2019.

Neste trabalho apresentamos o Algoritmo de Inversa Aproximada por blocos, o qual gera um precondicionador para sistemas lineares esparsos de grande porte. Faremos o tratamento matemático e apresentaremos as condições suficientes para execução do Algoritmo. Além disso, apresentaremos algumas provas matemáticas das boas propriedades referentes à robustez do AINV.

Palavras-chave: Sistema Linear; Precondicionador; Inversa Aproximada; Matrizes em bloco.

ABSTRACT

ALMEIDA, Moisés Ceni de. *Block Approximate Inverse Preconditioner*. 95 f. Thesis (PhD in Mechanical Engineering) - Engineering Faculty, Rio de Janeiro State University (UERJ), Rio de Janeiro, 2019.

In this work, we present a block approximate inverse algorithm, which generates a preconditioner for sparse linear systems. We discuss the mathematical treatment and present sufficient conditions for the algorithm. In addition, we present some mathematical proofs of the good properties regarding the robustness of the AINV.

Keywords: Linear System; Preconditioner; Approximate Inverse; Block Matrices.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

INTRODUÇÃO

Afim de resolver um sistema linear $Ax = b$, onde A é matriz quadrada não-singular, é comum o uso de um métodos diretos baseados na eliminação gaussiana com pivoteamento parcial. Infelizmente, tanto em tempo de computação, quanto em uso memória, esses métodos são inviáveis para sistemas lineares grandes, esparsos e não estruturados.

Uma alternativa viável são os métodos de projeção em subespaços de Krylov (KSP) [40, 69, 70]. No caso de matrizes simétricas e positivo definidas (SPD), a escolha canônica é o método de gradientes conjugados (CG, na sigla em inglês) [44]. Contudo, é fato conhecido que os KSPs podem convergir lentamente sem um precondicionador adequado, [40]. Em geral, a ideia por trás de qualquer precondicionador é construir um operador P tal que $PAx = Pb$ ou $APy = b$, com $Py = x$, seja mais apropriado para a aplicação de um método de Krylov do que o sistema original $Ax = b$. Idealmente, o precondicionador P deveria ser igual a A^{-1} requerendo apenas uma iteração, mas isso é mais caro que quanto resolver o sistema linear original. Então, por uma questão de eficiência, um precondicionador, quando aplicado a um sistema linear específico, necessita levar em consideração uma *fase de construção*, onde o precondicionador P é montado, e uma *fase de aplicação*, onde o método de Krylov é aplicado ao sistema precondicionado. A arte de criar precondicionadores é encontrar um balanço entre o tempo gasto entre cada uma dessas fases.

Em relação à aplicação, precondicionadores podem ser classificados em *implícitos* ou *explícitos*. Os precondicionadores implícitos baseiam-se no cálculo de uma aproximação de A, como, por exemplo, a fatoração LU incompleta (ILU) [60]. Nesse caso, na fase de aplicação, são resolvidos dois sistemas lineares triangulares.

As estratégias de precondicionadores explícitos, por outro lado, dependem da computação e armazenamento de uma aproximação esparsa de A^{-1} que reduz a etapa de aplicação a um produto de uma matriz esparsa por um vetor denso. Enquanto a construção de alguns precondicionadores explícitos pode ser difícil de se paralelizar, a aplicação é massivamente paralela, com bibliotecas nativas com desempenho notável em várias arquiteturas paralelas e em vários processadores (CPU, GPU) [11, 12, 33, 42, 53, 80]. Alguns autores vão além e argumentam que os algoritmos de inversa aproximada podem ser mais

robustos na prática do que a classe de precondicionadores implícitos [9, 13]. AINV [12], FSAI [53,80], ISM-based [23], SPAI [42], and BIF [25,26] estão entre os algoritmos pertencentes à classe inversa aproximada. Notavelmente, as fases de instalação e aplicação do FSAI e SPAI são totalmente paralelas. Em contraste Algoritmos como o AINV, a ISM, baseada na Inversa de Sherman-Morrison e a Fatoração Balanceada Incompleta (BIF) são baseados em algoritmos de ortogonalização de Gram-Schmidt, clássico ou modificado, (CGS ou MGS) ou ILU, que podem apresentar um baixo nível de paralelismo dependendo de decisões de implementação, como o uso de algoritmos CGS ou MGS.

Neste trabalho, vamos estudar a teoria envolvendo a versão bloco do AINV. As principais vantagens para se ter estrutura de bloco são as melhorias de desempenho e a estabilidade. Primeiramente, a blocagem é uma técnica padrão para melhorar o desempenho dos cálculos matriciais especialmente com matrizes esparsas, onde o endereçamento indireto de esquemas de armazenamento esparsos reduz a localização durante o produto de matriz esparsa por vetor denso [2, 27, 35]. Em muitas aplicações, como na simulação do reservatório de petróleo, cada ponto de malha pode estar relacionado a propriedades distintas (pressão e saturações de diversos hidrocarbonetos), originando uma estrutura de matriz com blocos densos e quadrados. Portanto, esses problemas podem se beneficiar da combinação de paralelismo proveniente da aplicação de um precondicionador do tipo inversa aproximada e operações de álgebra linear densa devido à ocorrência de blocos densos. Além disso, explícitos ou não, os precondicionadores são aproximações que devem ser razoavelmente significativas, especialmente quando se trata de equações lineares de sistemas físicos. Consequentemente, se na matriz original os coeficientes são blocados por orientações físicas, então é preferível um precondicionador blocado da mesma maneira. Na verdade, alguns autores advogam que o ILU em blocos pode ser mais robusto e mais eficiente do que a versão escalar desse algoritmo [72, 73].

Existem alternativas em blocos para alguns dos métodos de inversa aproximada. Entre eles podemos encontrar, BFSAI [37, 48–50] , BSPAI [8, 77], Block ISM-based [28] e BSAINV [10]. Todas essas alternativas pretendem fornecer uma melhor combinação das duas características principais dos precondicionadores para matrizes de blocos esparsos: paralelismo, com base na multiplicação matriz-vetor na fase de aplicação e aumento da intensidade computacional, devido ao uso de operações de álgebra linear em blocos densos.

Neste trabalho, apresentamos o desenvolvimento teórico de um precondicionador

explícito para matrizes SPD em bloco proposto em [10]. Neste artigo, os autores descrevem brevemente sua alternativa de bloco SAINV (SBAINV), onde S significa estabilizado, com um passo extra, que retira a possibilidade de quebras no algoritmo. Os autores concluem que, para problemas de mecânica estrutural, o SBAINV produz resultados inferiores ao SAINV quanto ao número de iterações e ao tempo. Além disso, o filtro de preenchimento pode afetar o desempenho do SBAINV.

Neste trabalho, vamos estudar sistematicamente o BAINV e o SBAINV, além de apresentar a prova da estabilidade do BAINV para matrizes $\mathcal{M}$ e, mais geralmente, matrizes $\mathcal{H}$.

Um fato importante sobre os precondicionadores produzidos pelo AINV é que eles costumam ser densos. Assim, seu cálculo para grandes matrizes é inviável. Portanto, faz-se indispensável um processo de esvaziamento do precondicionar durante seu processamento. Esse processo é chamado de descarte da matriz. Apresentamos uma análise de algumas novas técnicas de descarte e os precondicionadores oriundos deles.

Este trabalho está organizado do seguinte modo: o Capítulo 1 trata da resolução de Sistemas Lineares via métodos iterativos, em especial os Métodos de Krylov. No capítulo 2 faremos uma análise de artigos relacionados ao Precondicionador AINV. No capítulo 3 nós apresentaremos formalmente o Algoritmo de Inversa Aproximada em Blocos (BAINV) e demonstraremos algumas propriedades importantes, além de apresentar a prova formal para uma classe de matrizes que generaliza o conceito de matrizes simétricas e positivo definidas. Nos Capítulos 4 e 5 apresentaremos as provas de que o BAINV é livre de quebras para matrizes $\mathcal{M}$ e $\mathcal{H}$. Finalmente, no Capítulo 6 apresentaremos variantes em bloco de alguns algoritmos e a proposição de trabalhos futuros.

1 SISTEMAS LINEARES E MÉTODOS ITERATIVOS

A fim de introduzir os conceitos e as ideias principais para a solução de um sistema linear através de um método de Krylov, usaremos como principais referências e seguiremos fortemente os trabalhos de Ipsen, Motta e Lago, respectivamente, [47, 55, 63].

Considere o sistema linear:

$$Ax = b, \tag{1}$$

onde A é matriz quadrada de ordem n e $b \in \mathbb{R}^n$.

Neste trabalho trataremos de sistemas como o da Equação (1), quando A for uma matriz grande, quando n é muito grande e esparsa (i.e, com muitos zeros) ou caso a matriz seja dada implicitamente, como por exemplo uma função $f(x)$ definida como uma sub-rotina dentro de um Algoritmo. Tais matrizes surgem naturalmente da discretização de EDP's com o uso do Método das Diferenças Finitas (FDM, na sigla em inglês), com o Método de Volumes Finitos (MVF, na sigla em inglês) ou com o Método dos Elementos Finitos (FEM, na sigla em inglês). Para tais matrizes, métodos diretos são desaconselháveis por possuírem um custo de ordem n^3. Nessas ocasiões, os métodos iterativos são mais competitivos.

Os métodos iterativos mais aplicados atualmente são os métodos de Krylov. Os métodos de Krylov são considerados uma das dez mais importantes classes de métodos numéricos, [34]. Como discutiremos a seguir, a solução da Equação (1) reside em um espaço de Krylov que possui a mesma dimensão do polinômio mínimo de A. Assim, quando o polinômio mínimo de A possui grau pequeno então o espaço procurado também é pequeno e, nesse caso, o método tem a oportunidade de convergir rapidamente.

1.1 Operadores de Projeção

Nesta seção discutiremos alguns pontos relevantes da Álgebra Linear que nos ajudarão nas discussões posteriores.

Definição 1. *Seja P uma transformação linear. Então diremos que $P : \mathbb{R}^n \to \mathbb{R}^n$ é uma projeção se $P^2 = P$. Neste caso, diremos também que P é idempotente.*

Definição 2. *A transformação $I : \mathbb{R}^n \to \mathbb{R}^n$ tal que $I(x) = x$ para todo $x \in \mathbb{R}^n$ é chamada de transformação identidade.*

Definição 3. *Dada uma transformação $T : \mathbb{R}^n \to \mathbb{R}^n$, definimos o núcleo da transformação T como sendo o conjunto $\mathcal{N}(T) = \{x \in \mathbb{R}^n; T(x) = 0\}$ e a imagem da transformação T como sendo o conjunto $Im(T) = \{y \in \mathbb{R}^n; \exists x \in \mathbb{R}^n, T(x) = y\}$.*

O próximo teorema nos encaminhará aos resultados mais aguardados dessa seção: a partição do espaço $\mathbb{R}^n$ como soma direta do núcleo de uma projeção com sua imagem.

Teorema 1. *Se P é uma projeção então:*

- $\mathcal{N}(P) = Im(I - P)$;
- $\mathcal{N}(P) \cap Im(P) = \{0\}$;
- *Se* $x \in Im(P) \Longrightarrow P(x) = x$.

Demonstração. Dado $x \in \mathcal{N}(P)$, então $P(x) = 0$ e, consequentemente, $x - P(x) = (I - P)(x) = x \in Im(I - P)$. Por outro lado, se $y \in Im(I - P)$ então existe $x \in \mathbb{R}^n$ tal que $(I - P)(x) = x - P(x) = y$ e, aplicando P em ambos os lados teremos $P(y) = P(x) - P(x) = 0$, donde concluímos que $y \in \mathcal{N}(P)$.

Agora, seja $x \in \mathcal{N}(P) \cap Im(P)$. Então, $P(x) = 0$ e existe y tal que $P(y) = x$. Assim, $P(y) = P(x) = 0$ e, como $P(y) = x$ devemos ter $x = 0$. A inclusão $\{0\} \subset \mathcal{N}(P) \cap Im(P)$ é trivial.

O terceiro item segue imediatamente da definição de projeção. □

Do item 2 Teorema 1 segue que:

$$\mathbb{R}^n = \mathcal{N}(P) \oplus Im(P).$$

Assim, portanto, do item 1 temos que qualquer vetor $x \in \mathbb{R}^n$ pode ser escrito como $x = x_1 + x_2$, aonde $x_1 \in Im(I - P)$ e $x_2 \in Im(P)$.

Reciprocamente, dados dois subespaços M e N tais que a soma direta resulte em $\mathbb{R}^n$, então existe uma única projeção P tal que $Im(P) = M$ e $\mathcal{N}(P) = N$ e, consequentemente, se $u = P(x) \in M$ então $x - u \in N$ (para uma demonstração completa, veja [63]).

Sem perda de generalidade, a matriz de transformação do operador de projeção P também será denotada por P.

Definição 4. *Uma projeção P é dita ortogonal se*

$$P^{\mathsf{T}} = P.$$

Dito de outra forma, uma projeção é dita ortogonal se é simétrica. Caso contrário, a projeção será dita Oblíqua.

Teorema 2. *Seja P uma projeção ortogonal. Então $\mathcal{N}(P) = Im(P)^{\perp}$*

Demonstração. Seja $x \in \mathcal{N}(P)$ e $y \in Im(P)$ de modo que existe x_1 com $P(x_1) = y$. Devemos ter:

$$\langle x, y\rangle = \langle x, P(x_1)\rangle = \langle P(x), x_1\rangle = 0.$$

Por outro lado, se $y \in Im(P)^{\perp}$ então $\langle P(x), y\rangle = 0$ para todo $P(x) \in Im(P)$. Assim, $\langle P(x), y\rangle = \langle x, P(y)\rangle = 0$ e, como $x \in \mathbb{R}^n$ é genérico, então $P(y) = 0$ e $y \in \mathcal{N}(P)$. □

É fato conhecido da Álgebra Linear que qualquer espaço vetorial Euclidiano, em particular o espaço $\mathbb{R}^n$, pode ser decomposto como uma soma direta $\mathbb{R}^n = U \oplus U^{\perp}$, aonde U é um subespaço qualquer de $\mathbb{R}^n$ e $U^{\perp}$ representa o complemento ortogonal de U. Nesse sentido, se P é uma projeção ortogonal então

$$\mathbb{R}^n = \mathcal{N}(P) \oplus Im(P)$$

Teorema 3. *Se P é uma projeção ortogonal, então:*

$$\mathcal{N}(P^{\mathsf{T}}) = Im(P)^{\perp} \text{ e } \mathcal{N}(P) = Im(P^{\mathsf{T}})^{\perp}$$

Demonstração. Seja $x \in \mathcal{N}(P^{\mathsf{T}})$ e considere um $y \in \mathbb{R}^n$. Então, $0 = \langle P^{\mathsf{T}}(x), y\rangle = \langle x, P(y)\rangle$ e, assim, $x \perp Im(P)$. Agora, se $x \in Im(P)^{\perp}$ então $0 = \langle x, P(y)\rangle = \langle P^{\mathsf{T}}(x), y\rangle$ e $x \in \mathcal{N}(P^{\mathsf{T}})$.

A segunda igualdade segue analogamente. □

Teorema 4. *Se P é uma projeção ortogonal então:*

$$\|x\|_2^2 = \|P(x)\|_2^2 + \|(I - P)(x)\|_2^2$$

Demonstração. De fato, segue imediatamente de $x = P(x) + (I - P)(x)$ e do fato de que $P(x) \perp (I - P)(x)$. □

Assim, é imediato verificar que $\|P(x)\|_2^2 \leqslant \|x\|_2^2$ e, consequentemente, $\dfrac{\|P(x)\|_2}{\|x\|_2} \leqslant 1$. Mais ainda, se $x \in Im(P)$ então $P(x) = x$ e $\dfrac{\|P(x)\|_2}{\|x\|_2} = 1$. Assim, $sup_{\|x\|_2 \neq 0} \dfrac{\|P(x)\|_2}{\|x\|_2} = 1$, de modo que $\|P\|_2 = 1$.

1.2 Método de Projeção

Considere o sistema (1) e dois subespaços $\mathcal{K}$ e $\mathcal{L}$ de $\mathbb{R}^n$. Queremos obter uma projeção oblíqua sobre $\mathcal{K}$ com direção ortogonal a $\mathcal{L}$ de tal modo que $\mathbb{R}^n = \mathcal{N}(P) \oplus Im(P)$.

O procedimento é o seguinte: considere um chute inicial x_0 para o sistema (1). Vamos procurar a solução $\tilde{x}$ no espaço $x_0 + \mathcal{K}$ tal que $b - A\tilde{x} \perp \mathcal{L}$.

Como $\tilde{x} \in x_0 + \mathcal{K}$, temos $\tilde{x} = x_0 + k$, onde $k \in \mathcal{K}$. O vetor resíduo inicial é dado por $r_0 = b - Ax_0$ e, de posse da solução exata, podemos observar que, $b - A\tilde{x} = b - A(x_0 + k) = r_0 - Ak$. Logo, queremos encontrar uma aproximação $\tilde{x} = x_0 + k$ tal que

$$\langle r_0 - Ak, w \rangle = 0 \tag{2}$$

para todo $w \in \mathcal{L}$. O vetor erro é medido por $e = xsol - x$, aonde $xsol$ é a solução do sistema (1).

Considere V e W matrizes cujas colunas representem bases de $\mathcal{K}$ e $\mathcal{L}$, respectivamente e suponha que $\mathcal{K}$ e $\mathcal{L}$ sejam de mesma dimensão m. Escreva $k = Vy$.

$$\begin{aligned}
\tilde{x} &= x_0 + Vy \\
Vy &= \tilde{x} - x_0 \\
AVy &= A(\tilde{x} - x_0) \\
AVy &= -(b - A\tilde{x}) + (b - Ax_0) \\
AVy &= r_0 - \tilde{r} \\
W^{\mathsf{T}}AVy &= W^{\mathsf{T}}(r_0 - \tilde{r}) \\
W^{\mathsf{T}}AVy &= W^{\mathsf{T}}r_0 \\
y &= (W^{\mathsf{T}}AV)^{-1}W^{\mathsf{T}}r_0
\end{aligned}$$

Logo, o processo fica assim descrito:

Algoritmo 1: Método da Projeção

```
Imput: matriz A, right-hand b, a stop criterion τ and an initial guess x;
Output: an approximate solution x̃. ;
Select K and L and their respectives bases V and W;
while ‖r‖ ⩽ τ do
    r ← b − Ax;
    y ← (W^T AV)^{-1} W^T r;
    x ← x + Vy;
```

O estudo mais completo feito em [69] nos permite uma melhor compreensão a respeito de como os métodos de projeção funcionem e quais são as condições necessárias para tal. Assim, por exemplo, o autor prova que se A é SPD e $\mathcal{K} = \mathcal{L}$ então $W^{\mathsf{T}}AV$ é inversível para quaisquer bases V e W. Por outro lado, se A é apenas não singular, mas $\mathcal{K} = A\mathcal{L}$ então também teremos $W^{\mathsf{T}}AV$ inversível.

Mais importante, se A é SPD e $\mathcal{K} = \mathcal{L}$ e se $\tilde{x}$ minimiza a A-norma do erro, isto é, $\tilde{x} = \min_{x \in x_0 + \mathcal{K}} \langle A(xsol - x), xsol - x\rangle^{\frac{1}{2}}$ então $\tilde{x}$ resulta de um método de projeção ortogonal sobre $\mathcal{K}$. Analogamente, se A é não singular e $\mathcal{K} = A\mathcal{L}$ e se $\tilde{x}$ minimiza a norma Euclidiana do resíduo, isto é $\tilde{x} = \min_{x \in x_0 + \mathcal{K}} \langle b - Ax, b - Ax\rangle^{\frac{1}{2}}$, então $\tilde{x}$ é um operador de projeção oblíqua sobre $\mathcal{K}$ com direção ortogonal a $\mathcal{K}$.

1.3 Espaços de Krylov

A partir de agora daremos início ao estudo dos Espaços de Krylov. Tais espaços tem relevância fundamental para nossos objetivos.

Definição 5. *Seja $A \in \mathbb{R}^{n\times n}$ e $b \in \mathbb{R}^n$. Chamamos de Espaço de Krylov associado à A e b ao espaço:*

$$\mathcal{K} = span\{b, Ab, A^2b, \ldots\}. \tag{3}$$

Já o espaço

$$\mathcal{K}_k = span\{b, Ab, A^2b, \ldots, A^{k-1}b\} \tag{4}$$

é chamado de subespaço de Krylov de dimensão k. Quando não houver ambiguidade usaremos apenas a notação $\mathcal{K}_k$.

Como $b \in \mathbb{R}^n$, então $A^j b \in \mathbb{R}^n$ para todo j inteiro. Assim, $dim(\mathcal{K}_j) \leqslant n$ e deve existir um primeiro k tal que $A^k b$ seja linearmente dependente dos vetores anteriores em

$\mathcal{K}_j$. Ou seja, existem $\alpha_i, 0 \leqslant i \leqslant k-1$ tais que:

$$A^k b = \sum_{i=0}^{k-1} \alpha_i A^i b$$

Definição 6. *O polinômio característico da matriz $A \in \mathbb{R}^{n\times n}$ é o polinômio*

$$p(t) = det(A - tI),$$

e pode ser escrito como $p(t) = \prod_{i=1}^{d}(t-\lambda_i)^{n_i}$, com $\lambda_i, 1 \leqslant i \leqslant d$ um autovalor de multiplicidade n_i e $\sum_{i=1}^{d} n_i = n$.

Definição 7. *O polinômio mínimo p de um vetor v com relação a matriz $A \in \mathbb{R}^{n\times n}$ é o polinômio mônico de menor grau tal que $p(A)v = 0$. O grau do polinômio p é chamado o grau v com relação a A.*

Agora, como $A^k b = \sum_{i=0}^{k-1} \alpha_i A^i b$ temos um polinômio com $p(A)b = 0$, a saber $p(t) = t^k - \sum_{i=0}^{k-1} \alpha_i t^i$.

Assim, podemos enunciar o seguinte Teorema:

Teorema 5. *Seja k o grau do polinômio mínimo de b com relação a A. Então, $dim(\mathcal{K}_j(A,b)) = min\{j,k\}$.*

Demonstração. Veja [55]. □

Vamos provar agora que a solução do sistema linear (1) vive num espaço de Krylov. De fato, seja k o grau do polinômio mínimo de A e escrevamos:

$$p(t) = \sum_{i=0}^{k} \alpha_i t^i;$$

Assim, como $p(A) = 0$ temos que:

$$\begin{aligned} A^{-1}p(A) &= A^{-1}\alpha_0 I + \alpha_1 A + \ldots + \alpha_k A^k = 0 \\ \alpha_0 A^{-1} &= -\textstyle\sum_{i=1}^{k-1} \alpha_i A^{i-1} \\ A^{-1} &= -\dfrac{1}{\alpha_0} \textstyle\sum_{i=1}^{k-1} \alpha_i A^{i-1} \end{aligned}$$

Assim, multiplicando ambos os lados por b e percebendo que a solução exata $xsol = A^{-1}b$ teremos: $xsol = A^{-1}b = -\frac{1}{\alpha_0}\sum_{i=1}^{k-1}\alpha_i A^{i-1}b$, de modo que $x \in \mathcal{K}_k(A,b)$.

Teorema 6. *Seja xsol a solução exata do sistema linear* (1) *para A não singular com polinômio mínimo de grau k. Então,*

$$xsol = x_0 + \mathcal{K}_k(A, r_0),$$

onde x_0 é um aproximação inicial de xsol.

Demonstração. Se $x_0 = 0$ então $\mathcal{K}_k(A,b) = \mathcal{K}_k(A,r_0)$ e podemos usar $\mathcal{K}_k(A,r_0)$. Suponha agora $x_0 \neq 0$. Assim, $b = b - Ax_0 + Ax_0 = Ax_0 + r_0$ e, portanto, $xsol = A^{-1}b = A^{-1}(Ax_0 + r_0) = x_0 + A^{-1}r_0$ e $xsol \in x_0 + \mathcal{K}_k(A,r_0)$. □

No tocante aos principais métodos de projeção, daremos enfoque em apenas dois deles: o GMRES (Método do Resíduo Mínimo Generalizado) e o CG (Método dos Gradientes Conjugados).

1.4 Método do Resíduo Mínimo Generalizado

Antes de discutirmos o GMRES, falaremos brevemente sobre outros dois antecessores: o Método de Arnoldi, o Algoritmo do Resíduo Conjugado Generalizado (GCR) e o Método da Ortogonalização Completa (FOM).

O método de Arnoldi foi inicialmente proposto em 1951, [5]. O método foi concebido para problemas de autovalores, e nos interessa, já que constrói uma base ortonormal $V = \{v_1, v_2, \ldots, v_m\}$ para o espaço de Krylov $\mathcal{K}_m(A, v_1)$.

