

Everton Boos

**MÉTODO ITERATIVO BASEADO NA ITERAÇÃO
DE NEWTON PARA PROBLEMAS DISCRETOS
MAL POSTOS**

Florianópolis
Fevereiro/2018

Everton Boos

**MÉTODO ITERATIVO BASEADO NA ITERAÇÃO
DE NEWTON PARA PROBLEMAS DISCRETOS
MAL POSTOS**

Dissertação submetida ao Programa
de Pós-Graduação em Matemática
Pura e Aplicada para a obtenção
do Grau de mestre em Matemática
Pura e Aplicada.

Universidade Federal de Santa Catarina – UFSC
Centro de Ciências Físicas e Matemáticas
Departamento de Matemática

Orientador: Prof. Dr. Fermín S. V. Bazán

Florianópolis
Fevereiro/2018

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Boos, Everton

Método iterativo baseado na iteração de Newton
para problemas discretos mal postos / Everton Boos
; orientador, Fermín Sinforiano Viloche Bazán, 2018.
116 p.

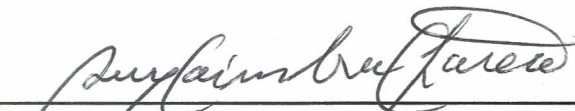
Dissertação (mestrado) - Universidade Federal de
Santa Catarina, Centro de Ciências Físicas e
Matemáticas, Programa de Pós-Graduação em Matemática
Pura e Aplicada, Florianópolis, 2018.

Inclui referências.

1. Matemática Pura e Aplicada. 2. Problemas
discretos mal postos. 3. Métodos iterativos. 4.
Semi-convergência. 5. Critérios de parada. I.
Bazán, Fermín Sinforiano Viloche. II. Universidade
Federal de Santa Catarina. Programa de Pós-Graduação
em Matemática Pura e Aplicada. III. Título.

Esta Dissertação foi julgada aprovada para a obtenção do Título de “mestre em Matemática Pura e Aplicada”, e aprovada em sua forma final pelo Programa de Pós-Graduação em Matemática Pura e Aplicada.

Florianópolis, 21 de fevereiro de 2018.



Prof. Dr. Ruy Coimbra Charão
Coordenador - UFSC


Banca Examinadora:



Prof. Dr. Fermín S. V. Bazán
Orientador - UFSC



Prof. Dr. Hugo José Lara Urdaneta
UFSC - Blumenau



Prof. Dr. Julião de Bem Francisco
UFSC



Prof. Dr. Luciano Bedin
UFSC

Aos meus pais, pelo apoio e amizade incondicionais.

AGRADECIMENTOS

Aos meus pais, Laercio e Ana Carla, por estarem presentes a todo momento, apoiando e ajudando no que fosse preciso. Seu modelo de força de vontade sempre me mostrou que, por mais pedregoso que o caminho pareça, as adversidades não de cair perante o esforço (ou seria teimosia?).

Ao professor Fermín, por esses anos de orientação e conversas em meio à transmissão de conhecimentos. Além de sua contribuição científica, me ensinou que muito acima do talento ou habilidades pessoais inatas, resultados são gerados por perseverança e trabalho duro, algo que levarei por toda a vida.

Aos meus mestres em geral, interessados na passagem de seus ensinamentos às novas gerações de pesquisadores. Em especial, aos professores participantes da banca, pelas contribuições valiosas à apresentação final deste trabalho.

À todos os amigos e colegas de jornada trazidos pela Matemática que se fizeram essenciais durante estes últimos anos, seja elucidando os conhecimentos ou apresentando palavras encorajadoras. Todos os cafés, as conversas, as discussões acaloradas sobre as disciplinas e os momentos de descontração foram de suma importância e culminaram a este momento.

À todos aqueles que, de um jeito ou de outro, foram a calma entre as tempestades, trazendo luz aos dias de trevas e fazendo valer a pena cada momento.

Ao CNPq, pelo apoio financeiro durante todo o mestrado.