Algoritmo 2: Método de Arnoldi

```
1 Choise a vector v_1 with ||v_1|| = 1;
2 for j ← 1 to m − 1 do
3 |   for i ← 1 to j do
4 |   |   h_ij = ⟨Av_j, v_i⟩
5 |   v̂_{j+1} = Av_j − Σ_{i=1}^{j} h_ij v_i;
6 |   h_{j+1,j} = ||v̂_{j+1}|| if h_{j+1,j} = 0 then end;
7 |   v_{j+1} = v̂_{j+1} / h_{j+1,j}
```

O resultado mais relevante sobre o método de Arnoldi é que $span\{v_1, \ldots, v_m\} = span\{v_1, Av_1, \ldots, A^{m-1}v_1\} = \mathcal{K}_m(A, v_1)$. Para uma prova completa, veja [63]. Nesse caso,

a matriz $V_m = [v_1, \ldots, v_m]$ satisfaz $AV_m = V_{m+1}H_m$, onde H_m é uma matriz de Hessenberg superior.

Vamos abordar agora o Método do Resíduo Conjugado Generalizado (GRC), que é equivalente ao GMRES em aritmética exata e foi proposto inicialmente em 1983, [36] e foi concebido para solução de sistemas lineares onde a matriz de coeficientes é Hermitiana, não simétrica e positivo definida.

Algoritmo 3: Resíduo Conjugado Generalizado (GRC)

```
1 Choise x_0 and calculate r_0 = b − Ax_0, p_0 = r_0;
2 for j ← 0, ..., until converge do
3     α_j = ⟨r_j, Ap_j⟩ / ⟨Ap_j, Ap_j⟩;
4     x_{j+1} = x_j + α_j p_j;
5     r_{j+1} = r_j − α_j Ap_j;
6     for i ← 1 to j do
7         β_ij = −⟨Ar_{j+1}, Ap_i⟩ / ⟨Ap_i, Ap_i⟩;
8     p_{j+1} = r_{j+1} + Σ_{i=0}^{j} β_ij p_i.
```

Sobre esse método, é possível provar que o conjunto de resíduos produzidos é um subespaço de Krylov, mais especificamente, $span\{p_0, \ldots, p_i\} = span\{r_0, \ldots, r_i\} = \mathcal{K}_i(A, p_0)$. Além disso, prova-se que x_{i+1} minimiza $E(w) = \|b - Aw\|_2$ sobre o espaço $x_0 + \mathcal{K}_i(A, p_0)$, [63].

Já o Método de Ortogonalização Completa (FOM) foi inicialmente proposta em 1981, [74]. Neste método, a cada iteração buscaremos $c_k \in \mathcal{K}_k(A, r_0)$, de modo que $x_k = x_0 + c_k$ e

$$b - Ax_k \perp \mathcal{K}_k, \tag{5}$$

a qual é chamada condição de Galerkin.

A explicação de que é necessário qe $c_k \in \mathcal{K}_k$ consiste no seguinte: vamos procurar uma solução $c_k \in \mathcal{K}_k$ para o sistema $Ac_k = r_0$. Assim, como $b - Ax_0 = r_0 = Ac_k$ o que nos dá que $b = A(c_k + x_0)$. Desse modo, encontrando tal c_k teremos resolvido o sistema original. Para esse objetivo começamos com o vetor $v_1 = \dfrac{r_0}{\|r_0\|_2}$ e cada iteração gera um novo $v_{k+1} \in \mathcal{K}_{k+1}$ através do processo de Arnoldi (veja o Algoritmo 2) obtendo assim uma base ortonormal para $\mathcal{K}_{k+1}(A, r_0)$. Segue que, como $c_k \in \mathcal{K}_k$ podemos escrever c_k como

$c_k = V_k y_k, y \in \mathbb{R}^k$ e, portanto:

$$AV_k y_k = V_{k+1} H_k y_k = r_0 \tag{6}$$

Logo,

$$H_k y_k = V_{k+1}^{\mathsf{T}} r_0, \tag{7}$$

e, como $\dfrac{r_0}{\|r_0\|_2} = v_1 = V_{k+1} e_1$, de modo que:

$$r_0 = \|r_0\| V_{k+1} e_1. \tag{8}$$

De acordo com a condição de Galerkin, (5), $b - Ax_k = b - A(x_0 + c_k) = r_0 - Ac_k \perp \mathcal{K}_k$. Logo, como $r_0 = \|r_0\| V_{k+1} e_1$ e $c_k = V_k y_k$ teremos $Ac_k = AV_k y_k = V_{k+1} H_k y_k$ para finalmente obtermos:

$$V_{k+1}(\|r_0\| e_1 - H_k y_k) \perp \mathcal{K}_k. \tag{9}$$

Se conseguirmos um y_k tal que $\|r_0\| e_1 - H_k y_k = 0$ e, consequentemente, $y_k = \|r_0\| H_k^{-1} e_1$ teremos o resultado desejado. Infelizmente, não há garantias de que H_k seja inversível, o que nos levaria a uma quebra. Caso a quebra não ocorra, $c_k = V_k y_k$.

Algoritmo 4: Método da Ortogonalização Completa (FOM)

1 Entradas: A, b, x_0;
2 $r_0 = b - Ax_0$;
3 $v_1 = \dfrac{r_0}{\|r_0\|_2}$;
4 **for** $j \leftarrow 1$**to**n **do**
5 $v_{j+1} = Av_j$;
6 **for** $j \leftarrow 1$**to**j **do**
7 $\eta_{ij} = \langle v_i, v_{j+1} \rangle$;
8 $v_{j+1} = v_{j+1} - \eta_{ij} v_i$;
9 $\eta_{j+1,j} = \|v_{j+1}\|_2$;
10 Se $\eta_{j+1,j} = 0$ go to 13;
11 $v_{j+1} = \dfrac{v_{j+1}}{\eta_{j+1,j}}$
12 $y = \|r_0\| H_k^{-1} e_1$;
13 $x = V_j y$.

O GMRES (Método do Resíduo Mínimo Generalizado) foi inicialmente por proposto em 1986, [71]. Nesse método, assim como no FOM busca-se uma correção c_k de modo que $x_k = x_0 + c_k$ e consequentemente $r_k = b - Ax_k = r_0 - Ac_k$. Mais precisamente,

procuraremos c_k tal que:

$$c_k = \arg\min_{c \in \mathcal{K}_k} \|r_0 - Ac\|_2.$$

Semelhantemente ao que foi feito no FOM teremos:

$$\min_{c \in \mathcal{K}_k} \|r_0 - Ac\|_2 = \min_{y \in \mathbb{R}^n} \|V_{k+1}(\|r_0\|e_1 - H_k y_k)\|_2$$

e, como V_{k+1} é unitário então $\min_{c \in \mathcal{K}_k} \|r_0 - Ac\|_2 = \min_{y \in \mathbb{R}^n} \|\|r_0\|e_1 - H_k y_k\|_2$. Com isso, o problema original recai sobre um problema de mínimos quadrados:

$$H_k y_k = \|r_0\|e_1.$$

Caso seja possível calcular uma decomposição QR econômica de H_k teremos $Q_k R_k y_k = \|r_0\|e_1$ e, portanto, $y_k = R_k^{-1} Q_k^{\mathsf{T}} \|r_0\|e_1$. Como Q_k é unitária e $Q^{\mathsf{T}} H_k = R_k$ teremos:

$$\min_{y \in \mathbb{R}^n} \|Q_k(\|r_0\|e_1 - H_k y_k)\|_2 = \min_{y \in \mathbb{R}^n} \|g_k - R_k y\|,$$

aonde $g_k = \|r_0\|Q_k e_1$, de modo que y é a solução do sistema $R_k y = g_k$.

De um modo geral, fatoração QR pode ser bem cara. Porém, nesse caso em particular, é possível calcular a fatoração com um custo baixo através das rotações de Givens. Assim, aplica-se k rotações de Givens na matriz H_k de modo que sua parte triangular inferior fique zerada, ou seja, $G_1 G_2 \dots G_k H_k = R_k$.

1.5 Método dos Gradientes Conjugados

O método dos Gradientes Conjugados foi publicado em 1952, [44] e tem grande importância histórica. O método foi concebido para sistemas lineares cuja matriz de coeficientes é simétrica e positivo definida.

Defina o erro e_k como $e_k = x_k - A^{-1}b$. Perceba que $Ae_k = Ax_k - b = -r_k$. Essa relação é essencial, visto que não dispomos de A^{-1}.

Suponha que existe um conjunto de vetores $p_0, p_1, \dots, p_n$ ortogonais entre si tal que $e_0 = x_0 - A^{-1}b = -\sum_{i=0}^{n-1} \alpha_i p_i$. Estes vetores serão chamados de vetores de busca.

Algoritmo 5: Método do Resíduo Mínimo Generalizado (GMRES)

A, b, x_0;
$r_0 \leftarrow b - Ax_0$;
$v_1 \leftarrow \dfrac{r_0}{\|r_0\|_2}$;
$j \leftarrow 1$ **while** $\|r_{j-1}\|_2 > tol$ **do**
    $w_j = Av_j$;
    for $j \leftarrow 1$ **to** j **do**
        $h_{ij} = \langle v_i, v_{j+1} \rangle$;
        $w_{j+1} = w_{j+1} - h_{ij} v_i$;
    $h_{j+1,j} = \|w_{j+1}\|_2$;
    $v_{j+1} = \dfrac{v_{j+1}}{h_{j+1,j}}$;
    Atualiza $g_j = G_j g_{j-1}$;
    $H_j y = \|r_0\| e_1$;
    $x_j = x_0 + V_j y$;
    $r_j = r_0 - V_{j+1} y$

Tais vetores deveriam ser construídos de tal modo que:

$$x_{k+1} = x_0 + \sum_{i=0}^{k} \alpha_i p_i = x_k + \alpha_k p_k. \tag{10}$$

O objetivo agora é encontrar tais vetores e seus respectivos coeficientes. Primeiramente,

$$\begin{array}{rcccl} e_k & = & x_k - A^{-1}b & = & \sum_{i=0}^{k-1} \alpha_i p_i + x_0 - A^{-1}b \\ & = & \sum_{i=0}^{k-1} \alpha_i p_i - \sum_{i=0}^{n} \alpha_i p_i & = & -\sum_{i=k}^{n} \alpha_i p_i. \end{array} \tag{11}$$

Assim, $e_k \perp p_j, j < k$, pois os vetores de busca são ortogonais entre si. Usando novamente a mesma técnica da equação (11) temos que:

$$e_{k+1} = e_k + \alpha_k p_k. \tag{12}$$

Assim, multiplicando ambos os lado da equação (12) por p_k teremos:

$$\begin{array}{rcl} 0 & = & \langle p_k, e_{k+1} \rangle \\ & = & \langle p_k, e_k \rangle + \alpha_k \langle p_k, p_k \rangle \\ \alpha_k & = & -\dfrac{\langle p_k, e_k \rangle}{\langle p_k, p_k \rangle}. \end{array} \tag{13}$$

Infelizmente, mais uma vez, o erro não está disponível. Assim, o que iremos exigir é a A-ortogonalidade. Pois desse modo, teremos:

$$\alpha_k = -\frac{\langle p_k, Ae_k\rangle}{\langle p_k, Ap_k\rangle} = \frac{\langle p_k, r_k\rangle}{\langle p_k, Ap_k\rangle}.$$

Note que desse modo, α_k pode ser calculado, já que: $r_{k+1} = -Ae_{k+1} = -A(e_k + \alpha_k p_k) = r_k - \alpha_k Ap_k = r_0 - \sum_{i=0}^{k} \alpha_i Ap_i$.

Falta agora encontrar as direções conjugadas. Para esse objetivo, vamos supor que:

$$p_k = r_k + \sum_{i=0}^{k-1} \beta_{k,i} p_i, \tag{14}$$

com $p_0 = r_0$.

Observe que para $j < k$ temos $\langle p_j, r_k\rangle = \langle p_j, \sum_{i=k}^{n-1} \alpha_i Ap_i\rangle = 0$, pois p_j é A-ortogonal a p_i para $i > j$. Assim, p_j e r_k são ortogonais. Mais que isso, os resíduos são mutuamente ortogonais, pois:

$$0 = \langle p_j, r_k\rangle = \langle r_k + \sum_{i=0}^{k-1} \beta_{k,i} p_i, r_k\rangle = \langle r_j, r_k\rangle + \sum_{i=0}^{k-1} \beta_{k,i}\langle p_i, r_k\rangle = \langle r_j, r_k\rangle.$$

Desse modo, como $\langle r_j, r_k\rangle = \langle r_k, r_j\rangle = 0$ temos o resultado desejado. Essas propriedades nos serão úteis adiante.

Multiplicando ambos os lados da equação (14) por $Ap_j, j < k$ teremos:

$$\begin{aligned} 0 \;=\; \langle p_k, Ap_j\rangle \;&=\; \langle r_k, Ap_j\rangle + \textstyle\sum_{i=0}^{k-1} \beta_{k,i}\langle p_i, Ap_j\rangle \\ &=\; \langle r_k, Ap_j\rangle + \beta_{k,j}\langle p_j, Ap_j\rangle \\ \beta_{k,j} \;&=\; -\frac{\langle r_k, Ap_j\rangle}{\langle p_j, Ap_j\rangle}. \end{aligned} \tag{15}$$

A partir de agora vamos simplificar as expressões que definem α e β: multiplicando ambos os lados de $r_{j+1} = r_j - \alpha_j Ap_j$ por r_k teremos: $\langle r_{j+1}, r_k\rangle = \langle r_j, r_k\rangle + \alpha_j\langle Ap_j, r_k\rangle$. Agora, se $j = k$ teremos

$$\alpha_k = \frac{\langle r_k, r_k\rangle - \langle r_{k+1}, r_k\rangle}{\langle Ap_k, r_k\rangle}.$$

Agora, como $\langle r_{k+1}, r_k \rangle = 0$ teremos:

$$\alpha_k = \frac{\|r_k\|_2^2}{\langle Ap_k, r_k \rangle}.$$

Finalmente, como $p_k = r_k + \beta_k p_{k-1}$ e p_k é A ortogonal a p_{k-1} temos $\langle Ap_k, r_k \rangle = \langle Ap_k, p_k \rangle$ e

$$\alpha_k = \frac{\|r_k\|_2^2}{\|p_k\|_A^2}.$$

Analogamente, obteremos:

$$\beta_k = \frac{\|r_k\|_2^2}{\|r_{k-1}\|_2^2}.$$

Algoritmo 6: Método dos Gradientes Conjugados (CG)

1 A, b, x_0;
2 $p_0 = r_0$;
3 **while** $\|r_i\|_2 > tol$ **do**
4 $\quad \alpha_i = \dfrac{\|r_i\|_2^2}{\|p_i\|_A^2}$;
5 $\quad x_{i+1} = x_i + \alpha_i p_i$;
6 $\quad r_{i+1} = r_i - \alpha_i A p_i$;
7 $\quad \beta_i = \dfrac{\|r_{i+1}\|_2^2}{\|r_i\|_2^2}$;
8 $\quad p_{i+1} = r_{i+1} + \beta_{i+1} p_i$.

2 ANÁLISE DE ARTIGOS

Vamos apresentar neste capítulo um histórico contendo o percurso do Algoritmo de Inversa Aproximada (AINV) proposto por Benzi, Meyer e Tüma de 1996 até os últimos resultados divulgados em 2016. Nesse percurso de 20 anos foram apresentadas diversas variações do AINV, que foram pensadas para diferentes famílias de matrizes: não singulares, simétricas e positivo definidas, não simétricas e positivo definidas, matrizes $\mathcal{M}$ e matrizes $\mathcal{H}$. Os focos teóricos dos trabalhos que discutiremos a seguir podem ser resumidos nos seguintes tópicos:

- Os autores procuraram determinar para quais matrizes os seus respectivos algoritmos eram livre de quebra;
- Para que efetivamente fosse calculada uma inversa aproximada, os autores procuraram qual seria a melhor estratégia para que vetores e matrizes envolvidas em cada iteração fossem esparsos. Tais estratégias eram chamadas de estratégias de descarte.
- Os autores procuraram reescalonamentos e reordenamentos que pudessem contribuir com os resultados computacionais.
- Verificar qual é a fatoração de A^{-1} obtida ou até mesmo de A.

Dessa forma, a nossa discussão sobre o conteúdo é voltada quase que em sua integralidade nos focos apresentados acima.

Uma procura particular nossa foi compreender em quais desses algoritmos a versão bloco foi devidamente explorada e nenhuma prova matemática foi apresentada nos raros trabalhos sobre o tema. Assim, faremos um tratamento matemático nos capítulos posteriores.

2.1 A sparse approximate inverse preconditioner for the conjugate gradient method. [11]

Em 1996, Benzi, Meyer e Tüma, em [11], apresentaram a versão escalar do AINV para matrizes simétricas e positivo definidas (SPD). A ideia é encontrar vetores $Z_i, i = 1, \ldots, n$ conjugados por A, aonde $A_{n\times n}$ é SPD utilizando um processo baseado em Gramm-Schmidt. Encontrados esses vetores forma-se a matriz Z tal que $Z^T AZ = D$ e, portanto, $A^{-1} = ZD^{-1}Z^T$, com D matriz diagonal. As entradas da matriz D são chamadas de pivôs

Algoritmo 7: AINV

1 $z_i^{(0)} \leftarrow e_i, i = 1, \ldots, n;;$
2 **for** $1 \leftarrow 1$ **to** n **do**
3 **for** $J \leftarrow i$ **to** n **do**
4 $p_j^{(i-1)} \leftarrow a_i^T z_j^{(i-1)}$
5 **if** $i \leftarrow n$ **then**
6 go to 11
7 **for** $j \leftarrow i+1$ **to** n **do**
8 $z_j^{(i)} \leftarrow z_j^{(i-1)} - \dfrac{p_j^{(i-1)}}{p_i^{(i-1)}} z_i^{(i-1)}$
9 $D \leftarrow Diag(p_1, p_2, ..., p_n)$ and $Z \leftarrow (z_1, z_2, ..., z_n)$

e são calculadas como $p_i = a_i^T z_i$, aonde a_i^T é uma linha da matriz A. Assim, o AINV calcula uma fatoração da inversa exata de A, o que é inviável para sistemas lineares de grande porte. A viabilidade do algoritmo então se dá na construção dos vetores z_i, quando é implementada um descarte de entradas, para inserir uma estrutura esparsa em Z.

As principais dificuldades advindas desse procedimento estão relacionadas aos pivôs: é possível que p_i deixe de ser positivo, podendo assim ser negativo (o que faz com o que o precondicionador gerado não seja apropriado para o método do Gradiente Conjugado) ou zero (o que pararia o processo).

No mesmo artigo, os autores concluem que o algoritmo é livre de quebra (ou seja, não permite pivôs iguais a zero ou negativos) para duas famílias de matrizes: as matrizes $\mathcal{M}$ e, mais geralmente para as matrizes $\mathcal{H}$.

Uma possibilidade de investigação é uma consequência importante do resultado acima: podemos realizar um "shift", ou seja, encontrar um α tal que $A' = A + \alpha I$ é diagonalmente dominante e toda matriz diagonalmente dominante é $\mathcal{H}$.

No caso geral, é possível encontrar matrizes com as quais o AINV quebra. Como veremos mais adiante, o BAINV possui mais estabilidade que AINV escalar. Como mostraremos a seguir, o Bloco AINV possui maior estabilidade que o AINV escalar proposto pelo artigo.

Quanto a implementação, os autores também propuseram uma modificação para evitar quebras: quando o pivô p_i for muito pequeno (menor que $\sqrt{\epsilon_M}$, onde ϵ_M é a precisão da máquina), deve ser trocado por $max\{\sqrt{\epsilon_M}, \mu\rho\theta\}$, com $\mu = 0.1$, ρ é o máximo dentre os $p_j^{(i-1)}, j \geq i$ e θ é a norma do infinito da coluna i de Z.

Quanto à performance, em um conjunto de testes com matrizes retiradas da Matrix Market, o AINV foi comparável ao IChol, tanto em número de iterações quanto em tempo de execução.

2.2 A sparse approximate inverse preconditioner for nonsymmetric linear systems. [12]

Em 1998, Benzi e Tüma, [12], os autores apresentaram a versão não simétrica do AINV. A ideia é encontrar duas matrizes W e Z tais que $W^TAZ = D$ e, então, $A^{-1} = ZD^{-1}W^T$, com um processo conhecido como Biconjugação. O algoritmo foi essencialmente proposto por Fox, [38].

No Algoritmo 8 abaixo, c_i^T representa a i-ésima coluna de A^T.

Algoritmo 8: NS AINV

1 $w_i^{(0)} = z_i^{(0)} \leftarrow e_i, i = 1, \ldots, n;$
2 **for** $i \leftarrow 1$ **to** n **do**
3 **for** $j \leftarrow 1$ **to** n **do**
4 $p_j^{(i-1)} \leftarrow a_i^T z_j^{(i-1)};$
5 $q_j^{(i-1)} \leftarrow c_i^T w_j^{(i-1)};$
6 **if** $i \leftarrow n$ **then**
7 go to 9
8 **for** $j \leftarrow i+1$ **to** n **do**
9 $z_j^{(i)} \leftarrow z_j^{(i-1)} - \dfrac{p_j^{(i-1)}}{p_i^{(i-1)}} z_i^{(i-1)};$
10 $w_j^{(i)} \leftarrow w_j^{(i-1)} - \dfrac{q_j^{(i-1)}}{q_i^{(i-1)}} w_i^{(i-1)};$
11 $D \leftarrow Diag(p_1, p_2, ..., p_n)$, $Z \leftarrow (z_1, z_2, ..., z_n)$ and $W \leftarrow (w_1, w_2, ..., w_n)$.

Uma observação essencial aqui é que, em aritmética exata:

$$p_i = w_i^T A z_i = z_i^T A^T w_i = q_i$$

Queremos chamar atenção para o seguinte fato importante: os pivôs p_i podem ser escritos como razão dos menores principais de A:

$$p_i = \frac{\Delta_i}{\Delta_{i-1}}, (i = 1, \ldots, n)$$

aonde Δ_i denota o menor principal de ordem i de A e $\Delta_0 = 1$. Portanto, o Algoritmo

não quebra se e somente se todos os menores principais são não nulos. Esse fato será generalizado para o contexto bloco mais adiante.

No texto os autores concluíram que o Algoritmo 8 com descarte é livre de quebras para matrizes $\mathcal{H}$, mas que pode produzir um precondicionar G tal que $I - GA$ tenha raio espectral maior do que 1 (ao contrário do ILU), mesmo no caso simétrico de matrizes $\mathcal{M}$, o que compromete a performance tal qual é o caso da matriz:

$$\begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{pmatrix},$$

que, aplicado ao AINV, com descarte por tolerância de 0.5 produz G com $\rho(I - GA) = 1.2153$. Um resultado também importante é que se A é uma $\mathcal{M}$ matriz e $G = \widetilde{Z}\widetilde{D}^{-1}\widetilde{W}^T \approx A^{-1}$ teremos:

$$D_A \leq G \leq A^{-1},$$

ou mais geralmente, se G_1 e G_2 são duas aproximações da inversa de A de tal maneira que o descarte usado em G_1 seja maior que o descarte usado em G_2 então

$$D_A \leq G_1 \leq G_2 \leq A^{-1}.$$

Os experimentos numéricos se deram nos métodos QMR, Bi-CGSTAB e o GMRES com restarte de 20 passos. Baseados em muitos testes numéricos, os autores concluíram que o precondicionador pode ser usado para enriquecer muitos solvers diferentes e que o raio de convergência é comparável, em média, com o ILU(0).

2.3 Robust Approximate Inverse Preconditioning for the Conjugate Gradient Method. [14]

Em 2000, Benzi, Cullum e Tüma, [14], apresentaram uma versão livre de quebra e estabilizada do AINV para matrizes SPD: o SAINV.

Nessa versão, a formulação de $p_i = a_i^T z_i$ é trocada por $p_i = z_i^T A z_i$, que é feita em dois passos: $v_i = Az_i$ e $p_i = v_i^T z_i$. Como A é SPD, $p_i > 0$, mesmo que z_i seja alterado (não anulado totalmente). Quando não há descarte, essas duas formulações são equivalentes,

Algoritmo 9: SAINV

1 $z_i^{(0)} \leftarrow e_i, i = 1, ..., n;$
2 **for** $i \leftarrow 1$ **to** n **do**
3 $\quad v_i \leftarrow Az_i^{(i-1)};$
4 $\quad$ **for** $j \leftarrow i$ **to** n **do**
5 $\quad\quad p_j^{(i-1)} \leftarrow v_i^T z_j^{(i-1)};$
6 $\quad$ **if** $i \leftarrow n$ **then**
7 $\quad\quad$ go to 10
8 $\quad$ **for** $j \leftarrow i+1$ **to** n **do**
9 $\quad\quad z_j^{(i)} \leftarrow z_j^{(i-1)} - \dfrac{p_j^{(i-1)}}{p_i^{(i-1)}} z_i^{(i-1)};$
10 $D \leftarrow diag(p_1, p_2, ..., p_n)$ and $Z \leftarrow (z_1, z_2, ..., z_n)$.

como mostraremos mais adiante. Os autores respondem a uma pergunta central: Qual é o custo dessa nova forma de calcular os pivôs? Num primeiro olhar, parece que o Algoritmo 9 requer n produtos matriz-vetor. Para ver que isso não é verdade, os autores argumentam os vetores z_i são mantidos esparsos, de modo que os produtos matriz-vetor são efetuados no modo esparso-esparso. Assim, v_i é formado a partir de poucas colunas de A (aquelas que correspondem aos não-zeros de z_i) e, adicionalmente, como, z_i é triangular superior, somente as i primeiras colunas de A entram no cálculo. Portanto, o custo é de ordem n do sistema.