RESUMO

Propomos um método iterativo para a resolução de problemas discretos mal postos baseado em iterações matriciais geradas através do método de Newton, conhecido por convergir para a pseudo-inversa de uma matriz. Essencialmente, tomando os iterados gerados pelo método de Newton matricial por X_k , construímos os vetores $x^{(k)} = X_k b$, em que b é o vetor com dados do problema. Por construção, estes iterados convergem para a solução de norma mínima do problema de mínimos quadrados $\min_{x \in \mathbb{R}^n} \|Ax - b\|_2$. Mostramos que a sequência $\{x^{(k)}\}$ é quadraticamente convergente e ilustramos o seu comportamento numérico. No caso de dados de entrada com ruído, analisamos o comportamento de semi-convergência nos iterados e concluímos que as iterações devem ser truncadas para controlar a propagação do ruído. Como resultado, encontramos uma estimativa de erro para o caso em que o princípio da discrepância é utilizado para determinar o parâmetro de truncamento. Na tentativa de contornar o alto esforço computacional, estudamos variantes do método que incluem estratégias de projeção, resultando em alternativas atrativas para aplicação em problemas de grande porte. Diversos resultados numéricos são apresentados para ilustrar a efetividade do método em problemas-teste bem conhecidos da literatura.

Palavras-chave: Problemas discretos mal postos. Métodos iterativos. Semi-convergência. Critérios de parada.

ABSTRACT

We propose an iterative method to solve discrete ill-posed problems based on matrix iterations generated by Newton's method, known to converge to the pseudoinverse of a matrix. Essentially, letting the Newton matrix iterate be X_k , we construct the vectors $x^{(k)} = X_k b$, where b is the data vector of the problem. By construction, these iterates converge to the minimum norm solution of the least squares problem $\min_{x \in \mathbb{R}^n} \|Ax - b\|_2$. We show that the sequence $\{x^{(k)}\}$ is quadratically convergent and illustrate its numerical behavior. In the case of noisy data, we analyze the semi-convergence behavior of the iterates and conclude that the iterations must be truncated to control the propagation of the noise error. As a result, we derive an error estimate for the case when the discrepancy principle is used as stopping parameter rule. In order to overcome the high computational effort, we study variants of the method that include projection strategies, resulting on attractive alternatives that are well suited for large-scale problems. Several numerical results are presented to illustrate the effectiveness of the method on well known test problems from literature.

Keywords: Discrete ill-posed problems. Iterative methods. Semi-convergence. Stopping rules.

LISTA DE ABREVIATURAS E SIGLAS

SVD	Decomposição em valores singulares (<i>singular value decomposition</i>)
TSVD	<i>Truncated SVD</i>
LSQR	Método <i>Least Squares QR</i>
GKB	Bidiagonalização de Golub-Kahan
DP	Princípio da discrepância (<i>discrepancy principle</i>)
MPR	Regra do produto mínimo (<i>minimum product rule</i>)
NS	Método de Newton escalado (<i>Newton scaled</i>)
NC	Método de Newton cúbico
NS- ε	Método variante de NS, que suprime valores singulares menores que ε
DCT	Transformada discreta de cosseno (<i>discrete cosine transform</i>)
NL	Nível de ruído (<i>noise level</i>)
OP	Parâmetro ótimo (<i>optimal parameter</i>)
N-Bid	Método de Newton com projeção no subespaço de Krylov gerado pelo processo de bidiagonalização de Lanczos
N-Trid	Método de Newton com projeção no subespaço de Krylov gerado pelo algoritmo de Arnoldi/Lanczos
N-DCT	Método de Newton com projeção no subespaço gerado pela base DCT

LISTA DE SÍMBOLOS

\mathbb{R}	Corpo dos números reais
$\mathbb{R}^{m \times n}$	Espaço das matrizes de m linhas e n colunas com coeficientes em \mathbb{R}
\mathbb{R}^n	Espaço de vetores (coluna) com coeficientes em \mathbb{R} . De fato, $\mathbb{R}^n \cong \mathbb{R}^{n \times 1}$
$\langle x, y \rangle$	Produto interno canônico em \mathbb{R}^n , isto é, $\langle x, y \rangle = \sum_{i=1}^n x_i y_i \equiv x^T y$, para $x, y \in \mathbb{R}^n$
$\ x\ _2$	Norma vetorial canônica em \mathbb{R}^n dada por $\ x\ _2 = \sqrt{\langle x, x \rangle} \equiv \sqrt{x^T x} = \sqrt{x_1^2 + \dots + x_n^2}$, para $x \in \mathbb{R}^n$
I, I_n	Matriz identidade de ordem n
$\mathbf{0}$	Matriz ou vetor composto de zeros somente
e_k	k -ésimo vetor da base canônica, isto é, e_k é vetor composto por zeros, em que a k -ésima entrada é igual a 1
$\mathcal{N}(A)$	Núcleo do operador A
$\mathcal{R}(A)$	Imagem do operador A
A^T	Matriz transposta de A , isto é, se $A = (a_{ij}) \in \mathbb{R}^{m \times n}$, então $A^T = (b_{ij}) \in \mathbb{R}^{n \times m}$ é tal que $b_{ij} := a_{ji}$
A^\dagger	Pseudo-inversa de Moore-Penrose de A
$\sigma_i, \sigma_i(A)$	i -ésimo valor singular de A
$\ A\ _2$	2-norma matricial (também chamada de norma espectral), dada por $\ A\ _2 = \sup_{x \neq \mathbf{0}} \frac{\ Ax\ _2}{\ x\ _2} \equiv \sqrt{\lambda_{\max}(A^T A)}$, em que $\lambda_{\max}(A^T A)$ corresponde ao maior autovalor de $A^T A$
$\ A\ _F$	Norma de Frobenius de $A \in \mathbb{R}^{m \times n}$, dada por $\ A\ _F^2 = \sum_{i=1}^m \sum_{j=1}^n a_{ij} ^2$

$\rho(A)$	Raio espectral da matriz quadrada A , dado pelo máximo do valor absoluto dos autovalores de A
$\kappa(A)$	Número de condição de A , i.e., $\kappa(A) = \ A\ _2 \ A^\dagger\ _2$
k_m	Iteração mínima de parada
k_M	Iteração máxima de parada
E_m	Erro relativo médio
t_m	Tempo médio de execução
ℓ_m	Menor tamanho de subespaço determinado
ℓ_M	Maior tamanho de subespaço determinado

SUMÁRIO

1	INTRODUÇÃO	19
2	PRELIMINARES	23
2.1	DO CONTÍNUO AO DISCRETO	26
2.2	EFEITO DO RUÍDO NO PROBLEMA DISCRETO	29
2.3	TSVD E MÉTODOS DE PROJEÇÃO	31
2.3.1	Método LSQR	34
2.4	CRITÉRIOS DE PARADA PARA MÉTODOS ITERATIVOS	38
2.4.1	Princípio da discrepância (DP)	38
2.4.2	L-curve	39
2.4.3	Regra do produto mínimo (MPR)	41
2.5	ASPECTOS DA REGULARIZAÇÃO DE TIKHONOV	42
3	O MÉTODO DE NEWTON	47
3.1	MÉTODO ITERATIVO PARA A PSEUDO-INVERSA	48
3.1.1	Cálculo de β	50
3.2	MÉTODO DE NEWTON MATRICIAL	53
3.3	O MÉTODO ITERATIVO VETORIAL	59
3.4	ASPECTOS COMPUTACIONAIS	65
4	ITERAÇÕES DE NEWTON REGULARIZADAS	69
4.1	ESTIMATIVA TEÓRICA	71
4.2	RESULTADOS NUMÉRICOS	74
5	ESTRATÉGIAS DE ACELERAÇÃO	81
5.1	VARIAÇÕES AO MÉTODO DE NEWTON	81
5.1.1	Aceleração via <i>scaling</i>	82
5.1.2	Aceleração por polinômio cúbico	86
5.1.3	Suprimindo pequenos valores singulares	90
5.1.4	NCS: método misto	94
5.1.5	Aplicação à problemas com perturbações	95
5.2	ACELERAÇÃO POR PROJEÇÃO	98
5.2.1	Determinação da dimensão do subespaço	100
5.2.2	Subespaço dado a priori	103
5.2.3	Resultados numéricos	105
6	CONCLUSÃO	111
	REFERÊNCIAS	113