Os autores apresentam também um método, devido a Axelsson e Kolotilina, [6] e [7], que encontra uma matriz $\mathcal{M}$ que possui uma relação direta com uma dada matriz SPD A. A ideia é dividir A em duas partes: $A = B + R$, aonde R contém todas as entradas positivas fora da diagonal de A e tomar a matriz $\hat{A} = B + \Delta$, aonde Δ é a matriz diagonal que satifaz $\Delta e = Re$, com e vetor só de 1's. Assim, $\hat{A} = A + (\Delta - R)$ é uma matriz associada a matriz A que é $\mathcal{M}$. O AINV clássico pode ser perfeitamente aplicado a $\hat{A}$ sem quebras.

Quanto a performance, os autores recomendam o uso de um rescalonamento da matriz, como o rescalonamento pela diagonal (Jacobi), e o uso de um reordenamento da matriz com MMD (Minimum Multiple Degree). Atualmente, o Matlab recomenda o uso do AMD (Approximate Minimum Degree), [4]. Os autores concluem que o SAINV é um precondicionador confiável, tanto em número de iterações quanto em tempo de precondicionamento e que não precisa de uma forma esparsa pré-estabelecida e, sim,

de uma tolerância para as entradas da matriz Z, tipicamente, 10^{-1}. Os experimentos mostraram que o SAINV é superior a outros precondicionadores paralelizáveis tais como o de Jacobi, o precondicionador polinômio de mínimos quadrados e o FSAI.

2.4 Refining an Approximate Inverse. [22]

Ainda em 2000, Bridson e Tang, [22] apresentaram o trabalho "Refining an Approximate Inverse". Os autores apresentam uma versão não simétrica para o SAINV (que não é livre de quebras).

Algoritmo 10: SAINV-NS

1 $w_i^{(0)} = z_i^{(0)} \leftarrow e_i, i = 1, \ldots, n$;
2 **for** $j \leftarrow 1$ **to** n **do**
3 $\quad l \leftarrow Az_j$;
4 $\quad u \leftarrow w_j^T A$;
5 $\quad d_{jj} \leftarrow uz_j$ or $w_j^T l$, whichever is cheapest;
6 $\quad$ **for** $i > j$ *and* $l_i \neq 0$ **do**
7 $\quad\quad w_i \leftarrow w_i - drop(\frac{l_i}{d_{jj}} w_j)$;
8 $\quad$ **for** $i > j$ *and* $u_i \neq 0$ **do**
9 $\quad\quad z_i \leftarrow z_i - drop(\frac{u_i}{d_{jj}} z_j)$;
10 Return D, Z and W

A fatoração que se pretende obter é $A^{-1} = ZD^{-1}W^{\mathsf{T}}$. Nós devemos observar que o descarte é suavemente diferente do praticado no NS-AINV. No Algoritmo 8 primeiro avaliamos o valor de $|z_i|$ e depois o descartamos. Aqui, primeiro avaliamos $\frac{u_i}{d_{jj}} z_j$ para saber se devemos atualizar z_i.

No mesmo os autores propõem uma versão em bloco, não simétrica para o SAINV, que foi mais lenta que o SAINV na maior parte dos exemplos:

2.5 Stabilized and Block Approximate Inverse Preconditioners for Problens in Solid and Structural Mechanics, [10]

Em 2001, Benzi, Kouhia e Tůma apresentaram o trabalho "Stabilized and Block Approximate Inverse Preconditioners for Problens in Solid and Structural Mechanics", [10]. No texto é apresentada uma a versão simétrica em bloco do SAINV que é, naturalmente, livre de quebras para matrizes Simétricas e Positivo Definidas. Os autores

Algoritmo 11: SBAINV-NS

$W_i^{(0)} = Z_i^{(0)} \leftarrow E_i$;
Get block row i of A: R;
for $j < i, U_{ji} \neq 0$ *(determined by symbolic factorization)* **do**
  $W_i \leftarrow W_i - W_j(RZ_jD_{jj}^{-1})^T$;
;
Get block column i of A: C **for** $j < i, L_{ji} \neq 0$ *(determined by symbolic factorization)* **do**
  $Z_i \leftarrow Z_i - W_j(D_{jj}^{-1}W_jC)$;
Set $D_{ii} = W_i^T AZ_i$. Return D, Z and W.

recomendam ainda o uso de um rescalonamento de Bloco Jacobi, que calcula os fatores de Cholesky dos blocos diagonais de A, isto é, $A_{ii} = L_iL_i^T$ e faz $\hat{A} = G^{-1}AG^{-T}$, aonde $G = diag(L_1.L_2, \ldots, L_N)$.

O bloco A_{ij} tem ordem $n_i \times n_j$ e a i-ésima linha de A é denotada por $A_i^T = [A_{i1}\ A_{i2}\ \ldots A_{iN}]$.

Algoritmo 12: SBAINV

$Z_i^{(0)} \leftarrow E_i, i = 1, \ldots, n$;
for $i \leftarrow 1$ **to** n **do**
  $V_i \leftarrow AZ_i^{(i-1)}$;
  for $j \leftarrow i$ **to** n **do**
    $P_j^{(i-1)} \leftarrow V_i^T Z_j^{(i-1)}$;
  if $i = n$ **then**
    go to 10
  for $j \leftarrow i + 1$ **to** n **do**
    $Z_j^{(i)} \leftarrow Z_j^{(i-1)} - Z_i^{(i-1)}(P_i^{(i-1)})^{-1}P_j^{(i-1)}$;
$D \leftarrow Diag(P_1, P_2, ..., P_n)$ and $Z \leftarrow (Z_1, Z_2, ..., Z_n)$.

Os autores salientam que a estrutura em blocos pode surgir, tanto naturalmente, da discretização do problema real, como por exemplo, quando usamos o método dos elementos finitos para discretizar uma equação diferencial parcial, quanto artificialmente, caso melhore o desempenho. Os autores consideram ainda dois tipos de descarte: o absoluto, quando uma entrada é descartada se é menor que um parâmetro pré-estabelecido e o descarte relativo, quando uma entrada é descartada se é menor que um parâmetro multiplicado pela norma da linha de A fixada na iteração.

Primeiramente, os autores comparam o AINV escalar com descarte de preenchimento zero (nesse caso o preenchimento de Z deve ser igual ao preenchimento da parte

triangular superior de A) com o BAINV sem rescalonamento, com rescalonamento de Jacobi e com rescalonamento de Bloco Jacobi e concluem que, essa ordem vai do pior precondicionador para o melhor no caso da matriz S3RMQ4M1. Na sequência, o SBAINV é comparado a outros precondicionadores como o Ichol, com o Ichol de Ajiz-Jennings em blocos, [1], e com o SAINV. Os autores concluem que o SAINV é um precondicionador atrativo e que o SBAINV tem uma convergência suavemente mais lenta que versão escalar, mas que pode ser mais eficiente que o confiável precondicionador de Ajiz-Jennings.

2.6 A Robust Preconditioner with Low Memory Requirements for Large Sparse Least Squares Problems. [15]

Em 2003 foi a vez de Benzi e Tüma apresentarem o trabalho "A Robust Preconditioner with Low Memory Requirements for Large Sparse Least Squares Problems", [15].

A proposta dos autores foi resolver um problema de mínimos quadrados da forma

$$\min_{x \in \mathbf{R}^n} \|b - Ax\|,$$

onde A é grande e esparsa, de ordem $m \times n$, com posto completo. Naturalmente, as técnicas dos autores são aplicáveis no caso $m = n$, mas o caso de maior interesse é quando $m > n$. O sistema modificado é $Cx = f, C = A^T A, f = A^T b$. Métodos diretos esparsos são confiáveis, mas podem ser caros quanto ao armazenamento e número de operações. A alternativa são métodos iterativos. O algoritmo que será utilizado pelos autores é CGLS, o que significa que precisamos de um bom precondicionador. Nesse trabalho, os autores apresentaram o RIF(Robust Incomplete Factorization), que encontra um precondicionador para o método dos mínimos quadrados usando C-ortogonalização. Como $C = A^T A$ é simétrico e positivo definido, já que A tem posto completo, a maneira como isso será feita é através do SAINV. O desejo é que esse precondicionador tenha as seguintes propriedades:

- Nenhuma entrada de $C = A^T A$ precise ser explicitamente calculada.
- A fatoração incompleta não quebre;
- O armazenamento intermediário seja insignificante.

A primeira propriedade é garantida pelo seguinte fato: o SAINV calcula vetores z_i e z_j tais que $z_i^T C z_j = 0$. Assim, $0 = z_i^T C z_j = z_i^T A^T A z_j = (Az_i)^T A z_j$ e $A^T A$ não precisa ser explicitamente calculada. O segundo fato é garantido pelo SAINV já que C é matriz simétrica e positivo definida.

Os autores comentam também que pode ser vantajoso escalar as linhas de A para $\|a\|_2 = 1$ (isto é, C tem diagonal unitária), pois isso tende a melhorar o condicionamento das equações normais e ajuda a utilizar uma tolerância de descarte absoluta τ. Um comentário importante dos autores é que, tipicamente, o produto interno de AZ_i com AZ_j é zero (eles não possuem entradas não nulas nas mesmas posições) e que a implementação deve fazer uso de informação estrutural para evitar checar quais produtos são zero, o que corresponderia a $\mathcal{O}(n^2)$ operações.

Duas variações podem ser obtidas: os vetores $\widetilde{Z}_i$ são armazenados, que são as colunas do fator Z, já os multiplicadores $\langle z_i, z_j\rangle_C / \langle z_j, z_j\rangle_C$ são descartados depois de usados.

Um segundo algoritmo é proposto quando escolhemos armazenar os multiplicadores e descartar os vetores Z_i assim que são usados para formar as partes correspondentes do fator de Cholesky $L = [l_{ij}]$ de C. Deste modo, o algoritmo calcula uma fatoração incompleta de Cholesky sem cálculo de raízes quadradas $C = A^T A \approx \widetilde{L}\widetilde{D}\widetilde{L}$. Note que $\widetilde{d}_j = \widetilde{z}_j C \widetilde{z}_j > 0$, pois C é matriz SPD e o algoritmo também é livre de quebra e, por isso, é chamado de RIF (Robust Incomplete Factorization).

Nesse ponto há uma observação importante, não há versões em bloco dos métodos descritos acima, podendo ser objeto de pesquisa. Porém, o RIF, possui um concorrente de peso: O BIF (Balanced Incomplete Factorization), [23, 25, 26]. Nessa versão, é utilizada a fórmula de Shermonn-Morrison para criar a fatoração incompleta.

2.7 An Improvement of Sainv and Rif Precondtionings of CG Method by Double Dropping Strategy. [39]

Em 2004, foi a vez de Seiji Fujino e Yusuke Ikeda, [39] apresentarem sua versão do SAINV: o ISAINV, aonde esse I se refere a improved.

Essa versão utiliza uma estratégia diferente de descarte: além do descarte por tolerância da magnitude de uma entrada de Z, antes a magnitude de $p_j^{(i-1)}/p_i^{(i-1)}$ é avaliada. Caso essa magnitude seja menor que uma tolerância previamente definida (e possivelmente

Algoritmo 13: ISAINV

```
1  z_i^(0) ← e_i, i = 1, ..., n;
2  for i ← 1 to n do
3  |  v_i ← A z_i^(i−1);
4  |  for j ← i to n do
5  |  |  p_j^(i−1) ← v_i^T z_j^(i−1);
6  |  if i = n then
7  |  |  go to 13
8  |  for j ← i + 1 to n do
9  |  |  if |d_j/d_i| > tol − 2 then
10 |  |  |  for k ← 1 to i do
11 |  |  |  |  if |z_jk^(i−1) − (p_j^(i−1)/p_i^(i−1)) z_ik^(i−1)| > tol − 1 then
12 |  |  |  |  |  z_j^(i) ← z_j^(i−1) − (p_j^(i−1)/p_i^(i−1)) z_i^(i−1);
13 D ← Diag(p_1, p_2, ..., p_n) and Z ← (z_1, z_2, ..., z_n).
```

diferente da tolerância nas entradas de Z), a coluna de Z_j não será atualizada na i-ésima iteração. Os autores também propôem que essa mesma estratégia pode ser usada no RIF, gerando o IRIF. Os testes indicaram que o IRIF é um precondicionador confiável e que, em geral, produz uma densidade (razão entre não zeros de Z por não zeros de A) muito baixa comparado aos demais precondicionadores, a saber o Sainv, o Isainv e o próprio RIF. Os autores concluem que o número de iterações e o tempo total (precondicionamento mais aplicação) podem ser significativamente menores. No entanto, os autores comentam que o IRIF não é tão efetivo para problemas advindos do Métodos dos Elementos Finitos (FEM, na sigla em inglês) aplicados a problemas de Mecânica dos Sólidos e que, portanto, há espaço para trabalhos futuros.

Nesse sentido, há espaço para uma futura implementação do ISBAINV. Já que essa proposta de descarte não é utilizada no contexto de matrizes em bloco.

2.8 New breakdown-free variant of AINV method for nonsymmetric positive definite matrices. [66]

Em 2008, Rafiei e Toutounian apresentaram o trabalho New breakdown-free variant of AINV method for nonsymmetric positive definite matrices. O objetivo é apresentar

uma variante do AINV para matrizes não simétricas e positivo definidas (NSPD). Os autores utilizam uma forma recursiva para obter os pivôs e mostram que não é preciso utilizar a biconjugação. Eles demonstram que $p_j^{(i-1)} = w^T A e_j$ e, assim conseguem provar mais geralmente que:

$$p_j^{(i-1)} = a_{ij} - \sum_{k=1}^{i-1} \frac{p_j^{(k-1)}}{p_k^{(k-1)}} q_i^{(i-1)}, 1 \leq i \leq j \leq n.$$

Analogamente,

$$q_j^{(i-1)} = a_{ji} - \sum_{k=1}^{i-1} \frac{q_j^{(k-1)}}{q_k^{(k-1)}} p_i^{(i-1)}, 1 \leq i \leq j \leq n.$$

Um comentário importante, essas relações aqui, são ajustáveis para o caso bloco. Portanto, o algoritmo proposto posteriormente é, também, ajustável para blocos.

Usando as relações acima e o fato de que $A^{-1} = ZD^{-1}W^{\mathsf{T}}$ e, consequentemente, $Z^{-1} = U = D^{-1}W^T A$, aonde U é o fator triangular superior da fatoração LDU, conclui-se que $u_{ij} = \dfrac{p_j^{(i-1)}}{p_i^{(i-1)}}$. Assim, como $p_j^{(i-1)}$ pode ser calculado sem o conhecimento de Z, podemos calcular a matriz U conhecendo apenas W.

A fatoração desejada é $A^{-1} = U^{-1}D^{-1}W^{\mathsf{T}}$. Agora, para calcular U^{-1} usaremos a série de Newmann com truncamento de quarta ordem:

$$U^{-1} \approx I + (I - U) + (I - U)^2 + (I - U)^3 + (I - U)^4.$$

O Algoritmo 14 descreve esta implementação.

Quantos aos experimentos, o conjunto de testes mostrou que o precondicionador é efetivo em reduzir o custo de sua construção, além de reduzir a densidade do precondicionador. Como os próprios autores dizem:

> Numerical experiments show that the new preconditioner is effective at reducing the number of iterations and is sparser and cheaper (in terms of preconditioning costs and total costs) than the well-known AINV-A preconditioner. Also we can conclude that SAINV-NS (SAINV-VAR) is a robust and effective preconditioner for iterative methods to obtain the solution of large and sparse nonsymmetric systems of linear equations.

O algoritmo AINV-A foi desenvolvido por Kharchenko e outros, e nada mais é que

Algoritmo 14: SAINV-VAR

$w_i^{(0)} \leftarrow e_i, i = 1, ..., n;$
for $i \leftarrow 1$ **to** n **do**
  $v_i \leftarrow Aw_i^{(i-1)};$
  $q_i^{(i-1)} \leftarrow (w_i^{(i-1)})^T v_i;$
  $p_i^{(i-1)} \leftarrow q_i^{(i-1)};$
  if $i = n$ **then**
    go to 13
  for $j \leftarrow i + 1$ **to** n **do**
    $q_j^{(i-1)} \leftarrow (w_j^{(i-1)})^T v_i;$
    $p_j^{(i-1)} \leftarrow a_i j - \sum_{k=1}^{i-1} u_k j q_i^{(i-1)};$
    $u_{ij} \leftarrow \dfrac{p_j^{(i-1)}}{p_i^{(i-1)}};$
    $w_j^{(i)} \leftarrow w_j^{(i-1)} - \dfrac{q_j^{(i-1)}}{q_i^{(i-1)}} w_i^{(i-1)};$
$w_i \leftarrow w_i^{(i-1)}$ and $q_i \leftarrow q_i^{(i-1)}$. Return $W \leftarrow [w_1, ..., w_n], U \leftarrow [U_{ij}]_{ij}$ and $D \leftarrow diag(p_1, ..., p_n)$.

o SAINV, feito de modo independente em [51].

Há um caminho muito natural de se propor este resultado para o caso bloco. E isto pode ser tema de investigação.

2.9 A Sparse Approximate Inverse Preconditioner for Nonsymmetric Positive Definite Matrices. [75]

Em 2010, Salkuyeh, [75], apresentou o trabalho "A Sparse Approximate Inverse Preconditioner for Nonsymmetric Positive Definite Matrices", na qual propõe um método baseado no FFAPINV, de 2002, [82], só que para matrizes não simétricas e positivo definidas. O texto de Lee e Zhang mostra que o FFAPINV está bem definido para matrizes $\mathcal{M}$. Já a versão de Salkuyeh, pode ser aplicada a qualquer matriz NSPD. Salkuyeh também mostrou em um trabalho anterior que, sem descarte, o AINV e o FFAPINV são matematicamente equivalentes, [76].

Vamos rever o FFAPINV: queremos encontrar matrizes W e Z tais que $WAZ = D$, com D matriz diagonal. Para tal, as linhas da matriz W e as colunas Z são construídas recursivamente, a partir da identidade e das colunas previamente construídas, isto é,

$z_1 = e_1$ e:

$$z_j = e_j + \sum_{i=1}^{j-1} \alpha_i z_i, j = 2, \ldots, n.$$

Analogamente,

$$w_j = e_j^T + \sum_{i=1}^{j-1} \beta_i w_i, j = 2, \ldots, n.$$

Assim, primeiramente, observe que $d_1 = w_i A z_i = a_{11}$. Agora, para $k = 1, \ldots, j-1$ temos

$$\begin{aligned} 0 &= w_k A z_j \\ &= w_k A e_j + \textstyle\sum_{i=1}^{j-1} \alpha_i w_k A z_i \\ &= w_k A_{*j} + \alpha_k w_k A z_k \\ &= w_k A_{*j} + \alpha_k d_k \end{aligned}$$

Assim,

$$\alpha_k = -\frac{w_k A_{*j}}{d_k}, k = 1, \ldots, j-1.$$

Do mesmo modo, obtemos que:

$$\beta_k = -\frac{A_{j*} z_k}{d_k}, k = 1, \ldots, j-1.$$

Algoritmo 15: FFAPINV

```
1  z_1^(0) ← e_1; w_1 = e_1^T, i = 1, ..., n;
2  for j ← 2 to n do
3      z_j ← e_j, w_j ← e_j^T;
4      for i ← 1 to j − 1 do
5          α_k ← −(w_k A_*j)/d_k;
6          β_k ← −(A_j* z_k)/d_k;
7          if |α_k| > τ then
8              z_j ← e_j + α_i z_i;
9          if |β_k| > τ then
10             w_j ← e_j^T + β_i w_i;
11         Drop entries of zj and wj whose absolute values are smaller than τ
12     d_j ← w_j A z_j
```

Devemos observar que o descarte é avaliado nos multiplicadores $z\alpha$ e β antes de

atualizar as colunas de Z e W.

Afim de construir o FFAPINV para matrizes não Simétricas e positivo xefinidas (NSPD), precisamos primeiramente observar que $d_j = z_j^T A z_j = w_j A w_j^T$ para $j = 1, \ldots, n$. De fato, como $WAZ = D$ segue que $Z^T AZ = Z^T W^{-1} D$. Assim, como $Z^T W^{-1} D$ é triangular inferior e $diag(Z^T W^{-1} D) = D$, podemos concluir que $diag(Z^T AZ) = D$ e, portanto, $d_j = z_j^T A z_j$. A segunda igualdade demonstra-se de modo análogo. Dessa discussão temos que $d_j > 0$ para matrizes NSPD.

Algoritmo 16: FFAPINV-NSPD

```
z_i^(0) ← e_i; w_1 ← e_1^T, i = 1, ..., n;
for j ← 2 to n do
    z_j ← e_j, w_j ← e_j^T;
    for i ← 1 to j − 1 do
        α_k ← −(w_k A_{*j}) / d_k;
        β_k ← −(A_{j*} z_k) / d_k;
        if |α_k| > τ then
            z_j ← e_j + α_i z_i;
        if |β_k| > τ then
            w_j ← e_j^T + β_i w_i;
        Drop entries of zj and wj whose absolute values are smaller than τ
    d_j ← A_{j*} z_k;
    if d_j = 0 then
        d_j ← w_j A z_j
```

Quantos aos experimentos, os autores puderam concluir que FFAPINV-NSPD é muito efetivo e melhora tanto o número de iterações quanto o tempo de convergência da CPU. Os testes indicaram ainda que FFAPINV-NSPD se mostrou melhor que o SAINV-NS.

2.10 Robust Incomplete Factorization for Nonsymmetric Matrices. [67]

Em 2011 Rafiei e Bollhöfer apresentaram o trabalho "Robust incomplete factorization for nonsymmetric matrices". Nele, os autores propuseram o RIF-NS(sigla do algoritmo que dá nome ao artigo). O algoritmo faz biconjugação e um dos focos principais são as estratégias de descarte propostas, que empregam as técnicas que o próprio Bollhöfer utiliza em trabalhos anteriores, [18], [19], [21].

A versão "right-looking" é apresentada no Algoritmo 17.

Algoritmo 17: Right-looking AINV with extraction of L and U

$z_i^{(0)} \leftarrow e_i; w_1 \leftarrow e_1, i = 1, ..., n;$
for $i \leftarrow 1$ **to** n **do**
  $v_i \leftarrow Ae_i$, $u_i \leftarrow A^T e_i$ {not positive definite};
  $v_i \leftarrow Aw_i^{(i-1)}$, $u_i \leftarrow Az_i^{(i-1)}$ {positive definite};
  $q_i^{(i-1)} \leftarrow (w_i^{(i-1)})^T v_i$, $p_i^{(i-1)} \leftarrow (z_i^{(i-1)})^T u_i$;
  $d_{ii} \leftarrow q_i^{(i-1)}$ or $p_i^{(i-1)}$;
  for $j \leftarrow i+1$ **to** n **do**
    $q_j^{(i-1)} \leftarrow (w_j^{(i-1)})^T v_i$, $p_j^{(i-1)} \leftarrow (z_j^{(i-1)})^T u_i$;
    $L_{ji} \leftarrow \dfrac{q_j^{(i-1)}}{q_i^{(i-1)}}$, $U_{ij} \leftarrow \dfrac{p_j^{(i-1)}}{p_i^{(i-1)}}$;
    apply a dropping rule to L_{ji} and U_{ij};
    $w_j^{(i)} \leftarrow w_j^{(i-1)} - \dfrac{q_j^{(i-1)}}{q_i^{(i-1)}} w_i^{(i-1)}$, $z_j^{(i)} \leftarrow z_j^{(i-1)} - \dfrac{p_j^{(i-1)}}{p_i^{(i-1)}} z_i^{(i-1)}$;
    for all $l \leq i$ apply a dropping rule to $w_{lj}^{(i)}$ and $z_{lj}^{(i)}$
Let $z_i \leftarrow z_i^{(i-1)}$ and $w_i \leftarrow w_i^{(i-1)}$ for $1 \leq i \leq n$;
$Z \leftarrow [z_1, \ldots, z_n], W \leftarrow [w_1, \ldots, w_n]$ and $D \leftarrow diag[d_{ii}]$.