1 INTRODUÇÃO

Diversas aplicações na ciência e na engenharia conduzem a sistemas de equações lineares de grande porte da forma $Ax = \tilde{b}$ ou, mais geralmente, ao problema de mínimos quadrados associado

$$\min_{x \in \mathbb{R}^n} \|Ax - \tilde{b}\|_2, \quad (1.1)$$

para $A \in \mathbb{R}^{m \times n}$ severamente mal condicionada e $\tilde{b} = b + e$, em que $b \in \mathbb{R}^m$ denota os dados exatos e o vetor e corresponde ao ruído proveniente de imprecisões ou erros de medição. Problemas dessa forma são chamados de problemas discretos mal postos [22] e surgem naturalmente, por exemplo, na discretização de equações integrais de primeira espécie em processamento de sinais e restauração de imagens ou quando estamos interessados em conhecer a estrutura interna de um sistema através de medições externas, como acontece na tomografia computadorizada [20, 26, 41].

É fato conhecido que a solução para (1.1) gerada através da pseudo-inversa de Moore-Penrose A^\dagger [28, 36], denotada por $\tilde{x}_{LS} = A^\dagger \tilde{b}$, é dominada pelo ruído nos dados de entrada. Neste cenário, métodos de regularização são essenciais para computar aproximações estáveis da solução livre de ruídos $x_{LS} = A^\dagger b$ procurada. A literatura possui diversas abordagens com este intuito, como a regularização de Tikhonov [45], que consiste basicamente em trocar (1.1) por um problema relacionado bem condicionado cuja solução é aceita como aproximação para x_{LS} .

Outra maneira de lidar com o problema é através de métodos iterativos, tais como CGLS/LSQR [10, 33], MINRES/MR-II [17, 32] e GMRES/RRGMRES [12, 40], entre muitos outros, para os quais o índice k da iteração atua como parâmetro de regularização. Isto ocorre porque, para muitos métodos iterativos, nos passos iniciais, os iterados tendem a convergir para x_{LS} , para então se deteriorarem com o crescimento de k , até convergirem finalmente para a solução de (1.1), dominada pelo ruído. Este fenômeno, conhecido como semi-convergência [31], sugere que a interrupção das iterações após um certo número de passos tem efeito regularizante. Para tanto, critérios de parada específicos são essenciais para determinar boas soluções.

Neste trabalho, propomos um método iterativo para computar aproximações estáveis para a solução livre de ruídos do problema (1.1), $x_{LS} = A^\dagger b$, baseado no fato de que se $\{X_k\}$ é uma sequência matricial

que converge para A^\dagger , então a sequência vetorial $\{x^{(k)}\}$, definida por

$$x^{(k)} = X_k b, \quad (1.2)$$

converge para a solução de mínimos quadrados de norma mínima $x_{LS} = A^\dagger b$. Entre uma vasta gama de possibilidades presentes na literatura, a geração da sequência matricial $\{X_k\}$ se dá através da recorrência

$$X_{k+1} = X_k(2I - AX_k), \quad (1.3)$$

que consiste em um caso particular de um método de alta ordem [9] e possui um paralelo com as iterações geradas pelo método de Newton quando aplicado a um certo problema não linear. De fato, ao buscarmos a raiz da função $f(x) = a - x^{-1}$ com o objetivo de encontrar o recíproco de $a \neq 0$, o método de Newton produz os iterados

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} = x_k(2 - ax_k),$$

que convergem quadraticamente para a^{-1} , desde que o iterando inicial x_0 seja escolhido apropriadamente. Esta ideia pode ser estendida [34, 43] para a função matricial $f(X) = M - X^{-1}$, em que M é não singular, gerando a recorrência $X_{k+1} = X_k(2I - MX_k)$, de modo que $X_k \rightarrow M^{-1}$. O caso mais geral é representado pelas iterações (1.3), para $A \in \mathbb{R}^{m \times n}$ qualquer, que convergem para a pseudo-inversa A^\dagger . A convergência quadrática característica do método de Newton é preservada nas iterações matriciais (1.3), também referenciadas na literatura como método de Schultz [43].