A estratégia de descarte aqui tem um destaque especial: o uso do critério natural, ou seja, o uso de dois parâmetros ϵ_Z e ϵ_W tais que

$$|w_{lj}^{(i)}| < \epsilon_W, |z_{lj}^{(i)}| < \epsilon_Z,$$

para $1 \leq l < i < j \leq n$ é feito na maioria dos textos. Porém, se nenhuma descarte é feito em L_{ij} e U_{ij}, então as matrizes L e U tendem a ser densas. No texto de Benzi et al., [15], isso é realizado com filtragem posterior, que, na prática é utilizar outros dois parâmetros ϵ_L e ϵ_U de modo que são descartados os termos tais que

$$|L_{ji}| < \epsilon_L, |U_{ij}| < \epsilon_U.$$

Rafiei e Böllhofer sugerem usar então usar o mesmo parâmetro $\epsilon_{L,W}$ para L e W, bem como $\epsilon_{U,Z}$ para U e Z que satisfazem as seguintes relações:

$$|w_{lj}^{(i)}| < \epsilon_{L,W}, |z_{lj}^{(i)}| < \epsilon_{U,Z},$$

e também

$$|L_{ji}|\|W_{:,i}\|_\infty < \epsilon_{L,W}, |U_{ij}|\|Z_{:,i}\|_\infty < \epsilon_{U,Z}.$$

Rafiei e Böllhofer sugerem usar então usar o mesmo parâmetro $\epsilon_{L,W}$ para L e W, bem como $\epsilon_{U,Z}$ para U e Z que satisfazem as seguintes relações:

$$|w_{lj}^{(i)}| < \epsilon_{L,W}, |z_{lj}^{(i)}| < \epsilon_{U,Z},$$

e também

$$|L_{ji}|\|W_{:,i}\|_\infty < \epsilon_{L,W}, |U_{ij}|\|Z_{:,i}\|_\infty < \epsilon_{U,Z}.$$

Uma outra formulação usa, novamente a relação

$$p_j^{(i-1)} = a_{ij} - \sum_{k=1}^{i-1} \frac{p_j^{(k-1)}}{p_k^{(k-1)}} q_i^{(j-1)}, 1 \leq i \leq j \leq n,$$

de modo que,

$$p_j^{(i-1)} = a_{ij} - \sum_{k=1}^{i-1} L_{ik} d_{kk} U_{kj}.$$

Essas identidades geram o Algoritmo 18.

No Algoritmo 18 as entradas $w_{lj}^{(i)}$ e L_{ji} são descartadas com a mesma tolerância $\epsilon_{L,W}$. Logo após, depois de aproximar o valor de $\|e_i^T U^{-T}\|_\infty$, pelo Algoritmo 19, os valores U_{ij} são descartados usando $|U_{ij}\|e_i^T U^{-T}\|_\infty < \epsilon_U$.

Os autores apresentam também a versão left-looking do Ainv:

Já a versão Left-looking do RIF-NS é:

Por fim, os autores propõem ainda um novo scaling simétrico para preservar a positividade definida: Suponha que $\widetilde{A} = D^L A = (\tilde{a_{ij}})$, aonde D^L é definida por:

$$D^L = diag(d_{11}^L, \ldots, d_{nn}^L), \quad d_{ii}^L = \frac{a}{max_j |a_{ij}|}.$$

Suponha também que $B = \tilde{A} D^R = (b_{ij})$ com D^R definida por:

$$D^R = diag(d_{11}^R, \ldots, d_{nn}^R), \quad d_{jj}^R = \frac{a}{max_i |\tilde{a}_{ij}|}.$$

Dessas definições concluímos que $\max_i |b_{ij}| \leq 1, \quad \max_j |b_{ij}| \leq 1$. Defina a matriz

Algoritmo 18: Right-looking RIF-NS

$w_i^{(0)} \leftarrow e_i, 1 \leq i \leq n$;
for $i \leftarrow 1$ **to** n **do**
  $v_i = Ae_i$ {not positive definite};
  $v_i = Aw_i^{(i-1)}$ {positive definite};
  $d_{ii} = p_i^{(i-1)} = q_i^{(i-1)} = (w_i^{(i-1)})^T v_i$;
  for $j \leftarrow i+1$ **to** n **do**
    $q_j^{(i-1)} = (w_j^{(i-1)})^T v_i$;
    $L_{ji} = \dfrac{q_j^{(i-1)}}{q_i^{(i-1)}}$;
    apply a dropping rule to L_{ji};
    $w_j^{(i)} = w_j^{(i-1)} - \dfrac{q_j^{(i-1)}}{q_i^{(i-1)}} w_i^{(i-1)}$;
    for all $l \leq i$ apply a dropping rule to $w_{lj}^{(i)}$
  $p_j^{(i-1)} = a_{ij}, i+1 \leq j \leq n$;
  for $k \leftarrow 1$ **to** n **do**
    for $j \leftarrow i+1$ **to** n **do**
      $p_j^{(i-1)} = p_j^{(i-1)} - L_{ik} d_{kk} U_{kj}$;
  $U_{ij} = \dfrac{p_j^{(i-1)}}{p_i^{(i-1)}}, i+1 \leq j \leq n$;
  apply a dropping rule to U_{ij};
Let $w_i = w_i^{(i-1)}$ for $1 \leq i \leq n$;
$W = [w_1, w_2, \ldots, w_n]$, $D = diag[d_{ii}]$.

escalada $M = DAD$, onde

$$D = diag(d_{11}, \ldots, d_{nn}), \quad d_{ii} = \sqrt{d_{ii}^L d_{ii}^R}.$$

Podemos concluir que:

$$\begin{aligned} |m_{ij}||m_{ji}| &= (d_{ii}|a_{ij}|d_{jj})(d_{jj}|a_{ji}|d_{ii}) \\ &= (d_{ii}^L|a_{ij}|d_{jj}^R)(d_{jj}^L|a_{ji}|d_{ii}^R) \cdot \\ & \quad |b_{ij}||b_{ji}| \leq 1 \end{aligned}$$

Portanto, $|m_{ij}| \leq 1$ ou $|m_{ij}| \leq 1$. Como M é positivo definida então $(m_{ij} + m_{ji})^2 \leq 4m_{ii}m_{jj} = 4b_{ii}b_{jj} \leq 4$. Assim, se $|m_{ij}| \leq 1$ então $m_{ji} \leq 3$ e se $|m_{ji}| \leq 1$ então $m_{ij} \leq 3$, de modo que M é limitada.

Algoritmo 19: Condition Estimator

On entry: Unit upper triangular matrix $U = (U_{ij})$;
On return: $t = (t_1, \ldots, t_n)$ such that $t_i \approx \|e_i^T U^{-T}\|_\infty$;
$u = (u_1, \ldots, u_n) = (0, \ldots, 0)^T$;
$v = (v_1, \ldots, v_n) = (0, \ldots, 0)^T$;
for $i \leftarrow 1$ **to** n **do**
    $x_+ = 1 - v_i$,$x_- = -1 - v_i$;
    $y_+ = 1 - u_i$,$y_- = -1 - u_i$, $n_+ = n_- = 0$;
    Let I be the set of column indices of the nonzero entries of $U_{ii+1}, \ldots, U_{in}$;
    $v_+ = \|(v_j + U_{ij}x_+)_{j\in I}\|_1$, $v_- = \|(v_j + U_{ij}x_-)_{j\in I}\|_1$;
    if $v_+ > v_-$ **then**
        $x_i = x_+$ **else**
            $x_i = x_-$
    for $j \in I$ **do**
        $v_j = v_j + U_{ij}x_i$
    for $j \in I$ **do**
        if $|u_j + U_{ij}y_+| > max(2|u_j|, 1/2)$ **then**
            $n_+ = n_+ + 1$
        if $max(2|u_j + U_{ij}y_+|, 1/2) < |u_j|$ **then**
            $n_+ = n_+ - 1$
        if $|u_j + U_{ij}y_-| > max(2|u_j|, 1/2)$ **then**
            $n_- = n_- + 1$
        if $max(2|u_j + U_{ij}y_-|, 1/2) < |u_j|$ **then**
            $n_- = n_- - 1$
        if $n_+ > n_-$ **then**
            $y_i = y_+$ **else**
                $y_i = y_-$
        for $j \in I$ **do**
            $u_j = u_j + U_{ij}y_i$
        $t_i = max(|x_+|, |x_-|, |y_+|, |y_-|)$.

Quanto aos experimentos, os autores concluíram que, além de livre de quebras para matrizes positivo definidas, as novas estratégias de descarte tornam o precondicionador mais robusto. Os autores puderam observar também que, ambas ambas as versões (left-looking e right-looking) são efetivas em reduzir o número de iterações dos métodos de Krylov com precondicionamento, mas que a versão left-looking apresentou melhores resultados. Além disso, a versão left-looking foi mais rápida e mais apropriada em termos de geração e armazenamento dos fatores L e U.

Algoritmo 20: Left-looking AINV with extraction of L and U

for $i \leftarrow 1$ **to** n **do**
  $z_i^{(0)} \leftarrow e_i; w_i \leftarrow e_i;$
  for $j \leftarrow 1$ **to** $i-1$ **do**
    $q_i^{(j-1)} = (w_i^{(j-1)})^T A e_j$, $p_i^{(j-1)} = (z_i^{(j-1)})^T A^T e_j;$
    $L_{ij} = \dfrac{q_i^{(j-1)}}{q_j^{(j-1)}}$, $U_{ji} = \dfrac{p_i^{(j-1)}}{p_j^{(j-1)}};$
    apply a dropping rule to L_{ij} and U_{ji};
    $w_i^{(j)} = w_i^{(j-1)} - \dfrac{q_i^{(j-1)}}{q_j^{(j-1)}} w_j^{(j-1)}$, $z_i^{(j)} = z_i^{(j-1)} - \dfrac{p_i^{(j-1)}}{p_j^{(j-1)}} z_j^{(j-1)};$
    for all $l \leq i$ apply a dropping rule to $w_{lj}^{(i)}$ and $z_{lj}^{(i)}$
  ;
  $q_i^{(i-1)} = (w_i^{(i-1)})^T A e_i$, $p_i^{(i-1)} = (z_i^{(i-1)})^T A^T e_i$ (not positive definite);
  $q_i^{(i-1)} = (w_i^{(i-1)})^T A w_i^{(i-1)}$, $p_i^{(i-1)} = (z_i^{(i-1)})^T A^T z_i^{(i-1)}$ (positive definite);
  $d_{ii} = q_i^{(i-1)}$ or $p_i^{(i-1)}$
Let $z_i \leftarrow z_i^{(i-1)}$ and $w_i \leftarrow w_i^{(i-1)}$ for $1 \leq i \leq n$;
$Z = [z_1, \ldots, z_n], W = [w_1, \ldots, w_n]$ and $D = diag[d_{ii}]$.

2.11 Left-looking Version of AINV Preconditioner with Complete Pivoting Strategy. [68]

Em 2014, o trabalho "Left-looking version of AINV preconditioner with complete pivoting strategy", de Rafiei, [68], discute sobre o AINV aplicado a matrizes não simétricas. Baseado no trabalho de Bollhöfer e Saad, [20], que apresentaram uma estratégia de pivoteamento completo para a versão right-looking, o autor verifica a eficácia do método na versão left-looking. O Algoritmo 22 fica com as seguintes entradas e saídas: Entradas: $A \in \mathbb{R}^{n \times n}$, tolerâncias τ_w e τ_z ambos em $]0, 1]$ e um fator de pivoteamento $\alpha \in]0, 1]$. Saídas: $(\Pi A \Sigma)^{-1} \approx Z D^{-1} W^T$.

Da Tabela 1 a Tabela 14 apresentamos, esquematicamente, os diversos artigos discutidos até aqui.

Algoritmo 21: Left-looking RIF-NS

for $i \leftarrow 1$ **to** n **do**
  $w_i^{(0)} \leftarrow e_i$;
  for $j \leftarrow 1$ **to** $i-1$ **do**
    $q_i^{(j-1)} = (w_i^{(j-1)})^T A e_j$;
    $L_{ij} = \dfrac{q_i^{(j-1)}}{q_j^{(j-1)}}$ apply a dropping rule to L_{ji};
    $w_j^{(i)} = w_j^{(i-1)} - \dfrac{q_i^{(j-1)}}{q_i^{(i-1)}} w_i^{(i-1)}$;
    for all $l \leq j$ apply a dropping rule to $w_{li}^{(j)}$
  $d_{ii} = p_i^{(i-1)} = q_i^{(i-1)} = (w_i^{(i-1)})^T A e_i$ (not positive definite);
  $d_{ii} = p_i^{(i-1)} = q_i^{(i-1)} = (w_i^{(i-1)})^T A w_i^{(i-1)}$ (positive definite);
  $p_j^{(i-1)} = a_{ij}, i+1 \leq j \leq n$;
  for $k \leftarrow 1$ **to** n **do**
    for $j \leftarrow i+1$ **to** n **do**
      $p_j^{(i-1)} = p_j^{(i-1)} - L_{ik} d_{kk} U_{kj}$;
  $U_{ij} = \dfrac{p_j^{(i-1)}}{p_i^{(i-1)}}, i+1 \leq j \leq n$;
  apply a dropping rule to U_{ij};
Let $w_i = w_i^{(i-1)}$ for $1 \leq i \leq n$;
$W = [w_1, w_2, \ldots, w_n]$, $D = diag[d_{ii}]$.

Algoritmo 22: Left-looking version of A-biconjugation process with complete pivoting and dropping

$\Pi = \Sigma = I_n$;
for $i \leftarrow 1$ **to** n **do**
  $m_i \leftarrow n_i \leftarrow 0$, iter $\leftarrow 0$, $satisfied_p = satisfied_q = false$;
  while *Not* $satisfied_q$ **do**
    $z_i^{(0)} = e_i$, iter $\leftarrow$ iter $+1$;
    for $j \leftarrow 1$ **to** $i-1$ **do**
      $p_i^{(j-1)} \leftarrow e_j^T(\Pi A \Sigma) z_i^{(j-1)}$. $z_i^{(j)} = z_i^{(j-1)} - \dfrac{p_i^{(j-1)}}{p_i^{(i-1)}} z_j^{(j-1)}$, For all $l \leq j$ apply dropping rule to $z_{li}^{(j)}$ if its absolute value is less than τ_Z.
    if *iter = 1* **then**
      $q_i^{(i-1)} = e_i^T(\Pi A \Sigma) z_i^{(i-1)}$, **Else** $q_i^{(i-1)} = p_i^{(i-1)}$.
    for $j \leftarrow i+1$ **to** n **do**
      $q_i^{(i-1)} = e_j^T(\Pi A \Sigma) z_i^{(i-1)}$
    if $|q_i^{(i-1)}| < \alpha max_{m \geq i+1} |q_m^{(i-1)}|$ **then**
      $m_i = m_i + 1, \pi_{m_i}^{(i-1)} = I_n$, $satisfied_p = false$. Choose k such that $|q_k^{(i-1)}| = max_{m \leq i+1} |q_m^{(i-1)}|$. Interchange rows i and k of $\pi_{m_i}^{(i-1)}$ and elements $q_i^{(i-1)}$ and $q_k^{(i-1)}$ and $\Pi = \pi_{m_i}^{(i-1)} \Pi$.
    $satisfied_q = true$. **if** *Not* $satisfied_p$ **then**
      $w_i^{(0)} = e_i$. **for** $j \leftarrow 1$ **to** $i-1$ **do**
        $q_i^{(j-1)} = (w_i^{(j-1)})^T(\Pi A \Sigma) e_j$. $w_i^{(j)} = w_i^{(j-1)} - \dfrac{q_i^{(j-1)}}{w_j^{(j-1)}} w_j^{(j-1)}$, For all $l \leq j$ apply dropping rule to $w_{li}^{(j)}$ if its absolute value is less than τ_W.
      $p_i^{(i-1)} = q_i^{(i-1)}$. **if** $|p_i^{(i-1)}| < \alpha max_{m \geq i+1} |p_m^{(i-1)}|$ **then**
        $n_i = n_i + 1, \sigma_{m_i}^{(i-1)} = I_n$. $satisfied_p = false$. Choose l such that $|p_l^{(i-1)}| = max_{m \leq i+1} |p_m^{(i-1)}|$. Interchange rows i and l of $\sigma_{m_i}^{(i-1)}$ and elements $p_i^{(i-1)}$ and $p_l^{(i-1)}$. $\Sigma = \Sigma \sigma_{m_i}^{(i-1)}$.
      $satisfied_q = true$.
  $d_{ii} = q_i^{(i-1)}$.

AINV	
AUTORES	Benzi, Meyer e Tüma
ANO	1996
CONCEBIDO PARA	Matrizes SPD
SOLVER	CG
FATORAÇÃO	$A^{-1} \approx ZD^{-1}Z^{\mathsf{T}}$.
MATRIZ Z	Começa pela identidade: $z_i^{(0)} = e_i$. Atualiza usando "Gramm-Schmidt"com o produto Interno de A: $z_i^{(k)} = z_i^{(k-1)} - m_{ik} z_k^{(k-1)}$.
DESCARTES	Tolerância (normalmente 0.1) ou Estrutura de Zeros Pré-definida
ESCALONAMENTO	Máximo
LIVRE DE QUEBRA	Matrizes $\mathcal{M}$ e $\mathcal{H}$

Tabela 1 AINV

NS-AINV	
AUTORES	Benzi e Tüma
ANO	1998
CONCEBIDO PARA	Matrizes Não Singulares e Não Simétricas (NS)
SOLVER (nos testes)	Bi-CGSTAB, QMR e GMRES
FATORAÇÃO	$A^{-1} \approx ZD^{-1}W^{\mathsf{T}}$.
MATRIZES Z e W	Começam pela identidade: $w_i^{(0)} = z_i^{(0)} = e_i$. As colunas de Z são construídas sobre as linhas de A: $m_{ik} = \dfrac{a_k^T z_i}{a_i^T z_i}$, enquanto que as colunas de W são construídas sobre as colunas de A: $n_{ik} = \dfrac{c_k^T z_i}{c_i^T z_i}$. Atualizam usando: $z_i^{(k)} = z_i^{(k-1)} - m_{ik} z_k^{(k-1)}$ e $z_i^{(k)} = w_i^{(k-1)} - n_{ik} w_k^{(k-1)}$.
DESCARTES	Tolerância (normalmente 0.1) ou Estrutura de Zeros Pré-definida
ESCALONAMENTO	Máximo
LIVRE DE QUEBRA	Matrizes $\mathcal{M}$ e $\mathcal{H}$

Tabela 2 NS-AINV

SAINV	
AUTORES	Benzi, Cullum e Tüma
ANO	2000
CONCEBIDO PARA	Matrizes SPD
SOLVER (nos testes)	CG
FATORAÇÃO	$A^{-1} \approx ZD^{-1}Z^{\mathsf{T}}$.
PIVÔS	Calculados em dois passos: $v_i = Az_i$ e $p_i = v_i^{\mathsf{T}} z_i$.
MATRIZES Z	Começa pela identidade: $= z_i^{(0)} = e_i$. Atualiza usando: $z_i^{(k)} = z_i^{(k-1)} - m_{ik} z_k^{(k-1)}$.
DESCARTES	Tolerância (normalmente 0.1) ou Estrutura de Zeros Pré-definida
ESCALONAMENTO	Pela Diagonal
REORDENAMENTO	MMD
LIVRE DE QUEBRA	SPD

Tabela 3 SAINV

NS-SAINV	
AUTORES	Bridson e Tang
ANO	2000
CONCEBIDO PARA	Matrizes NS
SOLVER (nos testes)	CG e BICGSTAB
FATORAÇÃO	$A^{-1} \approx ZD^{-1}W^{\mathsf{T}}$.
MULTIPLICADORES	Pivô em um passo: $D_{ii} = w_i^{\mathsf{T}} A z_i$. Multiplicadores calculados como: $m_{ik} = \frac{a_k^{\mathsf{T}} z_i}{D_{ii}}$ e $n_{ik} = \frac{w_i^{\mathsf{T}} c_k}{D_{ii}}$.
MATRIZES Z e W	Começam pela identidade: $w_i^{(0)} = z_i^{(0)} = e_i$. Atualizam usando: $z_i^{(k)} = z_i^{(k-1)} - m_{ik} z_k^{(k-1)}$ e $z_i^{(k)} = w_i^{(k-1)} - n_{ik} w_k^{(k-1)}$.
DESCARTES	Tolerância (normalmente 0.1)
ESCALONAMENTO	Pela Diagonal
REORDENAMENTO	MIP
LIVRE DE QUEBRA	Não

Tabela 4 NS-SAINV

NS-SBAINV	
AUTORES	Bridson e Tang
ANO	2000
CONCEBIDO PARA	Matrizes NS (em blocos)
SOLVER (nos testes)	CG e BICGSTAB
FATORAÇÃO	$A^{-1} \approx ZD^{-1}W^{\mathsf{T}}$.
MULTIPLICADORES	Pivô em um passo: $D_{ii} = W_i^{\mathsf{T}} A Z_i$. Multiplicadores calculados como: $M_{ik} = {D_{ii}}^{-1} A_k^{\mathsf{T}} Z_i$ e $N_{ik} = {D_{ii}}^{-1} W_i^{\mathsf{T}} C_k$.
MATRIZES Z e W	Começam pela identidade: $W_i^{(0)} = Z_i^{(0)} = E_i$. Atualizam usando: $Z_i^{(k)} = Z_i^{(k-1)} - Z_k^{(k-1)} M_{ik}$ e $Z_i^{(k)} = W_i^{(k-1)} - W_k^{(k-1)} N_{ik}$.
DESCARTES	Tolerância (normalmente 0.1)
ESCALONAMENTO	Bloco Diagonal
REORDENAMENTO	MIP
LIVRE DE QUEBRA	Não

Tabela 5 NS-SBAINV

SBAINV	
AUTORES	Benzi, Kouhia e Tüma
ANO	2001
CONCEBIDO PARA	Matrizes SPD (em blocos - Apenas Matrizes de problemas de Mecânica Estrutural e dos Sólidos foram utilizadas)
SOLVER (nos testes)	CG
FATORAÇÃO	$A^{-1} \approx ZD^{-1}Z^{\mathsf{T}}$.
MULTIPLICADORES	Pivô em um passo: $D_{ii} = Z_i^{\mathsf{T}} A Z_i$. Multiplicadores calculados como: $M_{ik} = {P_{ii}}^{-1} A_k^{\mathsf{T}} Z_i$.
MATRIZ Z	Começam pela identidade: $Z_i^{(0)} = E_i$. Atualizam usando: $Z_i^{(k)} = Z_i^{(k-1)} - Z_k^{(k-1)} M_{ik}$.
DESCARTES	• AINV(0) - copia em Z a esparsidade de A. • Tolerância • Descarte Relativo (cada entrada de Z é descartada caso seja menor que uma tolerância multiplicada pela norma infinito da atual linha de A). • Forma pré-definida de zeros. • Filtragem posterior (depois que Z é calculado, usa-se outra tolerância, para descartar mais entradas de Z).
ESCALONAMENTO	Bloco Jacobi
REORDENAMENTO	MMD
LIVRE DE QUEBRA	Matrizes SPD

Tabela 6 SBAINV

RIF	
AUTORES	Benzi e Tüma.
ANO	2003
CONCEBIDO PARA	Matrizes NS
SOLVER (nos testes)	CGLS
FATORAÇÃO	$C = A^{\mathsf{T}}A$ e $C^{-1} \approx ZD^{-1}Z^{\mathsf{T}}$.
MULTIPLICADORES	O Pivô, que é: $D_{ii} = z_i^{\mathsf{T}}A^{\mathsf{T}}Az_i$ é calculado em dois passos: $u_i = Az_i$ e $D_{ii} = u_i^{\mathsf{T}}u_i$. Multiplicadores calculados como: $m_{ik} = \frac{u_k^{\mathsf{T}}u_i}{D_{kk}}$.
MATRIZ Z	Começa pela identidade: $z_i^{(0)} = e_i$. Atualizam usando: $z_i^{(k)} = z_i^{(k-1)} - m_{ik}z_k^{(k-1)}$.
DESCARTES	Tolerância (normalmente entre 0.1 e 0.5)
LIVRE DE QUEBRA	Para qualquer matriz NS

Tabela 7 RIF

ISAINV	
AUTORES	Fujino e Ikeda.
ANO	2004
CONCEBIDO PARA	Matrizes SPD
SOLVER (nos testes)	CG
FATORAÇÃO	$A^{-1} \approx ZD^{-1}W$.
MULTIPLICADORES	O Pivô, que é: $D_{ii} = z_i^{\mathsf{T}}Az_i$ é calculado em dois passos: $u_i = Az_i$ e $D_{ii} = u_i^{\mathsf{T}}z_i$. Multiplicadores calculados como: $m_{ik} = \frac{u_k^{\mathsf{T}}z_i}{D_{kk}}$.
MATRIZ Z	Começa pela identidade: $z_i^{(0)} = e_i$. Atualizam usando: $z_i^{(k)} = z_i^{(k-1)} - m_{ik}z_k^{(k-1)}$.
DESCARTES	Double Dropping: Se $\lvert d_j/d_i\rvert > tol - 2$ e se $\lvert z_{jk}^{(i-1)} - \frac{p_j^{(i-1)}}{p_i^{(i-1)}}z_{ik}^{(i-1)}\rvert > tol - 1$ então: $z_j^{(i)} \leftarrow z_j^{(i-1)} - \frac{p_j^{(i-1)}}{p_i^{(i-1)}}z_i^{(i-1)}$.
LIVRE DE QUEBRA	Para qualquer matriz SPD

Tabela 8 ISAINV

SAINV VARIETY	
AUTORES	Rafiei e Toutonian.
ANO	2008
CONCEBIDO PARA	Matrizes Não Simétricas e Positivo Definidas (NSPD).
SOLVER (nos testes)	QMR, BICSTAB e GMRES(10)
FATORAÇÃO	$A^{-1} \approx U_l^{-1} D^{-1} W^{\mathsf{T}}$, aonde U vai ser obtido nos multiplicadores e U^{-1} é estimado pela série de Newmann: $U_l^{-1} = I + (I-U) + \ldots + (I-U)^l$.
MULTIPLICADORES	O Pivô, que é: $D_{ii} = w_i^{\mathsf{T}} A w_i$ é calculado em dois passos: $v_i = A w_i$ e $D_{ii} = v_i^{\mathsf{T}} w_i$. Multiplicadores calculados como: $m_{ik} = \dfrac{v_k^{\mathsf{T}} w_i}{D_{kk}}$. As entradas de U são calculas como: $p_j^{(i-1)} = a_{ij} - \sum_{k=1}^{i-1} u_{kj} q_i^{(i-1)}$ e $U_{ij} = \dfrac{p_j^{(i-1)}}{p_i^{(i-1)}}$.
MATRIZ W	Matriz Z não é calculada diretamente. A matriz W começa pela identidade: $z_i^{(0)} = e_i$. Atualiza usando: $w_i^{(k)} = w_i^{(k-1)} - m_{ik} w_k^{(k-1)}$.
DESCARTES	Tolerância (entre 0.001 e 0.01)
LIVRE DE QUEBRA	Para qualquer matriz NSPD

Tabela 9 SAINV VARIETY

FFAPINV	
AUTORES	Zhang.
ANO	2000
CONCEBIDO PARA	Matrizes Não Simétricas e não singulares (NS).
SOLVER (nos testes)	GMRES
FATORAÇÃO	$A^{-1} \approx Z D^{-1} W$.
MULTIPLICADORES	O Pivô é: $D_i = w_i A z_i$. Multiplicadores calculados como: $m_{ik} = -\dfrac{w_k A_{*i}}{D_k}$ e $n_{ik} = -\dfrac{A_{i*} z_k}{D_k}$.
MATRIZES Z E W	Começam pela identidade: $z_i^{(0)} = e_i$ e $w_i^{(0)} = e_i^{\mathsf{T}}$. Atualiza usando: $z_i^{(k)} = z_i^{(k-1)} - m_{ik} z_k^{(k-1)}$ e $w_i^{(k)} = w_i^{(k-1)} - n_{ik} w_k^{(k-1)}$.
DESCARTES	Tolerância (normalmente 0.1)
LIVRE DE QUEBRA	Não

Tabela 10 FFAPINV

FFAPINV - NSPD	
AUTORES	Salkuyeh.
ANO	2010
CONCEBIDO PARA	Matrizes Não Simétricas e Positivo Definidas (NSPD).
SOLVER (nos testes)	GMRES
FATORAÇÃO	$A^{-1} \approx ZD^{-1}W$.
MULTIPLICADORES	O Pivô é: $D_i = A_{i*}z_i$ ou $D_i = z_i^{\mathsf{T}}Az_i$, caso $A_{i*}z_i = 0$. Multiplicadores calculados como: $m_{ik} = -\frac{w_k A_{*i}}{D_k}$ e $n_{ik} = -\frac{A_{i*}z_k}{D_k}$.
MATRIZES Z E W	Começam pela identidade: $z_i^{(0)} = e_i$ e $w_i^{(0)} = e_i^{\mathsf{T}}$. Atualiza usando: $z_i^{(k)} = z_i^{(k-1)} - m_{ik}z_k^{(k-1)}$ e $w_i^{(k)} = w_i^{(k-1)} - n_{ik}w_k^{(k-1)}$.
DESCARTES	Tolerância (normalmente 0.1)
LIVRE DE QUEBRA	Para quaisquer matrizes NSPD

Tabela 11 FFAPINV-NSPD

RL-AINV	
AUTORES	Rafiei e Bollhöfer.
ANO	2011
CONCEBIDO PARA	Matrizes Não Simétricas (NS).
SOLVER (nos testes)	BICGSTAB, GMRES(30) e TFQMR
FATORAÇÃO	$A^{-1} \approx ZD^{-1}W^{\mathsf{T}}$.
MULTIPLICADORES	Calculam-se primeiro $u_i = Az_i^{(i-1)}$ e $v_i = Aw_i^{(i-1)}$. Logo após calculam-se $p_j^{(i-1)} = (z_j^{(i-1)})^{\mathsf{T}}u_i$ e $q_j^{(i-1)} = (w_j^{(i-1)})^{\mathsf{T}}v_i$. Os pivôs são dados por $d_i = p_i^{(i-1)}$ ou $d_i = q_i^{(i-1)}$. Os multiplicadores são dados por: $L_{ji} = \frac{q_j^{(i-1)}}{q_i^{(i-1)}}$ e $U_{ij} = \frac{p_j^{(i-1)}}{p_i^{(i-1)}}$.
MATRIZES Z E W	Começam pela identidade: $z_i^{(0)} = e_i$ e $w_i^{(0)} = e_i$. Atualiza usando: $z_i^{(k)} = z_i^{(k-1)} - U_{ik}z_k^{(k-1)}$ e $w_i^{(k)} = w_i^{(k-1)} - L_{ki}w_k^{(k-1)}$.
DESCARTES	Dados $\epsilon_{L,W}$ e $\epsilon_{U,Z}$ serão descartados: $\lvert w_{lj}^{(i)}\rvert < \epsilon_{L,W}, \lvert L_{ji}\rvert < \epsilon_{L,W}, \lvert z_{lj}^{(i)}\rvert < \epsilon_{U,Z}$ e $\lvert U_{ij}\rvert < \epsilon_{U,Z}$. Caso $\lvert w_{lj}^{(i)}\rvert < \epsilon_{L,W}$ ocorra, um descarte "safe"é aplicado ao fator L: $\lvert L_{ji}\rvert\lVert W_{*i}\rVert_\infty < \epsilon_{L,W}$. Analogamente, caso $\lvert z_{lj}^{(i)}\rvert < \epsilon_{U,Z}$ ocorra, um descarte "safe"é aplicado ao fator U: $\lvert U_{ij}\rvert\lVert Z_{*i}\rVert_\infty < \epsilon_{U,Z}$.
LIVRE DE QUEBRA	Para quaisquer matrizes NSPD

Tabela 12 Right-Looking AINV

RL-RIF	
AUTORES	Rafiei e Bollhöfer.
ANO	2011
CONCEBIDO PARA	Matrizes Não Simétricas (NS).
SOLVER (nos testes)	BICGSTAB, GMRES(30) e TFQMR
FATORAÇÃO	$A \approx LDU$ (precondicionador do tipo ILU).
MULTIPLICADORES	O pivô é calculado em dois passos: $v_i = Aw_i^{(i-1)}$ e $d_{ii} = p_i^{(i-1)} = q_i^{(i-1)} = (w_i^{(i-1)})^{\mathsf{T}} v_i$. Temos $q_j^{(i-1)} = (w_j^{(i-1)})^{\mathsf{T}} v_i$ e os multiplicadores são dados por: $L_{ji} = \dfrac{q_j^{(i-1)}}{q_i^{(i-1)}}$. Temos $p_j^{(i-1)} = a_{ij}, i+1 \leq j \leq n$. Para $k = 1$ até $i-1$ temos $p_j^{(i-1)} = p_j^{(i-1)} - L_{ik} d_{kk} U_{kj}$ e $U_{ij} = \dfrac{p_j^{(i-1)}}{p_i^{(i-1)}}$.
MATRIZ W	Começa pela identidade: $w_i^{(0)} = e_i$. Atualiza usando: $w_i^{(k)} = w_i^{(k-1)} - L_{ki} w_k^{(k-1)}$.
DESCARTES	Dado $\epsilon_{L,W}$ serão descartados: $\lvert w_{lj}^{(i)} \rvert < \epsilon_{W,L}$ e $\lvert L_{lj}^{(i)} \rvert < \epsilon_{W,L}$. Caso aconteça de $\lvert w_{lj}^{(i)} \rvert < \epsilon_{W,L}$ então o descarte "safe"é aplicado para o fator L: $\lvert L_{lj}^{(i)} \rvert \lVert W_{*i} \rVert_\infty < \epsilon_{W,L}$. Um descarte é aplicado também no fator U: $\lvert U_{ij} \rvert \lVert e_i^{\mathsf{T}} U^{-\mathsf{T}} \rVert_\infty < \epsilon_U$. O valor de $\lVert e_i^{\mathsf{T}} U^{-\mathsf{T}} \rVert_\infty$ é utilizado um outro algoritmo que aproxima o valor.
LIVRE DE QUEBRA	Para quaisquer matrizes NSPD

Tabela 13 Right-Looking RIF

LL-NSAINV	
AUTORES	Rafiei.
ANO	2014
CONCEBIDO PARA	Matrizes Não Simétricas (NS).
SOLVER (nos testes)	BICGSTAB e TFQMR
FATORAÇÃO	$(\Pi A \Sigma)^{-1} \approx Z D^{-1} W^{\mathsf{T}}$.
MULTIPLICADORES	Os multiplicadores da matriz Z são $p_i^{(j-1)} = e_j^{\mathsf{T}}(\Pi A \Sigma) z_i^{(j-1)}$ e os da matriz W são: $q_i^{(j-1)} = (w_i^{(j-1)})^{\mathsf{T}}(\Pi A \Sigma) e_j$. Os pivôs de p_i são trocados pelo maior multiplicador daquela iteração (agindo nas linhas de A), já os pivôs de q_i são trocados pelo maior multiplicador daquela iteração (agindo nas colunas de A).
MATRIZES Z E W	Começam pela identidade: $w_i^{(0)} = z_i^{(0)} = e_i$. Atualiza usando: $w_i^{(k)} = w_i^{(k-1)} - \frac{q_i^{(j-1)}}{q_i^{(i-1)}} w_k^{(k-1)}$ e $z_i^{(k)} = z_i^{(k-1)} - \frac{p_i^{(j-1)}}{p_i^{(i-1)}} w_k^{(k-1)}$.
DESCARTES	Por tolerância.
REORDENAMENTO	MLND
LIVRE DE QUEBRA	Para quaisquer matrizes NSPD

Tabela 14 Left-Looking AINV With Complete Pivoting

3 O BAINV E SUA VALIDAÇÃO

Considere o espaço $\mathbb{R}^{n\times n}$. Suponha que $A \in \mathbb{R}^{n\times n}$ possua uma partição interna em blocos da seguinte maneira:

$$A = \begin{pmatrix} A_{11} & A_{12} & \dots & A_{1k} \\ A_{21} & A_{22} & \dots & A_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ A_{k1} & A_{k2} & \dots & A_{kk} \end{pmatrix} \tag{16}$$

aonde A_{ij} é matriz de ordem $n_i \times n_j$ para $i, j = 1, \dots, k$, $n = \sum_{i=1}^{k} n_i$. Note que, a partir dessa definição, os blocos diagonais são quadrados. No caso em que todos os blocos da partição são de mesma ordem, diremos que a partição de blocos é uniforme. As colunas (em bloco) da matriz A podem ser chamadas de vetores-bloco ou bloco-colunas. As matrizes $A_{ij} \in \mathbb{R}^{n_i\times n_j}, 1 \leq i, j \leq k$ são chamadas de blocos de A (ou entradas-bloco de A) e k é a dimensão bloco da matriz A. Os vetores bloco canônicos E_i com respeito a partição de A são a i-ésima do coluna-bloco da identidade de ordem n, I_n, com a mesma partição de A, com $i = 1, \dots, k$.

Definição 8 (A-conjugação)**.** *Seja $V = \{V_1, \dots, V_k\}$ com uma partição análoga a partição de A em 16. Diremos que V é conjugado por A ou ortogonal por A se*

$$V_i^{\mathsf{T}} A V_j = 0_{n_i\times n_j},\ i \neq j,$$

onde $0_{n_i\times n_j}$ é uma matriz nula de ordem $n_i \times n_j$. Nesse caso, também dizemos que V_i e V_j são conjugados por A.

Seja A uma matriz de ordem n, SBPD e seja um conjunto $\{Z_i\} \subset \mathbb{R}^{n\times n_i}$, $i = 1, \dots, k$ de vetores conjugados por A, então a matriz $n \times n$:

$$Z = [Z_1\ Z_2\ \dots\ Z_k],$$

é tal que

$$Z^{\mathsf{T}} A Z = D,$$

com D sendo uma matriz diagonal por blocos.

Como A é não singular, então, se Z é não singular, temos:

$$A^{-1} = ZD^{-1}Z^{\mathsf{T}}. \tag{17}$$

Essa decomposição é a base do BAINV.

Quando A é uma matriz não simétrica, procuramos duas matrizes Z e W tais que $W^TAZ = D$, ou seja, as colunas bloco de W são conjugadas por A às colunas bloco de Z. Assim, a fatoração da inversa de A obtida é $A^{-1} = ZD^{-1}W^{\mathsf{T}}$.

Em [11], o AINV foi apresentado para matrizes simétricas e positivo definidas. O critério de Sylvester, [43], diz que uma matriz quadrada é simétrica e positivo definida se e somente se todos os seus menores principais são estritamente positivos. Para a versão bloco do AINV, como mostraremos mais adiante, não precisamos de tal restrição quanto aos menores principais, só precisaremos que alguns menores principais sejam diferente de zero, mais especificamente, os menores principais de ordem $N_1 = n_1, N_2 = n_1 + n_2, \ldots, N_k = n_1 + \ldots + n_k = n$.

De modo a fixar a notação usada, vamos determinar o seguinte: O ij-ésimo bloco de A é denotado por A_{ij}. Já a i-ésima coluna de A é denotada por A_i. Por fim, a submatriz principal de A de ordem bloco N_i, ou seja, $A(1 : N_i, 1 : N_i)$, será denotada por $\boldsymbol{A_{N_i}}$.

Definição 9 (SBPD)**.** *Uma matriz simétrica A de dimensão bloco k é dita simétrica e bloco-positivo definida (SBPD) se* $\det(\boldsymbol{A_{N_i}}) > 0$, $i = 1, \ldots, k$.

Dizemos que A, não necessariamente simétrica, possui uma partição rígida de blocos $\det(\boldsymbol{A_{N_i}}) \neq 0$, $i = 1, \ldots, k$. Mais adiante, demonstraremos que essa condição é suficiente para garantir a validade do BAINV.

Nós pontuamos aqui que a Definição 9 acima, coincide com a definição de matriz SPD quando todos os blocos são de tamanho 1, pelo Critério de Sylvester, [43]. Mais geralmente, é possível que uma matriz SBPD não seja SPD, enquanto que, qualquer matriz SPD, independentemente da partição de blocos utilizada é SBPD.

Seja A, de ordem n e SBPD, e sejam A_i e Z_i as i-ésimas colunas em bloco de A e Z, respectivamente. O AINV em blocos (BAINV), em notação inspirada no Matlab, é descrito pelo Algoritmo 23.

Algoritmo 23: BAINV

1 $Z_1 \leftarrow E_1$;
2 $D_{11} \leftarrow A_{11}$;
3 **for** $J \leftarrow 2$ **to** N **do**
4 $\quad Z_J^{(0)} \leftarrow E_J$;
5 $\quad$ **for** $I \leftarrow 1$ **to** $J-1$ **do**
6 $\quad\quad M_J^{(I-1)} \leftarrow A_I^T Z_J^{(I-1)}$;
7 $\quad\quad Z_J^{(I)} \leftarrow Z_J^{(I-1)} - Z_I D_{II}^{-1} M_J^{(I-1)}$;
8 $\quad Z_J \leftarrow Z_J^{(J-1)}$;
9 $\quad D_{JJ} \leftarrow A_J^T Z_J$;
10 $D \leftarrow \begin{bmatrix} D_{11} & & \\ & \ddots & \\ & & D_{NN} \end{bmatrix}$ and $Z \leftarrow \begin{bmatrix} Z_1 & \cdots & Z_N \end{bmatrix}$;

No Algoritmo 23, os índices altos referem-se à iteração em curso.

Os blocos D_i são também chamados de pivôs.

Lema 1. *Se o algoritmo não quebrar (i.e., nenhum D_{II} singular for gerado), nós temos, para qualquer $1 \leqslant J \leqslant N$ fixado,*

$$Z_J = Z_J^{(K)} - \sum_{I=K+1}^{J-1} Z_I D_{II}^{-1} M_J^{(I-1)} \tag{18}$$

para todo $0 \leqslant K \leqslant J-1$.

Demonstração. Esse resultado segue imediatamente das linhas 4, 7 e 8 do Algoritmo 23 □

A seguir provamos que o Algoritmo 23 produz as matrizes desejadas.

Teorema 7. *Seja A, $n \times n$, uma matriz com uma partição rígida de blocos. O Algoritmo 23 produz, em aritmética exata, uma matriz D não singular e uma matriz Z triangular superior (por blocos) não singular, cujos blocos diagonais são a identidade e cujas colunas em bloco são conjugadas por A.*

Demonstração. Nós iremos proceder por indução. É suficiente provar que a hipótese H_J

abaixo vale para cada $J \in \{1, \ldots, N\}$.

$$H_J : \begin{cases} D_{JJ} \text{ é inversível;} & \text{(19a)} \\ Z_J^{\mathsf{T}} A Z_J = D_{JJ}; & \text{(19b)} \\ Z_I^{\mathsf{T}} A Z_J = Z_J^{\mathsf{T}} A Z_I = 0, \qquad I \in \{1, 2, \ldots, J-1\}; & \text{(19c)} \\ E_J^{\mathsf{T}} Z_J = I; & \text{(19d)} \\ E_I^{\mathsf{T}} Z_J = 0, \qquad I \in \{J+1, J+2, \ldots, N\}; & \text{(19e)} \\ A_I^{\mathsf{T}} Z_J = 0, \qquad I \in \{1, 2, \ldots, J-1\}; & \text{(19f)} \end{cases}$$

Para $J = 1$, as condições 19c e 19f são vazias e as condições 19a, 19b, 19d e 19e seguem do fato que $D_{11} = A_{11}$ e $Z_1 = E_1$. Agora, para o passo de indução, nós assumimos que $H_1, H_2, \ldots, H_{J-1}$ valem e os usaremos para provar H_J.

Usando o Lema 1 com $K = 0$ e pré-multiplicando por E_L^{T} teremos que

$$E_L^{\mathsf{T}} Z_J = E_L^{\mathsf{T}} Z_J^{(0)} - \sum_{I=1}^{J-1} E_L^{\mathsf{T}} Z_I D_{II}^{-1} M_J^{(I-1)}.$$

Então 19d e 19e seguem de $J \leqslant L \leqslant N$ e usando os fatos de que $Z_J^{(0)} = E_J$ e, por hipótese de indução, $E_L^{\mathsf{T}} Z_I = 0$ para $1 \leqslant I \leqslant J-1$ e $L > I$.

A fim de provar que 19f, nós iniciaremos pré-multiplicando a linha 7 do Algoritmo 23 por A_I^T para algum $I < J$, obtendo

$$A_I^T Z_J^{(I)} = A_I^T Z_J^{(I-1)} - A_I^T Z_I D_{II}^{-1} M_J^{(I-1)}.$$

Nós reconhecemos o primeiro termo no lado direito como $M_J^{(I-1)}$ (cf. linha 6 do Algoritmo 23) e, notando que $A_I^T Z_I = D_{II}$ (cf. linha 9 do Algoritmo 23), nós concluímos que $A_I^T Z_J^{(I)} = 0$. Agora, nós invocaremos o Lema 1 com $K = I$ e escrevemos

$$A_I^T Z_J = A_I^T Z_J^{(I)} - \sum_{L=I+1}^{J-1} A_I^T Z_L D_{LL}^{-1} M_J^{(L-1)}.$$

Nós acabamos de provar que o primeiro termo no lado direito é zero. Por conta da hipótese de indução, que garante que $A_I^{\mathsf{T}} Z_L$ é sempre zero, o segundo termo no lado direito é zero também.

O que nós provamos até agora nos garante que a $N \times J$ bloco matriz $Z_{:,1:J}$ é

bloco triangular superior e que $AZ_{:,1:J}$ é bloco triangular inferior. Nós observamos que $Z_{:,1:J}^T AZ_{:,1:J}$ é produto de duas matrizes triagular inferiores, $Z_{:,1:J}^T(AZ_{:,1:J})$, sendo então triangular inferior também. Analogamente, $Z_{:,1:J}^T AZ_{:,1:J}$ é o produto de duas matrizes bloco triangulares superior, $(AZ_{:,1:J})^T Z_{:,1:J}$ e, portanto, é bloco triangular superior também. Isso prova 19c.

Escrevendo a identidade como $I = \sum_{K=1}^{N} E_K E_K^T$, nós temos

$$Z_J^T AZ_J = \sum_{K=1}^{N} (E_K^T Z_J)^T (E_K^T A) Z_J = \\ = \sum_{K=1}^{J-1} (E_K^T Z_J)^T A_K^T Z_J + (E_J^T Z_J)^T A_J^T Z_J + \sum_{K=J+1}^{N} (E_K^T Z_J)^T A_K^T Z_J.$$

Perceba que as duas somas no último termo são zero devido as equações 19f e 19e da hipótese de indução. Então 19b segue de 19d e a linha 9 do Algoritmo 23.

Devemos notar que as hipóteses 19b e 19c implicam que $Z_{:,1:J}^T AZ_{:,1:J}$ igual a $\text{diag}(D_{11}, \ldots, D_{JJ})$, uma matriz bloco-diagonal com D_{JJ} em sua bloco-diagonal. Além disso, 19e garante $Z_{J+1:N,1:J}$ é zero, de modo que $\text{diag}(D_{11}, \ldots, D_{JJ}) = Z_{:,1:J}^T AZ_{:,1:J} = Z_{1:J,1:J}^T A_{1:J:,1:J} Z_{1:J:,1:J}$. A matriz $Z_{1:J,1:J}$ e sua transposta são matrizes triangulares com a diagonal de 1's, e, por isso, são inversíveis. Uma vez que $A_{1:J:,1:J}$ também é inversível, por hipótese, nós temos 19a.

□

Devemos notar que essa demonstração pode ser generalizada para o caso não simétrico, pois as contas que envolvem as colunas da matriz W são análogas àquelas que fizemos com as colunas da matriz Z.

Corolário 1. *Seja $A \in \mathbb{R}^{n \times n}$, com $n = m * k$, uma matriz SBPD. Seja $\boldsymbol{A}_{N_r}$ a submatriz líder principal de ordem $n_1 + \ldots + n_r$ de A. Então*

$$\det(\boldsymbol{A}_{N_r}) = \prod_{i=1}^{r} \det\left(P_i\right), \tag{20}$$

onde P_i, $1 \leq i \leq r \leq k$, são os blocos da diagonal da matriz D, computada pelo BAINV.

Demonstração. Da demonstração do Teorema 7 vimos que

$$\boldsymbol{Z}_{N_r}^{\mathsf{T}} \boldsymbol{A}_{N_r} \boldsymbol{Z}_{N_r} = \begin{pmatrix} D_1 & 0 & \cdots & 0 \\ 0 & D_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & D_r \end{pmatrix},$$

Assim, aplicando o determinante em ambos os lados e utilizando o fato de que $\det(\boldsymbol{Z}_{N_r}) = 1$, o resultado segue imediatamente. □

Corolário 2. *Seja A uma matriz SBPD de ordem n partiocinada como em 16. Considere a submatriz líder principal $\boldsymbol{A}_{N_i}$, com $i = 1, \ldots, k$, $\Delta_0 = 1$, e $\Delta_i = \det(\boldsymbol{A}_{N_i})$. Então:*

$$\det(D_i) = \frac{\Delta_i}{\Delta_{i-1}}.$$

Demonstração. Faremos indução em i. Para $i = 1$, do Corolário 1, temos que $\det(D_1) = \det(A_1) = \dfrac{\Delta_1}{\Delta_0}$.