Apresentamos propriedades teóricas da sequência (1.2) gerada através do método de Newton (1.3), provando sua convergência quadrática herdada da iteração matricial. Além disso, com algumas condições, garantimos que os iterados capturam as informações da matriz associadas aos maiores valores singulares preservando a ordem dos mesmos, isto é, adquirindo dados fornecidos pelos elementos de baixa frequência antes dos demais. Tal propriedade é de importância na construção de aproximações estáveis para x_{LS} a partir de dados com ruído, pelo fato de que a solução de problemas discretos mal postos é dominada pelos maiores valores singulares [20, 22].

Abordamos, também, uma análise da propriedade de semi-convergência exibida pelo método proposto, culminando em uma estimativa teórica para o caso em que o princípio da discrepância de Morozov [25, 29] é utilizado para determinar o parâmetro de truncamento k .

Visando uma aplicação prática do método desenvolvido, surgem dificuldades com relação ao custo operacional e alguma estratégia de

aceleração se faz necessária. Para tanto, estudamos variantes das iterações matriciais do método de Newton [35, 42] e também combinamos o método proposto com técnicas de projeção em subespaços de dimensão finita (como os gerados pela base da transformada discreta de cosseno [22] e subespaços de Krylov [16, 22]). A segunda estratégia resultou em métodos atrativos numericamente para aplicação em problemas de grande porte que, juntamente com os resultados teóricos, consistem das contribuições mais significativas deste trabalho [5].

O texto está organizado como segue. No Capítulo 2, descrevemos conceitos preliminares relacionados aos problemas discretos mal postos, ferramentas importantes como a decomposição em valores singulares (SVD), detalhes acerca da solução x_{LS} procurada e apresentamos brevemente métodos como TSVD [20], LSQR [33] e a regularização de Tikhonov [45]. Através da ideia da semi-convergência, abordamos alguns critérios de parada para métodos iterativos, a saber: o princípio da discrepância de Morozov [25, 29] (DP), um critério baseado na L-curve [19, 25] e a regra do produto mínimo [6, 11] (MPR).

No Capítulo 3 abordamos diretamente o método iterativo proposto por este trabalho, exibindo aspectos teóricos relacionados com a convergência da sequência (1.2), a conexão do método matricial com as iterações de Newton, ordem de convergência, monotonia na norma dos resíduos e dos iterados, entre outros. Algumas referências desta parte são [8, 9, 36, 44, 47].

O Capítulo 4 apresenta uma análise da semi-convergência do método proposto que resulta em uma cota do erro para a solução regularizada sob a hipótese que o critério de parada é determinado pelo princípio da discrepância. Exibimos, também, resultados numéricos em problemas provenientes do pacote *Regularization Tools* [21] e da Galeria do MATLAB.

No Capítulo 5, o foco é buscar estratégias para acelerar as iterações do método proposto, visando contornar o custo operacional. A primeira parte do capítulo se baseia no trabalho de Pan e Schreiber [35] e busca alternativas às iterações matriciais do método de Newton. Na segunda parte, combinamos o método proposto com técnicas de projeção do problema original (1.1) em subespaços que possam conter componentes importantes da solução sem ruído x_{LS} . Ilustramos as técnicas desenvolvidas em problemas de porte médio e em um caso de recuperação de imagem com *blur* e ruído, proveniente do pacote *Restore Tools* [30].

Por fim, apresentamos as considerações finais e as referências.

2 PRELIMINARES

Muitos fenômenos nas ciências naturais e engenharia são descritos através de modelos matemáticos e são fonte de inúmeros problemas em diversos ramos das ciências aplicadas. Formalmente, consideramos processos físicos, biológicos, etc., capazes de produzir algum efeito g como consequência de um estímulo f . Estes processos são usualmente modelados por equações da forma

$$\mathcal{A}f = g, \tag{2.1}$$

em que $\mathcal{A} : \mathcal{H}_1 \rightarrow \mathcal{H}_2$ é um operador (diferencial, integral, etc.), possivelmente não linear, $f \in \mathcal{H}_1$ é a função de entrada e $g \in \mathcal{H}_2$ é a função que contém os dados do problema, sendo \mathcal{H}_1 e \mathcal{H}_2 espaços normados (como espaços de Banach ou Hilbert, por exemplo).

Quando o modelo e a função de entrada são conhecidos, computar a saída do sistema é um problema relativamente simples de ser resolvido, chamado de *problema direto*. No entanto, um cenário usual é tentar determinar a causa (ou entrada) desconhecida a partir do conhecimento do efeito resultante, i.e., buscar f sabendo que conhecemos \mathcal{A} e g . Esta situação é conhecida como *problema inverso* e corresponde à inversão do operador \mathcal{A} , caso exista.

Um complicador adicional surge em problemas práticos: é usual a função g ser dada com imprecisões, em especial quando é obtida experimentalmente e, portanto, sujeita a erros de medição. Nestas situações, somos forçados a trabalhar com uma versão aproximada g_δ da função de dados, que contém um certo nível de ruído, ou seja,

$$\|g_\delta - g\| \leq \delta,$$

em que δ obviamente depende da precisão do instrumento de medição utilizado e normalmente a escolha da norma $\|\cdot\|$ é feita levando em conta características do processo em questão. Estes são apenas alguns aspectos preliminares da teoria de problemas inversos, em franca expansão nos últimos anos. Para maiores informações, o leitor é direcionado para [14, 20, 26, 41, 46], por exemplo.

Na análise e resolução de problemas inversos, questões sobre existência, unicidade e estabilidade da solução são pontos cruciais. Por exemplo, é importante saber o quanto pequenas variações em g_δ influenciam em mudanças na solução encontrada (estabilidade). Outro aspecto

é o caso em que \mathcal{A} não é injetor, implicando que \mathcal{A}^{-1} não existe, o que leva à existência de múltiplas soluções (unicidade). Finalmente, g_δ pode nem sequer pertencer ao espaço imagem do operador, resultando em não existência de solução para (2.1).

Com base nestas ideias, o matemático francês J. S. Hadamard introduziu o conceito de problema bem-posto, no início do século XX.

Definição 2.1. *O problema (2.1) é dito bem-posto no sentido de Hadamard se são satisfeitas as seguintes condições:*

- (i) *Existência:* para cada $g \in \mathcal{H}_2$, existe $f \in \mathcal{H}_1$ tal que $\mathcal{A}f = g$.
- (ii) *Unicidade:* para cada $g \in \mathcal{H}_2$, a equação (2.1) admite uma única solução.
- (iii) *Estabilidade:* a solução f depende continuamente dos dados de entrada, i.e., o operador \mathcal{A}^{-1} é contínuo.

Caso alguma das condições acima não seja cumprida, dizemos que o problema é mal posto.

As primeiras duas condições estão relacionadas com o lado direito estar ou não na imagem do operador e com o fato de \mathcal{A} ter ou não espaço nulo não trivial. A violação destas propriedades pode ser tratada a partir do uso de alguma inversa generalizada para \mathcal{A} , de modo a computarmos soluções que são em algum sentido ótimas. Por outro lado, a questão da dependência contínua da solução com relação aos dados de entrada em geral causa maiores dificuldades, especialmente quando buscamos computar soluções numericamente. Em situações práticas e com ruído nos dados, isto significa que devemos inicialmente considerar estratégias de estabilização, como veremos com maiores detalhes adiante.