Agora, usando a hipótese de indução

$$\Delta_{k+1} = \det(A_{k+1}) = \prod_{i=1}^{k+1} \det(D_i) =$$

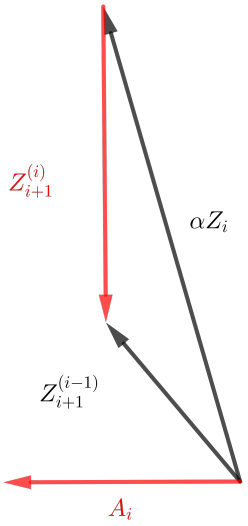
$$\frac{\Delta_1}{\Delta_0}\frac{\Delta_2}{\Delta_1} \cdots \frac{\Delta_k}{\Delta_{k-1}} \det(D_{k+1}) = \Delta_k \det(D_{k+1}).$$

□

3.1 Matemática do BAINV

Nessa seção vamos analisar como, matematicamente, o AINV funciona. Nesse sentido vamos buscar analisar a geometria e suas generalizações aos caso bloco. Assim, vamos acompanhar qual é o efeito de uma iteração, tanto no sentido geométrico quanto no preenchimento das matrizes em questão. Esse entendimento melhora o acompanhamento das demonstrações dos próximos capítulos.

Sobre a geometria do AINV, veja a Figura 1. Nela, vemos como o AINV constrói um vetor $Z_{i+1}^{(i)}$ ortogonal a A_i usando Z_i e $Z_{i+1}^{(i-1)}$.



-se

Figura 1 Representação Geométrica do BAINV

Geometricamente, a ortogonalidade surge da seguinte maneira: pega-se a última versão de Z_{i+1}, ou seja, $Z_{i+1}^{(i-1)}$ que foi calculada na iteração anterior e soma-se com um múltiplo de Z_i de tal modo que a diferença entre $Z_{i+1}^{(i-1)}$ e Z_i seja ortogonal.

Note que, na i-ésima iteração, o AINV leva a coluna Z_r, $r > i$, no complemento ortogonal de $\{Z_1, \ldots, Z_i\}$. Ou seja, em cada iteração, adiciona-se a ortogonalidade a mais uma linha de A. É interessante observar que, como $A_1^{\mathsf{T}} Z_1 > 0$, o vetor Z_2 é construído no semi-plano gerado por Z_1 que mantém a mesma orientação de e_1 e e_2. Mais geralmente, a orientação de $\{e_1, \ldots, e_i\}$ é mantida por $\{Z_1, \ldots, Z_i\}$, já que os menores principais da matriz A são positivos pelo critério de Sylvester.

Vamos considerar o seguinte exemplo, com o Ainv aplicado em sua versão escalar:

$$A := \begin{pmatrix} 2 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 0 & 2 \end{pmatrix}$$

Na primeira iteração do Algoritmo 23, teremos $D_1 = a_{11} = 2$ e $Z_1 = (1\ 0\ 0)^{\mathsf{T}}$. A primeira direção A-conjugada é e_1.

A primeira coluna de A é

$$a_1 = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix},$$

que tem projeção não nula em e_1. Para que o Algoritmo 2 não quebre nessa etapa devemos ter $\Delta_1 \neq 0$. Isso nada mais é que dizer que a projeção da primeira coluna de A em e_1 é não nula e essa projeção possui a mesma orientação de e_1.

Seguindo esta linha de pensamento, na segunda iteração do Algoritmo 2 temos: $D_2 = 2,5$ e $Z_2 = (-0,5\ 1\ 0)^\mathsf{T}$. Uma vez que a primeira iteração foi completa, para que a segunda esteja garantida, é necessário que $\Delta_2 \neq 0$ e é, pois é igual a 5. Note que a segunda direção A conjugada Z_2 está no plano xy e é linearmente independente de e_1 (naturalmente, se fosse linearmente dependente de Z_1, teríamos, necessariamente, $\Delta_2 = 0$). Mas o que isto quer dizer geometricamente?

Observe as duas primeiras colunas de A: $a_1 = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix}$, e $a_2 = \begin{pmatrix} 1 \\ 3 \\ 0 \end{pmatrix}$. As projeções desses vetores sobre o plano xy, gerado por e_1 e e_2 são $\tilde{a}_1 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$, e $\tilde{a}_2 = \begin{pmatrix} 1 \\ 3 \end{pmatrix}$. Como $\Delta_2 > 0$, teremos que esses vetores são linearmente independentes e possuem a mesma orientação de e_1 e e_2. Ou seja, na segunda iteração do Algoritmo 2, se não houve quebra, o que aconteceu na prática, é que estamos aumentando a dimensão do conjunto de direções A conjugadas, mantendo a orientação.

Na terceira e última iteração concluímos: $D_3 = 1$ e $Z_3 = e_3$. Temos ainda, $\Delta_3 = 10$, ou seja, as três colunas de A são linearmente independentes e possuem a mesma orientação de e_1, e_2 e e_3.

Vamos imprimir generalidade ao que fizemos acima. Inicialmente, suponha, por simplicidade, que a ordem dos blocos é um, ou seja, que A_{ij} seja um escalar. Vamos

considerar ainda que $m = 3$ (ordem da matriz A) para maior clareza de ideias. Assim, existem três vetores, v_1, v_2 e $v_3 \in \mathbb{R}^3$ tais que:

$$A = [v_1\ v_2\ v_3].$$

Na primeira iteração do algoritmo é calculado $D_1 = A_{11}$. Portanto, para que a segunda iteração seja completada é necessário que $A_{11} \neq 0$, ou seja, a projeção de v_1 em e_1 deve ser não nula, o que é compreensível uma vez que nossa primeira direção conjugada é exatamente e_1. Como a matriz A é SBPD, temos $\Delta_1 > 0$, o que implica que a projeção de v_1 em e_1 tem a mesma direção de e_1. Observe que se fosse, $\Delta_1 > 0$, o algoritmo não quebraria, o que aconteceria apenas é que a orientação de e_1 inverteria.

O que aconteceria caso fosse $D_2 = 0$?

$D_2 = \frac{Z_2 \mathsf{T} A Z_1}{Z_1 \mathsf{T} A Z_1}$ e $Z_2^{\mathsf{T}} = [b_1\ 1\ 0 \ldots 0]$. Então $D_2 = 0$ se $Z_2^{\mathsf{T}} A Z_1 = b_1 a_{11} + a_{12} = 0 = a_{11} + b_1 a_{21} = Z_1 \mathsf{T} A Z_2$. O que implica que $b_1 = -\frac{a_{12}}{a_{11}} = -\frac{a_{11}}{a_{21}}$ ou

$$a_{11}^2 = a_{12} \cdot a_{21}. \tag{21}$$

Mas $D_2 = Z_2 \mathsf{T} A Z_2 = b_1^2 a_{11} + b_1 a_{12} + b_1 a_{21} + a_{22} = -b_1 a_{12} + b_1 a_{12} - a_{11} + a_{22} = 0$ ou

$$a_{11} = a_{22}. \tag{22}$$

Assim, juntando as Equações 21 e 22, $\Delta_2 = 0$.

Devemos que $\Delta_2 \neq 0$. Ora isto quer dizer que as projeções de v_1 e v_2 sobre o plano gerado por e_1 e e_2 são não nulas. Esta parece uma condição razoável para que se tenha uma terceira direção conjugada bem definida.

Além disso, $\Delta_2 > 0$, quer dizer que a projeção do segundo vetor v_2 no plano xy deve estar localizado no semiplano superior dos determinados por v_1

Considere o espaço vetorial $\mathbb{R}^{n \times m}$, com n divisível por m e sobre ele, vamos induzir uma estrutura em blocos, de maneira que cada matriz (vetor bloco) $b \in \mathbb{R}^{n \times m}$ tenha seus blocos iguais de ordem $m \times m$. Vamos denotar o espaço $\mathbb{R}^{n \times m}$ com essa estrutura em

blocos por $\mathbb{R}[n,m]$. Por exemplo, o vetor bloco,

$$A = \begin{pmatrix} A_{11} \\ A_{21} \end{pmatrix}$$

onde, $A_{11} = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$ e $A_{21} = \begin{pmatrix} 4 & 2 \\ 3 & 1 \end{pmatrix}$ é um elemento de $\mathbb{R}[4,2]$.

Os k vetores $E_i, i = 1, \ldots, k$ geram o espaço $\mathbb{R}[n,m]$ no sentido de que qualquer $c \in \mathbb{R}[n,m]$ é escrito na forma

$$C = E_1\alpha_1 + E_2\alpha_2 + \ldots + E_k\alpha_k,$$

onde cada $\alpha_i \in \mathbb{R}^{n\times n}$ (Note que $\mathbb{R}^{n\times n}$ não é um corpo). Por exemplo, qualquer vetor bloco em $\mathbb{R}[4,2]$ pode ser escrito como:

$$\begin{pmatrix} a & b \\ c & d \\ e & f \\ g & h \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} e & f \\ g & h \end{pmatrix}$$

Infelizmente, como $\mathbb{R}^{2\times 2}$ não é um corpo, $\mathbb{R}[4,2]$ não é um espaço vetorial no sentido tradicional, mas é um espaço vetorial um sentido generalizado, onde as matrizes fazem o papel escalares.

Assim, diremos que $\mathbb{R}[n,m]$ é um espaço vetorial generalizado.

Em analogia com o espaço vetorial, $\mathbb{R}^m$ tentaremos agora induzir uma operação sobre $\mathbb{R}[n,m]$ que seja o equivalente do produto interno para o $\mathbb{R}^m$.
Sobre $\mathbb{R}[n,m] \times \mathbb{R}[n,m]$ considere o operador *Produto Interno Generalizado*: Dada a matriz A SBPD de ordem $m \times m$, defina $g_A : \mathbb{R}[n,m] \times \mathbb{R}[n,m] \longrightarrow \mathbb{R}^{n\times n}$ por

$$g_A(u,v) = u^{\mathsf{T}} A v.$$

Para a i-ésima coluna u_i de u e a j-ésima coluna v_j de v temos que $(u^{\mathsf{T}}Av)_{ij} = u_i^{\mathsf{T}}Av_j = \| u_i \| \cdot \| v_j \| cos(\theta_{ij})$, onde θ_{ij} é o ângulo entre u_i e v_j. Portanto, a matriz $g_A(u, v)$, em cada componente, determina o produto interno (induzido por A) entre uma coluna de u e uma coluna de v.

Perceba que quando $n = 1$, temos o produto interno no sentido usual. Se $A = I$, então teremos o produto interno Euclidiano.

Para que a aplicação g_A seja realmente uma generalização do conceito de produto interno, devemos ter:

- g_A simétrica;
- g_A bilinear;
- Se $u \in \mathbb{R}[n, m]$, a aplicação g_A é positivo definida, no sentido de que a imagem $u\mathsf{T}Au$ de qualquer par (u, u) é uma matriz positivo definida.

Lema 1. *A aplicação g é simétrica.*

Demonstração. Sejam $u, v \in \mathbb{R}[n, m]$. Queremos provar que $g(u, v) = g(v, u)$.
Seja $Z \in \mathbb{R}^{n \times n}$. É fácil ver que $Z^{\mathsf{T}}AZ$ é uma matriz simétrica. Em particular, o ij-ésimo bloco de $Z^{\mathsf{T}}AZ$ é $B_{ij} = (Z^{\mathsf{T}}AZ)_{ij} = (Z^{\mathsf{T}}AZ)_{ji} = B_{ji}$.
Defina a matriz $Z = [u\ v\ E_3\ \ldots\ E_k]$. Então,

$$u^{\mathsf{T}}Av = B_{12} = B_{21} = v^{\mathsf{T}}Au.$$

□

Lema 2. *A aplicação g é bilinear.*

Demonstração. Temos $g(u + w, v) = (u + w)^{\mathsf{T}}Av = u^{\mathsf{T}}Av + w^{\mathsf{T}}Av = g(u, v) + g(w, v)$.
É óbvio ainda que $g(\lambda u, v) = \lambda g(u, v)$, onde $\lambda \in \mathbb{R}$. □

Lema 3. *Seja $u \in \mathbb{R}[n, m]$ com posto completo. A aplicação g é positivo definida, no sentido de que a imagem $u\mathsf{T}Au$ de qualquer par (u, u) é uma matriz positivo definida.*

Demonstração. Seja $x \in \mathbb{R}^m$ não nulo. Como u é matriz $n \times m$ tem posto completo, então $ux \in \mathbb{R}^m$ é não nulo e, portanto, $x^\mathsf{T} u^\mathsf{T} Aux$

□

Sobre o preenchimento da matriz Z observe a seguinte figura:

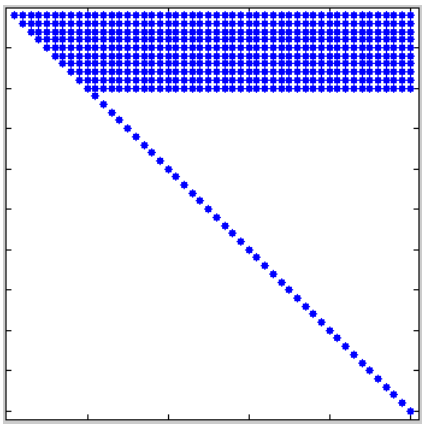


Figura 2 Matriz Z após algumas iterações.

Para melhor entender as demonstrações a seguir, é crucial entender o processo de preenchimento das matrizes Z, as quais preenchem linha bloco a linha bloco. Na Figura Figura 2, por simplicidade, consideramos o caso escarar e, assim, cada ponto azul representa um ponto de Z.

3.2 BAINV vs AINV

Naturalmente, o BAINV calcula a inversa exata da matriz A e, portanto, é muito caro. O cálculo exato então, para matrizes grandes é inviável, a menos que haja um descarte. Esse descarte é feito na parte estritamente superior de Z, ou seja, vamos zerar, segundo alguns critérios que serão vistos mais adiante, algumas entradas da matriz Z, para que a mesma fique esparsa. Na prática, o descarte é implementado na linha 7 do Algoritmo 23. No caso bloco, a matriz D também pode ter pivôs densos. Portanto, é importante implementar na linha 3 do Algoritmo 23 um descarte para essas matrizes. Eventualmente, podemos aplicar também, um descarte na inversa de D_i, que pode ser densa, mesmo que D_i seja esparsa. Para um pivô D_i o descarte é feito fora de sua diagonal. O tipo mais simples de descarte é por tolerância, quando descartamos uma entrada se sua magnitude é menor que um parâmetro pré estabelecido.

Baseado em um exemplo similar dado em [11] (no exemplo original a matriz é de ordem 3), vamos considerar a matriz

$$A = \begin{pmatrix} 2.00 & 0.40 & 0.10 & 0.00 \\ 0.40 & 1.08 & 2.00 & 0.00 \\ 0.10 & 2.00 & 3.96 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{pmatrix},$$

e uma tolerância de descarte igual 0.06 para a parte estritamente superior de Z. Aplicando o BAINV em A no caso de bloco de ordem 1 (o qual coincide com o algoritmo dado em [11]), há quebra em D_3:

$$D = \begin{pmatrix} 2.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 1.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & NaN \end{pmatrix}.$$

Agora, a título de comparação, usaremos no caso bloco o mesmo tipo de descarte. Considere novamente a matriz A com tolerância de descarte igual a 0.06 para a parte estritamente superior de Z. Vamos reaplicar o BAINV, mas desta vez com bloco de ordem 2: assim, primeiramente repartimos a matriz A em

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix},$$

onde

$$A_{11} = \begin{pmatrix} 2.00 & 0.40 \\ 0.40 & 1.08 \end{pmatrix}, A_{12} = \begin{pmatrix} 0.10 & 0.00 \\ 2.00 & 0.00 \end{pmatrix}. A_{21} = \begin{pmatrix} 0.10 & 2.00 \\ 0.00 & 0.00 \end{pmatrix}.$$

$$e \quad A_{22} = \begin{pmatrix} 3.96 & 0.00 \\ 0.00 & 1.00 \end{pmatrix}.$$

Nesse caso, o BAINV produz as matrizes:

$$D = \begin{pmatrix} 2.00 & 0.40 & 0.00 & 0.00 \\ 0.40 & 1.08 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.04 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{pmatrix}$$

e

$$Z = \begin{pmatrix} 1.00 & 0.00 & 0.04 & 0.00 \\ 0.00 & 1.00 & -1.98 & 0.00 \\ 0.00 & 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{pmatrix}.$$

Repare que não houve quebra. Essa diferença é explicada pelo Teorema 2, pois os menores principais necessários para o BAINV eram diferentes de zero nesse caso. Ou seja, a matriz não precisa ser SPD, basta ter menores principais positivos das ordens dos blocos. O critério de Sylvester diz que uma matriz simétrica é SP se e somente se todos os menores principais são positivos.

Para provar-se BAINV não quebra precisa-se olhar para menos menores principais, em relação ao AINV. No entanto, é necessário provar a não singularidade de uma matriz pivô; enquanto que no AINV, basta provar que o elemento da diagonal é não nulo. Ou seja, há menos restrições com relação aos blocos, porém, cada uma das restrições é mais difícil, matematicamente falando.

É importante citar que a versão estável do AINV escalar é livre de quebra no caso da matriz acima. A versão estável, troca $D_i^{(i-1)} = a_i^{\mathsf{T}} Z_i$ por dois produtos: $v_i = AZ_i$ e $D_j^{(i)} = Z_i^{\mathsf{T}} v_i$.

4 ROBUSTEZ DO DESCARTE PARA MATRIZES $\mathcal{M}$

Nesta seção, vamos demonstrar que o BAINV pode ser aplicado com descarte nas matrizes $\mathcal{M}$. Para isso, vamos utilizar algumas caracterizações dessa classe de matrizes, que estão reunidas em [65]. A classe das $\mathcal{M}$ matrizes foi estudada sistematicamente por Ostrowski [64]. Em [65], foram propostas 40 caracterizações para $\mathcal{M}$ matrizes não singulares.

Esse resultado já é conhecido quando a ordem do bloco é igual a um, ou seja, quando estamos tratando de matrizes escalares. A seguir são introduzidos os conceitos necessários e logo após, alguns Lemas, que serão utilizados no principal resultado dessa Seção.

Precisamos de algumas definições iniciais.

Definição 10. *Uma matriz A, de ordem n, é uma $\mathcal{Z}$ matriz (ou, simplesmente, $A \in \mathcal{Z}$) se $a_{ij} \leq 0$ para $i \neq j$.*

Definição 11. *Seja $\rho(B)$ o espectro[1] de B. A matriz $A \in \mathcal{Z}$, de ordem n, pertence à classe das $\mathcal{M}$ matrizes (ou, simplesmente, $A \in \mathcal{M}$), se $A = sI - B$, onde $B = (b_{ij})$, com $b_{ij} \geq 0$, $1 \leq i, j \leq n$, e $s \geq \rho(B)$ e A é não singular.*

Por essas definições, $\mathcal{M} \subset \mathcal{Z}$.

Um dos Lemas mais importantes para os nossos propósitos, nessa e na próxima seção é:

Lema 2 (Caracterização N39 em [65])**.** *Seja A, de ordem n, uma $\mathcal{Z}$ matriz. Então A é uma $\mathcal{M}$ matriz não singular se, e somente se, sua diagonal é positiva e existe uma matriz diagonal positiva tal que AD é estritamente diagonal dominante, ou seja,*

$$a_{ii}d_i \geq \sum_{j \neq i} |a_{ij}|d_j,$$

para $i = 1, \ldots, n$.

Os próximos dois lemas nos dão características dos pivôs vindos do BAINV no caso de, respectivamente, matrizes SPD e matrizes $\mathcal{M}$. Essas propriedades serão fundamentais no principal resultado dessa seção.

[1] $\rho(B) = \text{máx}\{|\lambda|; \lambda \in \Lambda(B)\}$, onde $\Lambda(B)$ é o conjunto dos autovalores de B.

Lema 3. *Seja A uma matriz SPD*[2] *e D_r, $1 \leq r \leq k$, um pivô produzido pelo Algoritmo 23. Então, D_r é uma matriz SPD.*

Demonstração. De $A^\mathsf{T} = A$, segue que $D^\mathsf{T} = (Z^\mathsf{T}AZ)^\mathsf{T} = Z^\mathsf{T}AZ = D$. Seja $x \in \mathbb{R}^m \setminus \{0\}$. Pelo Algoritmo 23, $D_r = Z_r^\mathsf{T}AZ_r$ e

$$x^\mathsf{T}D_r x = x^\mathsf{T}Z_r^\mathsf{T}AZ_r x.$$

Note que, como Z é não singular, $Z_r \in \mathbb{R}^{n\times m}$, com $n > m$, tem posto completo. Portanto, $u = Z_r x \neq 0$ e

$$x^\mathsf{T}D_r x = u^\mathsf{T}Au > 0,$$

já que A é SPD. Estamos considerando a desigualdade calculada compenente a componente. □

Lema 4. *Seja $A \in \mathcal{M}$ e SPD. Sejam D_r, $1 \leq r \leq k$, os pivôs produzidos pelo Algoritmo 23. Se $Z \geq 0$, então $D_r \in \mathcal{M}$.*

Demonstração. Primeiramente, provaremos que $D_r \in \mathcal{Z}$, para $1 \leq r \leq k$. De fato, pelo Algoritmo 23, para $1 \leq r \leq k$

$$D_r = \sum_{\ell=1}^{r-1} A_{r\ell}Z_{\ell r} + A_{rr}.$$

Sabemos que $A \in \mathcal{M}$, $A_{rr} \in \mathcal{Z}$ e $A_{rl} \leq 0$, para $r \neq l$. Assim, uma vez que $Z \geq 0$, temos que D_r é uma soma de $\mathcal{Z}$ matrizes.

Agora, pela Proposição 3, D_r é SPD, logo, pelo critério de Sylvester [43], D_r possui todos os menores principais positivos. Usando a Caracterização D_{11} apresentada em [65], concluímos que $D_r \in \mathcal{M}$ e é não singular. □

Um comentário importante aqui é que, quando A não é simétrica e não singular, mas é $\mathcal{M}$, ela possui todos os seus menores principais positivos, já que é matriz $\mathcal{M}$ não singular. Assim, fazendo-se as restrições das matrizes W, A e Z às suas submatrizes principais líderes de ordem i, temos $W_i^\mathsf{T}A_iZ_i = D_i$ e aplicando-se o determinante em ambos os lados, veremos que D também possui todos os seus menores positivos e, portanto, também valerá o Lema 4.

[2]Matriz SPD é simétrica, $A = A^\mathsf{T}$, e positivo definida, $x^\mathsf{T}Ax > 0$, caso $x \neq 0$.

Lema 5. *Se B e C são matrizes $n \times p$ tais que $B \leq C$ e A é matriz positiva $q \times n$ então $AB \leq AC$.*

Demonstração. Fixe as colunas B_j e C_j de B e C respectivamente. De $B \leq C$ segue que $B_j \leq C_j$. Fixe ainda a linha A_i de A. Como A é positivo segue que $A_iB_j \leq A_iC_j$ e, portanto, $AB \leq AC$. □

Lema 6. *Sejam A e $B \in \mathcal{M}$, de ordem n e não singulares, com $A \leq B$. Então, $B^{-1} \leq A^{-1}$.*

Demonstração. De $A \leq B$ temos, pelo Lema 5, $I_n = A^{-1}A \leq A^{-1}B$, pois $A^{-1} \geq 0$ pela Caracterização F_{15}, [65]. Agora, utilizando o fato de que $B^{-1} \geq 0$ teremos, multiplicando pela direita, que $B^{-1} \leq A^{-1}$. □

Reunimos as condições técnicas para enunciar e demonstrar um dos principais resultados deste trabalho.

Durante o Teorema 8 usaremos a seguinte notação: quando o BAINV for aplicado com descarte usaremos a notação $\widetilde{Z}$ e $\widetilde{D}$ para denotar as matrizes de saída do Algoritmo 7 e Z e D para quando o Algoritmo 7 for aplicado sem descarte.

Teorema 8. *Seja $A \in \mathcal{M}$ matriz em blocos, SPD e de ordem n, com uma partição em blocos de dimensão de bloco igual a k. Então,*

$$\det(\widetilde{D}_i) \neq 0.$$

Demonstração. Provaremos por indução que

$$H(r) := \begin{cases} M_j^{(r-1)} \leq \widetilde{M}_j^{(r-1)} \leq 0, & j > r, \quad (23a) \\ 0 \leq \widetilde{Z}_j^{(r-1)} \leq Z_j^{(r-1)}, & j > r, \quad (23b) \\ D_r \leq \widetilde{D}_r, & \quad (23c) \\ \widetilde{D}_r \in \mathcal{M}, & \text{com } \det(\widetilde{D}_r) \neq 0. \quad (23d) \end{cases}$$

para $1 \leq r \leq k$.

Para $r = 1$, temos que, uma vez que $A \in \mathcal{M}$ e não sendo realizado descarte na

matriz inicial $Z = I$,

$$A_{1j} = M_j^{(0)} \leq \widetilde{M}_j^{(0)} \leq 0, \text{ para } 2 \leq j \leq k.$$

Para as colunas da matriz Z, $\widetilde{Z}_j^{(0)} = Z_j^{(0)} = E_j \geq 0$.

Com relação ao pivô, $D_1 = A_{11} \leq \widetilde{P}_1$ (nesse ponto já ocorreu descarte, mas apenas fora da diagonal do pivô, pois Z_1 é a primeira coluna bloco da identidade).

Note que $\widetilde{D}_1$ tem diagonal positiva (pois é maior que uma matriz $\mathcal{M}$) e

$$(\widetilde{D}_1)_{ij} := \begin{cases} (D_1)_{ij} \\ 0 \end{cases}$$

Assim, $\widetilde{D}_1$ é matriz $\mathcal{Z}$. Agora, como $D_1 \in \mathcal{M}$ tem-se que existe matriz diagonal $D, m \times m$ tal que $(D_1)_{ii}di > \sum_{j \neq i} |(D_1)_{ij}|dj$ para $i = 1, \ldots, m$. Portanto, como $(\widetilde{D}_1)_{ii} \geq (D_1)_{ii}$ e $|(D_1)_{ij}| \geq |(\widetilde{D}_1)_{ij}|$, temos que $(\widetilde{D}_1)_{ii}di > \sum_{j \neq i} |(\widetilde{D}_1)_{ij}|dj$, para $i = 1, \ldots, m$. Portanto, pelo Lema 2, $\widetilde{D}_1$ é uma matriz $\mathcal{M}$ não singular.

Vamos provar que H_s ser verdadeira, para $2 \leq s < r$, implica que H_r será verdadeira.

Antes de começarmos pelos multiplicadores, vale lembrar o seguinte fato: até a $(r-1)$-ésima iteração $Z_j^{(r-1)}, j > r$ só preencheu nas $r-1$ primeiras posições bloco e na posição bloco j, pois trata-se de uma matriz identidade de ordem m, como indica o esquema a seguir:

$$Z_j^{(r-1)} = (\times, \cdots, \times, 0, 0, \cdots, I, 0, \cdots, 0)^{\mathsf{T}}$$

Com relação aos diversos multiplicadores, $M_j^{(r-1)}$, podemos escrever

$$M_j^{(r-1)} = A_r^{\mathsf{T}} Z_j^{(r-2)} = \sum_{\ell=1}^{r-2} A_{r\ell} Z_{\ell j}^{(r-2)} + A_{rj}. \tag{24}$$

Pela hipótese de indução, $Z_j^{(r-2)} \geq \widetilde{Z}_j^{(r-2)} \geq 0$ e $A \in \mathcal{M}$, segue que $M_j^{(r-1)} \leq \widetilde{M}_j^{(r-1)} \leq 0$, para $j > r-1$, pois $A_{ij} \leq 0$, para $i \neq j$.

Já para as colunas de Z, vamos provar que $0 \leq \widetilde{Z}_j^{(r-1)} \leq Z_j^{(r-1)}$, para $j > r-1$. Temos que $\widetilde{Z}_j^{(r-1)} = \widetilde{Z}_j^{(r-2)} - \widetilde{Z}_{r-1}^{(r-2)} \widetilde{D}_{r-1}^{-1} \widetilde{M}_j^{(r-1)} \geq 0$. Então, como $0 \leq \widetilde{Z}_j^{(r-2)} \leq Z_j^{(r-2)}$,

$0 \leq -\widetilde{M}_j^{(r-1)} \leq -M_j^{(r-1)}$ e, pelo Lema 6, $0 \leq \widetilde{D}_{r-1}^{-1} \leq D_{r-1}^{-1}$ segue do Lema 5 que

$$0 \leq \widetilde{Z}_j^{(r-1)} \leq Z_j^{(r-1)}.$$

Com relação aos pivôs. Como Z é triangular superior e tem blocos identidade no bloco diagonal principal, então

$$D_r = A_r^{\mathsf{T}} Z_r^{(r-1)} = \sum_{\ell=1}^{r-1} A_{r\ell} Z_{\ell r}^{(r-1)} + A_{rr}.$$

Agora, já sabemos que $Z_r^{(r-1)} \geq \widetilde{Z}_r^{(r-1)} \geq 0$. Também, $A \in \mathcal{M}$, de maneira que, $A_{r\ell} \leq 0$, para $r \neq \ell$, assim, $A_{r\ell} Z_{\ell r} \leq A_{r\ell} \widetilde{Z}_{\ell r}$. Assim, $D_r \leq \widetilde{D}_r$. Havendo ainda descarte no pivô, só podemos aumentar $\widetilde{D}_r$ visto que é matriz $\mathcal{Z}$.

Pelo Lema 4, $D_r \in \mathcal{M}$ e, assim, $\widetilde{D}_r \in \mathcal{Z}$ e possui entradas diagonais estritamente positivas. Assim sendo, existe matriz diagonal $D, m \times m$, tal que $(D_r)_{ii} di > \sum_{j \neq i} |(D_r)_{ij}| dj$ para $i = 1, \ldots, m$. Portanto, como $(\widetilde{D}_r)_{ii} \geq (D_r)_{ii}$ e $|(D_r)_{ij}| \geq |(\widetilde{D}_r)_{ij}|$ tem-se que $(\widetilde{D}_r)_{ii} di > \sum_{j \neq i} |(\widetilde{D}_r)_{ij}| dj$ para $i = 1, \ldots, m$. Portanto, pelo Lema 2, $\widetilde{D}_r$ é uma matriz $\mathcal{M}$ não singular. □

Não podemos considerar que $\widetilde{\widetilde{Z}}_i$ é uma cópia de Z_i com apenas algumas entradas anuladas. A construção dos vetores bloco $\widetilde{\widetilde{Z}}_i$ e Z_i depende dos vetores $\widetilde{\widetilde{Z}}_j$ e Z_j, com $1 \leq j < i$, respectivamente. Não há como calcular $\widetilde{\widetilde{Z}}_i$ diretamente a partir de Z_i. Outro fato relevante é o tratamento dos pivôs, já que no caso escalar é apenas um número real e aqui é uma matriz quadrada. Provar que uma matriz é não singular é certamente mais complicado do que mostrar que um número é não nulo.

Por exemplo, considere a matriz a seguir, com uma estrutura em blocos de ordem 2,

$$\begin{pmatrix} 17 & -1 & -1 & -1 & -1 & -1 \\ -1 & 16 & -1 & -1 & -1 & -1 \\ -1 & -1 & 17 & -1 & -1 & -1 \\ -1 & -1 & -1 & 16 & -1 & -1 \\ -1 & -1 & -1 & -1 & 17 & -1 \\ -1 & -1 & -1 & -1 & -1 & 16 \end{pmatrix}$$

Sem descarte, o AINV produz $Z_3 = \begin{pmatrix} 0.0720 & 0.0720 \\ 0.0763 & 0.0763 \\ 0.0720 & 0.0720 \\ 0.0763 & 0.0763 \\ 1.0000 & 0.0000 \\ 0.0000 & 1.0000 \end{pmatrix}$. Agora, com descarte de 0.063 temos $\widetilde{Z}_3 = \begin{pmatrix} 0.0000 & 0.0000 \\ 0.0756 & 0.0756 \\ 0.0675 & 0.0675 \\ 0.0714 & 0.0714 \\ 1.0000 & 0.0000 \\ 0.0000 & 1.0000 \end{pmatrix}$.

Mais uma vez, a demonstração feita pode ser generalizada para o caso não simétrico.

5 ROBUSTEZ DO DESCARTE PARA MATRIZES $\mathcal{H}$

Definição 12 (Matriz $\boldsymbol{\mathcal{H}}$)**.** *Uma matriz A, de ordem n, pertence à classe de matrizes $\mathcal{H}$ (ou, simplesmente, $A \in \mathcal{H}$) se $\widehat{A} = (\widehat{a}_{ij}) \in \mathcal{M}$, onde*

$$\widehat{a}_{ij} := \begin{cases} -|a_{ij}|, & \text{se } i \neq j, \\ a_{ii}, & \text{se } i = j. \end{cases}$$

*$\widehat{A}$ é denominada de **matriz $\mathcal{M}$ associada** à A.*

Definição 13. *Seja A de ordem n, a **matriz de comparação de A**, $\mathcal{C}(A) = (c_{ij})$, de ordem n, é definida por*

$$c_{ij} := \begin{cases} -|a_{ij}|, & \text{se } i \neq j, \\ |a_{ii}|, & \text{se } i = j. \end{cases}.$$

Repare que $A \in \mathcal{M} \Rightarrow A = \mathcal{C}(A)$.

Lema 7. *Se $A \in \mathcal{Z}$, $B \in \mathcal{M}$ não singular e $B \leq A$, então $A \in \mathcal{M}$ e A é não singular.*

Demonstração. Para esta demonstração faremos o uso do Lema 2, o qual propõe uma caracterização para $\mathcal{M}$ matrizes não singulares.

Temos que $a_{ii} \geq b_{ii} > 0$ e $b_{ij} \leq a_{ij} \leq 0$. Portanto, a matriz diagonal positiva tal que BD é estritamente diagonal dominante é tal que

$$d_i a_{ii} > \sum_{\substack{j=1 \\ j \neq i}}^{m} d_j |a_{ij}|,$$

e A é matriz $\mathcal{M}$. Logo, AD é estritamente diagonal positiva e D é uma matriz diagonal positiva. Assim, vale o Lema 2 e A é uma matriz $\mathcal{M}$ e não singular. □

Lema 8. *(Desigualdade de Ostrowsky) Se $A \in \mathcal{H}$ é não singular então $|A^{-1}| \leq \mathcal{C}(A)^{-1}$.*

Demonstração. Veja [54]. □

Lema 9. *Seja $A \in \mathcal{H}$. Se $\mathcal{C}(A)$ é não singular, então A é não singular.*

Demonstração. Veja [24]. □

Lema 10 (Teorema 4 em [65]). *Seja $A \in \mathbb{C}^{n\times n}$, tal que $a_{ii} \neq 0,\ 1 \leq i \leq n$. $A \in \mathcal{H}$ é não singular se, e somente se, para cada matriz $B \in \mathbb{C}^{n\times n}$, com $\mathcal{C}(B) \geq \mathcal{C}(A)$, B é não singular.*

Teorema 9. *Sejam $A \in \mathcal{H}$ e $\widehat{A}$ a matriz $\mathcal{M}$ associada à A, com $A \in \mathbb{R}^{n\times n}$ e $n = m \times k$, onde k é a ordem dos blocos em A. Sejam D_i e $\widehat{D}_i$ os pivôs produzidos por A e $\widehat{A}$ pelo BAINV sem descarte. Considere ainda os pivôs $\widetilde{D}_i$ produzidos pelo BAINV com descarte. Todos os pivôs tem ordem k. Então,*

$$\det(\widetilde{D}_i) \neq 0.$$

Demonstração. Levando em conta a notação apresentada no Algoritmo 23, vamos usar indução em r. Os vetores em bloco Z_i, $\widetilde{Z}_i$ e $\widehat{Z}_i$ e os multiplicadores M_i, $\widetilde{M}_i$ e $\widehat{M}_i$ seguem a notação proposta para os pivôs.

$$H(r) := \begin{cases} |\widetilde{M}_j^{(r-1)}| \leq -\widehat{M}_j^{(r-1)}, & j > r, \quad (25a) \\ |\widetilde{Z}_j^{(r-1)}| \leq \widehat{Z}_j^{(r-1)}, & j \geq r, \quad (25b) \\ \widehat{D}_r \leq \mathcal{C}(\widetilde{D}_r), & \det(\widetilde{D}_r) \neq 0, \quad (25c) \end{cases}$$

para $1 \leq r \leq k$.

Para $r = 1$ e $j > 1$, $|M_j^{(0)}| = |A_{1j}| = -\widehat{A}_{1j} = -\widehat{M}_j^{(0)}$. Portanto, $|\widetilde{M}_j^{(0)}| \leq -\widehat{M}_j^{(0)}$. Com relação a matriz Z, na primeira iteração, $Z_i^{(0)} = E_i$. Como não há descarte nessa etapa, $|\widetilde{Z}_j^{(r)}| \leq \widehat{Z}_j^{(r)}$.

Já para os pivôs $D_1 = A_{11}$ e $\widehat{D}_1 = \widehat{A}_{11}$, portanto, $\widehat{D}_1 = \mathcal{C}(D_1)$. Como

$$(\widetilde{D}_1)_{ij} := \begin{cases} (D_1)_{ij} \\ \\ 0 \end{cases}, \text{ para } i \neq j, \text{ e } (\widetilde{D}_1)_{ii} := (D_1)_{ii},$$

já que não há descarte nos elementos da diagonal principal, temos $\widehat{D}_1 \leq \mathcal{C}(\widetilde{D}_1)$. Como $\widehat{D}_1 \in \mathcal{M}$ e $\mathcal{C}(\widetilde{D}_1) \in \mathcal{Z}$, segue, do Lema 7, que $\mathcal{C}(\widetilde{D}_1) \in \mathcal{M}$ e é não singular. Logo, $\widetilde{D}_1 \in \mathcal{H}$, pois suas diagonais são as da matriz original, e , do Lema 9, é não singular.

Vamos provar que $H_{r-1} \implies H_r$ é verdadeira. Para os multiplicadores, por

analogia à (24),

$$|\widetilde{M}_r^{(r-1)}| = |\sum_{l=1}^{r-2} \widetilde{A}_{rl}\widetilde{Z}_{lj}^{(r-2)} + \widetilde{A}_{rj}| \leq \sum_{l=1}^{r-2} |\widetilde{A}_{rl}||\widetilde{Z}_{lj}^{(r-2)}| + |\widetilde{A}_{rj}| \leq$$
$$\leq \sum_{l=1}^{r-2} -\widehat{A}_{rl}\widehat{Z}_{lj}^{(r-2)} - \widehat{A}_{rj} = -\widehat{M}_j^{(r-1)}.$$

Analisando as colunas da matriz Z. Como, pela hipótese de indução, $\widehat{D}_{r-1} \leq \mathcal{C}(\widetilde{D}_{r-1})$, temos, pelo Lema 7, que $\mathcal{C}(\widetilde{D}_{r-1})$ é não singular e, pelo Lema 9, que $\widetilde{D}_{r-1}$ é matriz $\mathcal{H}$ e não é singular. Sendo assim, dos Lemas 6 e 8

$$|\widetilde{D}_{r-1}^{-1}| \leq \mathcal{C}(\widetilde{D}_{r-1})^{-1} \leq \widehat{D}_{r-1}^{-1}.$$

Com isso, podemos concluir que:

$$|\widetilde{Z}_j^{(r-1)}| = |\widetilde{Z}_j^{(r-2)} - \widetilde{Z}_{r-1}\widetilde{D}_{r-1}^{-1}\widetilde{M}_j^{(r-2)}| \leq |\widetilde{Z}_j^{(r-2)}| + |\widetilde{Z}_{r-1}||\widetilde{D}_{r-1}^{-1}||\widetilde{M}_j^{(r-2)}| \leq$$
$$\leq \widehat{Z}_j^{(r-2)} - \widehat{Z}_{r-1}\widehat{D}_{r-1}^{-1}\widehat{M}_j^{(r-2)} = \widehat{Z}_j^{(r-1)}.$$

Finalmente, do Algoritmo 7, temos

$$\widetilde{D}_r = A_{rr} + \sum_{l=1}^{r-1} A_{rl}\widetilde{Z}_{lr}. \tag{26}$$

Queremos mostrar que $\widehat{D}_r \leq \mathcal{C}(\widetilde{D}_r)$. Para isso, devemos comparar $\widehat{D}_r$ e $\mathcal{C}(\widetilde{D}_r)$ como matrizes escalares. Analisando a equação (26), vamos estudar os elementos fora da diagonal principal de $\widehat{D}_r$,

$$(\widehat{D}_r)_{ij} = (\widehat{A}_{rr})_{ij} + \sum_{l=1}^{r-1} (\widehat{A}_{rl}\widehat{Z}_{lr})_{ij}, i \neq j,$$

e os de sua diagonal principal,

$$(\widehat{D}_r)_{ii} = (\widehat{A}_{rr})_{ii} + \sum_{l=1}^{r-1} (\widehat{A}_{rl}\widehat{Z}_{lr})_{ii}.$$

Primeiramente, vamos entender como é a construção das entradas de $\widetilde{D}_r$ fora da

sua diagonal principal. Como $(\widehat{A}_{rr}) \in \mathcal{M}$, $(\widehat{A}_{rr})_{ij} \leq 0$; assim,

$$(\widehat{A}_{rr})_{ij} \leq (\widetilde{A}_{rr})_{ij} \leq -(\widehat{A}_{rr})_{ij}.$$

Pelas definições das matrizes, $\widehat{A}_{rl} \leq 0$ e $\widehat{A}_{rl} \leq \widetilde{A}_{rl} \leq -\widehat{A}_{rl}$, para $l \neq r$. Pela hipótese de indução, $|Z_{lr}| \leq \widehat{Z}_{lr}$, para $1 \leq l < r$. Portanto, temos

$$(\widehat{A}_{rr})_{ij} + \sum_{l=1}^{r-1} (\widehat{A}_{rl}\widehat{Z}_{lr})_{ij} \leq (\widetilde{A}_{rr})_{ij} + \sum_{l=1}^{r-1} (\widetilde{A}_{rl}\widetilde{Z}_{lr})_{ij} \leq -(\widehat{A}_{rr})_{ij} - \sum_{l=1}^{r-1} (\widehat{A}_{rl}\widehat{Z}_{lr})_{ij}$$

$$\Rightarrow -\left|(\widehat{A}_{rr})_{ij} + \sum_{l=1}^{r-1} (\widehat{A}_{rl}\widehat{Z}_{lr})_{ij}\right| \leq -\left|(\widetilde{A}_{rr})_{ij} + \sum_{l=1}^{r-1} (\widetilde{A}_{rl}\widetilde{Z}_{lr})_{ij}\right|. \quad (27)$$

Agora, vamos provar que para a diagonal principal de $\widetilde{D}_r$, calculada na equação (26), vale que $\widehat{D}_r \leq \mathcal{C}(\widetilde{D}_r)$. Agora, $|(\widetilde{A}_{rr})_{ii}| = (\widehat{A}_{rr})_{ii}$ e $\widehat{A}_{rl}\widehat{Z}_{lr} \leq \widetilde{A}_{rl}\widetilde{Z}_{lr}$, pois $\widehat{A}_{rl}\widehat{Z}_{lr}$ é negativo.

Nesse ponto, há dois casos a se considerar: $(\widetilde{A}_{rr})_{ii}$ e $\sum_{l=1}^{r-1} (\widetilde{A}_{rl}\widetilde{Z}_{lr})_{ii}$ ou possuem sinais iguais ou possuem sinais diferentes.

Caso tenham sinais diferentes, teremos

$$(\widehat{D}_r)_{ii} = (\widehat{A}_{rr})_{ii} - |\sum_{l=1}^{r-1} (\widehat{A}_{rl}\widehat{Z}_{lr})_{ii}| \leq ||(\widetilde{A}_{rr})_{ii}| - |\sum_{l=1}^{r-1} (\widetilde{A}_{rl}\widetilde{Z}_{lr})_{ii}|| = |(\widetilde{D}_r)_{ii}|.$$

Já se $(\widetilde{A}_{rr})_{ii}$ e $\sum_{l=1}^{r-1} (\widetilde{A}_{rl}\widetilde{Z}_{lr})_{ii}$ possuírem sinais iguais, então $(\widehat{D}_r)_{ii} = (\widehat{A}_{rr})_{ii} - |\sum_{l=1}^{r-1} (\widehat{A}_{rl}\widehat{Z}_{lr})_{ii}| \leq ||(\widetilde{A}_{rr})_{ii}| + |\sum_{l=1}^{r-1} (\widetilde{A}_{rl}\widetilde{Z}_{lr})_{ii}|| = |(\widetilde{D}_r)_{ii}|$.

Portanto, $\widehat{D}_r \leq \mathcal{C}(\widetilde{D}_r)$. Logo, como $\widehat{D}_r$ é uma matriz $\mathcal{M}$ e $\mathcal{C}(\widetilde{D}_r)$ é matriz $\mathcal{Z}$, segue do Lema 7 que $\mathcal{C}(\widetilde{D}_r)$ é matriz $\mathcal{M}$ e não é singular. Assim, pelo Lema 9, $\widetilde{D}_r$ é uma matriz $\mathcal{H}$ e não é singular. □

Devemos notar que, em nenhum dos lemas e nem mesmo na demonstração do Teorema 9 a simetria foi uma condição *sine qua non* para realização da demonstração, de modo que a mesma é passível generalização para o caso não simétrico.

6 AVANÇOS E FUTURAS PESQUISAS

Além do BAINV vamos propor algumas variações desses métodos de precondicionamento que terão suas experimentações feitas em trabalhos futuros. Exibimos aqui a generalização de alguns métodos apresentados e vamos apresentar adicionalmente as opções de descartes que podem ser utilizadas mais adiante em trabalhos futuros.

Algumas dessas variações que apresentaremos são livre de quebras, pois usam como pano de fundo, a versão estável SBAINV. Este é o caso do BRIF, apresentado no Algoritmo 24.

Algoritmo 24: BRIF

Let A a nonsingular matrix and $C = A^T A$ and a preset block structure in C.
$Z_i^{(0)} \leftarrow E_i, i = 1, ..., n;$
for $i \leftarrow 1$ **to** n **do**
  $V_i \leftarrow AZ_i^{(i-1)};$
  for $j \leftarrow i$ **to** n **do**
    $P_j^{(i-1)} \leftarrow V_i^T (AZ_j^{(i-1)});$
  if $i \leftarrow n$ **then**
    go to 13
  for $j \leftarrow i + 1$ **to** n **do**
    $Z_j^{(i)} \leftarrow Z_j^{(i-1)} - Z_i^{(i-1)} (P_i^{(i-1)})^{-1} P_j^{(i-1)};$
$D \leftarrow diag(P_1, P_2, ..., P_n)$ and $Z \leftarrow (Z_1, Z_2, ..., Z_n)$.

Naturalmente, essa versão é livre de quebra. Caso use a versão não estável, com menos custo computacional, o que podemos garantir é que para matrizes $\mathcal{H}$ o Algoritmo também é livre de quebra.

Baseado no trabalho de Rafiei e Toutounian, [66], podemos, através de algumas adaptações, propor uma variação do SAINV, para matrizes não simétricas e positivo definidas em bloco, que também é livre de quebras.

Primeiramente, vamos relembrar o AINV em bloco não simétrico e poderemos tirar algumas conclusões (no Algoritmo 25, C_i^T representa a i-ésima bloco coluna de A^{T}).

A primeira importante conclusão que queremos chegar é que:

$$P_i^{(i-1)} = W_i^T A Z_i = Z_i^T A W_i = Q_i.$$

Essa relação se demonstra com perfeita analogia ao que fizemos no capítulo 2.

Algoritmo 25: Block Biconjugation Process

1 $W_i^{(0)} = Z_i^{(0)} = E_i, i = 1, \ldots, n;$
2 **for** $i \leftarrow 1$ **to** n **do**
3 **for** $j \leftarrow 1$ **to** n **do**
4 $P_j^{(i-1)} = A_i^T Z_j^{(i-1)};$
5 $Q_j^{(i-1)} = C_i^T W_j^{(i-1)};$
6 **if** $i \leftarrow n$ **then**
7 go to 15
8 **for** $j \leftarrow i+1$ **to** n **do**
9 $Z_j^{(i)} := Z_j^{(i-1)} - Z_i^{(i-1)}(P_i^{(i-1)})^{-1} P_j^{(i-1)};$
10 $W_j^{(i)} := W_j^{(i-1)} - W_i^{(i-1)}(Q_i^{(i-1)})^{-1} Q_j^{(i-1)};$
11 $D = Diag(P_1, P_2, ..., P_n)$, $Z = (Z_1, Z_2, ..., Z_n)$ and $W = (W_1, W_2, ..., W_n)$.

A segunda conclusão diz respeito às seguintes igualdades:

$$P_j^{(i-1)} = W_i^T A E_j \text{ e } Q_j^{(i-1)} = Z_i^T A E_j,$$

para $1 \leq i \leq j \leq k$.

Para provar essas relações faremos uso do Lema 1, que nos diz que:

$$Z_j^{(i-1)} = E_j - \sum_{s=1}^{j-1} Z_s (P_s)^{-1} P_j^{(s)}.$$

Assim, como $P_j^{(i-1)} = W_i^T A Z_j^{(i-1)}$ e, sabendo que $W_i^T A Z_s = 0$, para $j > s$ obtemos o resultado desejado. A relação $Q_j^{(i-1)} = Z_i^T A E_j$ se obtém analogamente.

Teorema 10. *Sejam $P_j^{(i-1)}$ e $Q_j^{(i-1)}$ pivôs produzidos pelo Algoritmo 25. Então:*

$$P_j^{(i-1)} = A_{ij} - \sum_{k=1}^{(i-1)} [Q_i^{(k-1)}]^{\mathsf{T}} (P_k^{(k-1)})^{-1} P_j^{(k-1)}$$

e

$$Q_j^{(i-1)} = A_{ji} - \sum_{k=1}^{(i-1)} [P_i^{(k-1)}]^{\mathsf{T}} (Q_k^{(k-1)})^{-1} Q_j^{(k-1)}$$

Demonstração. Sabendo que $P_j^{(i-1)} = E_i^T A Z_j^{(i-1)}$ e $Z_j^{(i-1)} = E_j - \sum_{k=0}^{i-2} Z_{k+1}^{(k)} (P_{k+1})^{-1} P_j^{(k)}$, temos

$$P_j^{(i-1)} = E_i^T A (E_j - \sum_{k=0}^{i-2} Z_{k+1}^{(k)} (P_{k+1})^{-1} P_j^{(k)}).$$

Assim,

$$P_j^{(i-1)} = A_{ij} - \sum_{k=0}^{i-2} [Q_i^{(k-1)}]^{\mathsf{T}} (P_{k+1})^{-1} P_j^{(k)},$$

que é equivalente a:

$$P_j^{(i-1)} = A_{ij} - \sum_{k=1}^{i-1} [Q_i^{(k-1)}]^{\mathsf{T}} (P_k)^{-1} P_j^{(k-1)},$$

A segunda relação se obtém de modo análogo. □

Usando essas relações podemos propor o Algoritmo 26.

Algoritmo 26: SBAINV-Variety

1 $W_i^{(0)} \leftarrow E_i, i = 1, ..., n$;
2 **for** $i \leftarrow 1$ **to** n **do**
3 $\quad V_i \leftarrow AW_i^{(i-1)}$;
4 $\quad Q_i^{(i-1)} \leftarrow ((W_i^{(i-1)}))^T V_i$;
5 $\quad P_i^{(i-1)} \leftarrow Q_i^{(i-1)}$;
6 $\quad$ **if** $i = n$ **then**
7 $\quad\quad$ go to 13
8 $\quad$ **for** $j \leftarrow i + 1$ **to** n **do**
9 $\quad\quad Q_j^{(i-1)} \leftarrow (W_j^{(i-1)})^T V_i$;
10 $\quad\quad P_j^{(i-1)} \leftarrow A_{ij} - \sum_{k=1}^{i-1} (Q_i^{(i-1)})^T U_j$;
11 $\quad\quad U_j \leftarrow (P_i^{(i-1)})^{-1} P_j^{(i-1)}$;
12 $\quad\quad W_j^{(i)} \leftarrow W_j^{(i-1)} - W_i^{(i-1)} (Q_i^{(i-1)})^{-1} Q_j^{(i-1)}$;
13 $W_i \leftarrow W_i^{(i-1)}$ and $Q_i \leftarrow Q_i^{(i-1)}$. Return $W \leftarrow [W_1, ..., W_n], U \leftarrow U_{ij}$ and $D \leftarrow diag(P_1, ..., P_n)$.

Naturalmente, essa versão é equivalente ao SBAINV e, portanto, é livre de quebras. Nesse caso, nós calcularemos as matrizes U e W e, para construir a fatoração: $A^{-1} = U^{-1}D^{-1}W$ faremos uso da série de Newmann de quarta ordem a fim de estimar U^{-1}.

Podemos também generalizar o FFAPINV para bloco (ver Algoritmo 27): queremos encontrar matrizes W e Z tais que $WAZ = D$, com D matriz diagonal. Para tal, as linhas da matriz W e as colunas Z são construídas recursivamente, a partir da identidade e das colunas previamente construídas, isto é, $z_1 = e_1$ e:

$$Z_j = E_j + \sum_{i=1}^{j-1} Z_i \alpha_i, j = 2, \ldots, n.$$

Analogamente,

$$W_j = E_j^T + \sum_{i=1}^{j-1} W_i \beta_i, j = 2, \ldots, n.$$

Assim, primeiramente, observe que $D_1 = W_i A Z_i = A_{11}$. Agora, para $k = 1, \ldots, j-1$ temos

$$\begin{aligned} 0 &= W_k A Z_j \\ &= W_k A E_j + \sum_{i=1}^{j-1} W_k A Z_i \alpha_i \\ &= W_k A_{*j} + W_k A Z z_k \alpha_k \\ &= w_k A_{*j} + D_k \alpha_k \end{aligned}$$

Assim,

$$\alpha_k = -{D_k}^{-1} W_k A_{*j}, k = 1, \ldots, j-1.$$

Do mesmo modo, obtemos que:

$$\beta_k = -{D_k}^{-1} A_{j*} Z_k, k = 1, \ldots, j-1.$$

Algoritmo 27: BFFAPINV

```
1  Z_i^(0) ← E_i; W_1 = E_1^T, i = 1, ..., n;
2  for j ← 2 to n do
3      Z_j ← E_j, W_j ← E_j^T;
4      for i ← 1 to j − 1 do
5          α_k ← −d_k^{-1} w_k A_{*j};
6          β_k ← −d_k^{-1} A_{j*} z_k;
7          Z_j ← Z_j + Z_i α_i;
8          W_j ← W_j + W_i β_i;
9          Drop entries of Zj and Wj whose absolute values are smaller than τ
10     D_j ← W_j A Z_j
```

Novamente, afim de construir o FFAPINV para matrizes não simétricas e positivo definidas (NSPD), precisamos primeiramente observar que $D_j = Z_j^T A Z_j = W_j A W_j^T$ para $j = 1, \ldots, n$. De fato, como $WAZ = D$ segue que $Z^T AZ = Z^T W^{-1} D$. Assim, como $Z^T W^{-1} D$ é triangular inferior e $diag(Z^T W^{-1} D) = D$, podemos concluir que $diag(Z^T AZ) = D$ e, portanto, $D_j = Z^T AZ$. A segunda igualdade demonstra-se de modo análogo. Dessa discussão temos que D_j é positivo definida, ver Algoritmo 28.

Todas essas variações entre outras possuem uma possibilidade de investigação

Algoritmo 28: BFFAPINV-NSPD

```
1  Z_i^(0) ← E_i; W_1 ← E_1^T, i = 1, ..., n;
2  for j ← 2 to n do
3      Z_j ← E_j, W_j ← E_j^T for i ← 1 to j − 1 do
4          α_k ← −D_k^{-1} W_k A_{*j};
5          β_k ← −D_k^{-1} A_{j*} Z_k;
6          if |α_k| > τ then
7              Z_j ← E_j + Z_i α_i;
8          if |β_k| > τ then
9              W_j ← E_j^T + W_i β_i;
10         Drop entries of Zj and Wj whose absolute values are smaller than τ
11     D_j ← A_{j*} Z_k if det(D_j) = 0 then
12         D_j ← W_j A Z_j
```

quanto aos experimentos numéricos.

Outra possibilidade é investigar qual é o efeito dos diferentes tipos de descartes, reordenamentos e reescalonamentos para cada uma dessas variações. Abaixo listaremos algumas possibilidades:

Reescalonamentos:

- Máximo;
- Jacobi;
- Bloco Jacobi;
- DCR.

Reordenamentos:

- AMD;
- MIP.

Descartes:

- Por tolerância;
- Por tolerância de bloco;
- Por preenchimento pré-definido;

- Por bloco com preenchimento pré-definido;
- AINV(0) - só preenche aonde A tiver preenchimento;
- BAINV(0) - só preenche nas posições de bloco que são não nulos em A;
- Por tolerância de z_{li} vezes a linha i de A;
- Usando os fatores L e U.

CONCLUSÃO

Nesta tese, estudamos a estrutura matemática, as principais propriedades e o histórico contendo contribuições ao algoritmo de Inversa Aproximada proposto em [11], especialmente aquelas que possuem uma estrutura de blocos.

Primeiramente, nós discutimos as bases matemáticas de alguns métodos iterativos, em especial, o método dos Gradientes Conjugados (CG) e o método do Resíduo Mínimo Generalizado (GMRES). Nessa discussão apresentamos os algoritmos e algumas de suas propriedades.

Exibimos também diversas propostas de evolução do AINV e apresentamos as principais propriedades dessas versões. Generalizamos, matematicamente, tais algoritmos para as suas versões em bloco. Observamos que as várias formas de se conceber o AINV diferiam principalmente quanto: como o descarte é realizado, como os multiplicadores são calculados ou como a fatoração da inversa de A é feita. Observamos também que foram raros os trabalhos feitos com versões em bloco e, nesses raros trabalhos, nenhuma tratamento sobre estabilidade ou complexidade desses algoritmos foram exibidos.

Com relação à versão bloco do AINV, estabelecemos suas condições matemáticas de funcionamento. Nesse sentido, propusemos algumas definições e generalizações de conceitos básicos de Álgebra Linear. Exibimos também uma demonstração da validade do BAINV e apresentamos uma interpretação geométrica e algébrica das condições necessárias para validação do BAINV.

Adicionalmente, conseguimos verificar que as matrizes $\mathcal{M}$ e as matrizes $\mathcal{H}$ podem ser utilizadas no BAINV, mesmo com descarte, com garantias de ser livre de quebra. Durante esse processo, nós pudemos observar e demonstrar algumas propriedades dessas duas famílias de matrizes.

Por fim, propusemos versões em bloco de algumas variações do AINV e apresentamos suas demonstrações matemáticas de validação e propusemos um catálogo de trabalhos futuros.

REFERÊNCIAS

[1] AJIZ, M. A. e JENNINGS, A. A Robust Incomplete Cholesky Conjugate Gradient Algorithm. Int. J. Numerical Methods Engrg. 20. pp 949–966. 1984.

[2] ABDELFATTAH A., ANZT H., DONGARRA J., GATES M., HAIDAR A., KURZAK J., LUSZCZEK P., TOMOV S., YAMAZAKI I. e YARKHAN A. Linear algebra software for large-scale accelerated multicore computing. *Acta Numerica*, 25, pp. 1-160, 2016.

[3] AMESTOY P. R., DAVIS T. A. e DUFF I. S. An Approximate Minimum Degree Ordering Algorithm. SIAM J. Matrix Analysis & Applic., Vol 17, no 4, pp. 886-905, 1996.

[4] https://www.mathworks.com/help/matlab/ref/amd.html, acesso em 15/01/2018.

[5] ARNOLDI, W., "The principle of minimized iteration in the solution of the matrix eigenvalue problem", Quarterly of Applied Mathematics, v. 9, pp. 17– 25, 1951.

[6] AXELSSON O. Iterative Solution Methods, Cambridge University Press, Cambridge, 1994.

[7] AXELSSON O. e KOLOTINA L. Yu. Diagonally Compensated Reduction and Related Preconditioning Methods, Numer. Linear Algebra Appl., pp. 155-177, vol 1, 1994.

[8] BARNARDY S. T. e GROTEZ M. J. A block version of the spai preconditioner. In *Proceedings of the 9th SIAM conference on Parallel Processing for Scientific Computing,*, San Antonio, TX, 1999.

[9] BENZI M. Preconditioning techniques for large linear systems: a survey. *Journal of Computational Physics*, 182(2):418–477, November 2002.

[10] BENZI M., KOUHIA R. e TŮMA M. Stabilized and block approximate inverse preconditioners for problems in solid and structural mechanics. *Computer Methods in Applied Mechanics and Engineering*, 190(49-50):6533 – 6554, 2001.

[11] BENZI M., MEYER C. D, e TŮMA M. A sparse approximate inverse preconditioner for the conjugate gradient method. *SIAM Journal on Scientific Computing*, 17(5):1135–1149, 1996.

[12] BENZI M.e TŮMA M. A sparse approximate inverse preconditioner for nonsymmetric linear systems. *SIAM Journal on Scientific Computing*, 19:968–994, 1998.

[13] BENZI M.e TŮMA M. A comparative study of sparse approximate inverse preconditioners. *Applied Numerical Mathematics*, 30:305–340, 1999.

[14] BENZI M., CULLUM J. e TŮMA M. Robust Approximate Inverse Preconditioning for the Conjugate Gradient Method. SIAM J. Sci. Comput., 22(4), 1318—1332. 2000.

[15] BENZI M.e TŮMA M.. A Robust Preconditioner with Low Memory Requirements for Large Sparse Least Squares Problems. Siam J. S. C. Vol. 25, No. 2, pp. 499–512. 2003.

[16] BOOTH J. D., KIM K. e RAJAMANICKAM S. A comparison of high-level programming choices for incomplete sparse factorization across different architectures. In *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp 397-406, 2016.

[17] BOLLHÖFER M. e SAAD Y. ILUs and Factorized Approximate Inverses are Strongly Related. Part I: Overview of Results. 2000.

[18] BOLLHÖFER M. A robust ILU based on monitoring the growth of the inverse factors. Linear Algebra Appl. 338(1–3), pp 201–218, 2001.

[19] BOLLHÖFER M. e SAAD Y. On the relations between ILUs and factored approximate inverses. SIAM J. Matrix Anal. Appl. 24(1), pp 219–237, 2002.

[20] BOLLHÖFER M. e SAAD Y. A factored approximate inverse preconditioner with pivoting, SIAM J.Matrix Anal. Appl. 23, pp 692–705. 2002.

[21] BOLLHÖFER, M.: A robust and efficient ILU that incorporates the growth of the inverse triangular factors. Siam J. S. C. 25(1), pp 86–103, 2003.

[22] BRIDSON R. e TANG W. Refining an Approximate Inverse. Appeared in J. of Comp. and Appl. Math, vol. 123 (Numerical Analysis 2000 vol. III: Linear Algebra), pp. 293–306, 2000.

[23] BRU R., CERDÁN J., MARÍN J. e MAS J. Preconditioning sparse nonsymmetric linear systems with the Sherman–Morrison formula. *SIAM Journal on Scientific Computing*, 25(2):701–715, 2003.

[24] BRU R., CORRAL C., GIMENEZ I. e MAS J. Classes of general H-matrices . *Linear Algebra and its Applications 429, pp 2358–2366, 2008.*

[25] BRU R., MARÍN J., MAS J. e TŮMA M. Balanced incomplete factorization. *SIAM Journal on Scientific Computing*, 30(5):2302–2318, 2008.

[26] BRU R., MARÍN J., MAS J. e TŮMA M. Improved balanced incomplete factorization. *SIAM Journal on Matrix Analysis and Applications*, 31(5):2431–2452, 2010.

[27] BUTTARI A., LANGOU J., KURZAK J. e DONGARRA J. A class of parallel tiled linear algebra algorithms for multicore architectures. *Parallel Computing*, 35(1):38 – 53, 2009.

[28] CERDÁN J., FARAJ T., MALLA N., MARÍN J. e MAS J. Block approximate inverse preconditioners for sparse nonsymmetric linear systems. *Electronic Transactions on Numerical Analysis*, 37, pp 23-40, 2010.

[29] . CENI M., CARVALHO L. M. e SOUZA M. Um método para o Calculo da Inversa de Matrizes Simetricas e Positivas Definidas em Bloco. Proceeding Series of the Brazilian Society of Computational and Applied Mathematics. v. 5, n.1, 2017.

[30] CHAN, T. e van der VORST, H. *Parallel Numerical Algorithms, ICASE/LaRC Interdisciplinary Series in Science and Engineering*, chapter Approximate and incomplete factorizations, Kluwer, Dordrecht, pp 167-202. 1997.

[31] CHOW, E.; ANZT, H. e DONGARRA, J. *Asynchronous Iterative Algorithm for Computing Incomplete Factorizations on GPUs*, Springer International Publishing, pp 1-16, 2015.

[32] CHOW, E. e PATEL, A. Fine-grained parallel incomplete lu factorization. *SIAM Journal on Scientific Computing*, 37(2), pp 169-193, 2015.

[33] CHOW, E. E SAAD, Y. Approximate inverse preconditioners via sparse-sparse iterations. *SIAM Journal on Scientific Computing*, 19(3), pp 995–1023, 1998.

[34] DONGARRA J. e SULLIVAN F. Guest editors' introduction to the top 10 algorithms. Computing in Science and Engineering, 2(1), pp 22–23, 2000.

[35] DONGARRA, J., DUFF, I.S., SORENSEN, D. C. e van der VORST H. A. *Numerical Linear Algebra for High-Performance Computers*. SIAM, Philadelphia, PA, USA, 1998.

[36] EISENSTAT, S. C., ELMAN, H. C., SCHULTZ, M. H., "Variational Iterative Methods for Nonsymmetric Systems of Linear Equations", SIAM Journal on Numerical Analysis, v. 20, n. 2, pp. 345–357, 1983.

[37] FERRONATO M., JANNA C. e PINI G. A generalized block FSAI preconditioner for nonsymmetric linear systems. *Journal of Computational and Applied Mathematics*, 256(0), pp.230 - 241, 2014.

[38] FOX, L. An Introduction to Numerical Linear Algebra, Oxford University Press, Oxford, UK, 1964.

[39] FUJINO, S. e IKEDA, Y. An Improvement of Sainv and Rif Precondtionings of CG Method by Double Dropping Strategy. IPSJ, 45, pp. 10-17. 2004.

[40] GOLUB, G. H. van LOAN, C. F. *Matrix computations*, Johns Hopkins University Press, 1996.

[41] GAMBOLATI, G., PINI, G. e FERRONATO, M. Scaling improves stability of preconditioned CG-like solvers for FE consolidation equations. International Journal for Numerical and Analytical Methods in Geomechanics. 27(12), pp. 1043-1056. 2003.

[42] GROTE, M. J. e HUCKLE, T. Parallel preconditioning with sparse approximate inverses. *SIAM Journal on Scientific Computing*, 18(3), pp. 838-853, 1997.

[43] GILBERT, G. T. Positive Definite Matrices and Sylvester's Criterion. The American Mathematical Monthly Vol. 98, No. 1, pp. 44-46, 1991.

[44] HESTENES M. e STIEFEL E. Methods of conjugate gradients for solving linear systems. *J. R. N. B. S.*, 49, pp. 409–436, 1952.

[45] HORN, R. A.; JOHNSON, C.R. *Matrix Analysis*, Segunda Edição, 2013, Cambridge University Press.

[46] IPSEN, I. C. F. *Numerical Matrix Analysis, Linear Systems an Least Squares*, Soc. Ind. Appl. Math., Philadelphia, 2009.

[47] IPSEN, I. C. F. e MEYER, C. The Idea Behind Krylov Methods. The American Mathematical Monthly. Vol 105, No 10. pp. 889-899, 1998

[48] JANNA, C., FERRONATO, M. e GAMBOLATI, G. A block FSAI-ILU parallel preconditioner for symmetric positive definite linear systems. *SIAM Journal on Scientific Computing*, 32(5), pp. 2468–2484, 2010.

[49] C. JANNA, M. FERRONATO, e G. GAMBOLATI. The use of supernodes in factored sparse approximate inverse preconditioning. *SIAM Journal on Scientific Computing*, 37(1):C72–C94, 2015.

[50] C. JANNA, M. FERRONATO, e G. GAMBOLATI. Enhanced block fsai preconditioning using domain decomposition techniques. *SIAM Journal on Scientific Computing*, 35(5):S229–S249, 2013.

[51] KHARCHENKO, S. A., KOLOTILINA L. Yu., NIKISHIN A. A. e YEREMIN A. Yu., A robust AINVtype method for constructing sparse approximate inverse preconditioners in factored form, Numer. Linear Algebra With Appl., 8, pp. 165179, 2001.

[52] KUMARESAN, S., *Sylvester Criterion for Positive Definiteness*, Dept. of Math. and Stat., University of Hyderabad, 2004.

[53] KOLOTILINA, L.Y. e YEREMIN, A.Y. Factorized sparse approximate inverse preconditionings I. theory. *SIAM Journal on Matrix Analysis and Applications*, 14(1), pp. 45-58,1993.

[54] KOLOTILINA L.Y. Two-Sided Bounds for the Inverse of an H-Matrix. *LINEAR ALGEBRA AND ITS APPLICATIONS 225, pp. 114-123, 1995.*

[55] LAGO, R. F. Estudos Sobre os Métodos Iterativos de Krylov para Solução de Sistemas de Equações Lineares/Dissertação de mestrado em Engenharia de Sistemas e Computação. Rio de Janeiro: UFRJ/COPPE, 2010.

[56] LEE, E.J. e ZHANG, J. Fatored approximate inverse preonditioners with dynamic sparsity patterns, Tehnial Report No.488-07,Department of Computer Science,University of Ken- tuky, Lexington, KY, 2007.

[57] LIMA, E.L. *Álgebra Linear*. IMPA, 7ª edição, 2006.

[58] LIU, W., LI A., HOGG, J., DUFF, I. S. e VINTER B. *A Synchronization-Free Algorithm for Parallel Sparse Triangular Solves*, pp. 617-630. Springer International Publishing, Cham, 2016.

[59] MEENAKSHI B.R. e RAJIAN C. *On a product of positive semidefinite matrices*, Linear Algebra and its Applications 295, pp. 3-6, 1999.

[60] MEIJERINK, J. A. e van der VORST, H. A. An iterative solution method for linear systems of which the coefficient matrix is a symmetric m-matrix. *Math. Comp.*, 31, pp. 148-162, 1977.

[61] MANTEUFFEL, T.A. An incomplete factorization technique for positive definite linear system. *Mathematics of Computation*, 34, pp. 473–497, 1980.

[62] MORITA, N., HASHIMOTO, G. e OKUDA, H. Stabilized approximate inverse preconditioning based on A-orthogonalization for parallel finite element analysis. Japan Society for Industrial and Applied Mathematics. JSIAM lettes 8, pp 25-28. 2016.

[63] MOTTA, V. S. Solução de Sistemas Lineares de Grande Porte com Múltiplos Lados Direitos. Tese de Doutorado em Engenharia de Sistemas e Computação. Rio de Janeiro: UFRJ/COPPE, 2010.

[64] OSTROWSKI, A., Uber die Determinanten mit uberwiegender Hauptdiagonal, Comment. Math. Hek. 10, pp. 69-96, 1937.

[65] PLEMMONS, R. J. M-matrix characterizations. I—nonsingular M-matrices. *Linear Algebra and Its Applications*, 18, pp. 175-188, 1977.

[66] RAFIEI, A. e TOUTOUNIAN, F. New breakdown-free variant of AINV method for nonsymmetric positive definite matrices. Journal of Computational and Applied Mathematics 219. pp. 72-80. 2008.

[67] RAFIEI, A. e BOLLHÖFER, M. Robust incomplete factorization for nonsymmetric matrices. Numerische Mathematik, 118. pp. 247–269. 2011.

[68] RAFIEI, A. Left-looking version of AINV preconditioner with complete pivoting strategy. Linear Algebra and its Applications 445, pp. 103–126. 2014.

[69] SAAD, Y. *Iterative Methods for Sparse Linear Systems*. SIAM, 2nd edition, 2003.

[70] SAAD, Y. e van der VORST, H. A. Iterative solution of linear systems in the 20th century. *Journal of Computational and Applied Mathematics*, 123(1–2), Numerical Analysis. Vol. III: Linear Algebra, pp. 1-33, 2000.

[71] SAAD, Y. e SCHULTZ, M. H. "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems", SIAM Journal on Scientific and Statistical Computing, v. 7, n. 3, pp. 856–869, 1986.

[72] SAAD, Y. e ZHANG, J. BILUM: Block versions of multielimination and multilevel ILU preconditioner for general sparse linear systems. *SIAM Journal on Scientific Computing*, 20(6), pp. 2103-2121, 1999.

[73] SAAD, Y. e ZHANG, J. BILUTM: A domain-based multi-level block ILUT preconditioner for general sparse matrices. *SIAM Journal on Matrix Analysis and Applications*, 21, pp. 279–299, 2000.

[74] SAAD, Y. "Krylov Subspace Methods for Solving Large Unsymmetric Linear Systems", Mathematical Computing, v. 37, n. 155, pp. 105–126, 1981.

[75] SALKUYEH, D. K. A Sparse Approximate Inverse Preconditioner for Nonsymmetric Positive Definite Matrices. J. Appl. Math. & Informatics Vol. 28, No. 5-6, pp. 1131-1141, 2010.

[76] SALKUYEH, D. K. e ROOHANI, H.. On the Relation between the AINV and the FAPINV Algorithms. Hindawi Publishing Corporation. International Journal of Mathematics and Mathematical Sciences. 2009.g

[77] SEDLACEK, M. *Sparse Approximate Inverses for Preconditioning, Smoothing, and Regularization*. PhD thesis, Universität München, 2012.

[78] ZHAN, S. *On the determinantal inequalities*, Journal of Inequalities in Pure and Applied Mathematics, Volume 6, Article 105, 2005.

[79] TROTTEMBERG, U., OOSTERLEE, C. e SCHULLER, A. *Multigrid*. Academic Press, 2001.

[80] YEREMIN, A., KOLOTINILINA, L. e NIKISHIN, A. Factorized sparse approximate inverse preconditionings. III. iterative construction of preconditioners. *Journal of Mathematical Sciences*, 101(4), pp. 3237-3254, 2000.

[81] VARGA, R. S. Matrix Zteratiue Analysis, Prentice-Hall, Englewood Cliffs, N. J., 1962.

[82] ZHANG, J. A sparse approximate inverse technique for parallel preconditioning of general sparse matrices, Appl. Math. Comput., 130, pp. 63-85, 2002.