Para problemas lineares bem postos, vale observar que a dependência contínua da solução com respeito aos dados de entrada está intimamente ligada com o número de condição do operador, dado por

$$\kappa(\mathcal{A}) = \|\mathcal{A}\| \|\mathcal{A}^{-1}\|.$$

Com efeito, é fácil ver que, ao resolvermos o problema com imprecisões $\mathcal{A}(f + \delta f) = g + \delta g$, temos

$$\frac{\|\delta f\|}{\|f\|} \leq \kappa(\mathcal{A}) \frac{\|\delta g\|}{\|g\|},$$

para δg e δf representando as variações nos dados e na solução, respectivamente. Mesmo com a dependência contínua, caso $\kappa(\mathcal{A})$ seja grande,

podemos ter instabilidade nas soluções do problema, independentemente do mesmo ser bem-posto ou não. Neste sentido, caso $\kappa(\mathcal{A})$ seja próximo de 1, teremos estabilidade da solução com pequenas variações nos dados e dizemos que o problema é *bem-condicionado*. Por outro lado, com $\kappa(\mathcal{A})$ grande, dizemos que o problema é *mal condicionado*. Note que é uma noção pouco precisa, se comparada com o conceito de bem e mal posto da Definição 2.1: não é necessariamente claro, dado $\kappa(\mathcal{A})$, dizer se o problema é bem ou mal condicionado, embora este número seja frequentemente utilizado como critério comparativo entre problemas.

Dos ramos da matemática aplicada, a área de problemas inversos apresenta um dos maiores crescimentos das últimas décadas, especialmente por poder lidar com o grande número de problemas de interesse prático contemporâneo na ciência e na tecnologia. Alguns exemplos são encontrados em problemas de acústica, astronomia, tomografia computadorizada, geofísica, biologia matemática, processamento de sinais, problemas gravitacionais, inversão da equação do calor, restauração de imagens, estatística, sensoriamento remoto, ótica, identificação biométrica, entre muitos outros [20, 26, 41].

Um exemplo típico corresponde às equações integrais de Fredholm de primeira espécie, descritas na forma

$$(\mathcal{A}f)(s) := \int_a^b K(s, t)f(t)dt = g(s), \quad c \leq s \leq d, \quad (2.2)$$

em que a existência e/ou unicidade de soluções nem sempre são garantidas. Por exemplo, se consideramos que $K(s, t)$ é contínua em $[a, b] \times [c, d]$ e $f \in L^2[a, b]$, então o operador \mathcal{A} é tal que $\mathcal{A}f \in \mathcal{C}[c, d]$. Portanto, caso g não seja contínua, o problema (2.1) não tem solução, quebrando a primeira condição de Hadamard.

Agora, se considerarmos a equação

$$\int_{-\pi}^{\pi} s \operatorname{sen}(t)f(t)dt = g(s), \quad -\pi \leq s \leq \pi, \quad (2.3)$$

é fácil ver que o núcleo do operador $\mathcal{N}(\mathcal{A})$ tem dimensão infinita, pois a família de funções $f_n(s) = \operatorname{sen}(ns)$, para $n = 2, 3, \dots$, pertencem ao núcleo. Esta constatação implica que (2.3) possui infinitas soluções, tornando, também, o problema mal posto.

A questão da estabilidade também não é trivial, uma vez que da análise funcional sabemos que operadores desta forma (i.e., compactos) são tais que sua inversa é descontínua. Consequentemente, salvo o caso em que $K(s, t) = K_1(s)K_2(t)$ (conhecido como o caso do núcleo degenerado), o problema é sempre instável.

2.1 DO CONTÍNUO AO DISCRETO

Alguns problemas podem ser resolvidos analiticamente a partir da sua formulação contínua. Contudo, frequentemente precisamos de soluções numéricas; nestes casos, o ambiente contínuo impossibilita abordagens práticas e, então, é preciso discretizar o problema. Em outras palavras, precisamos expressar as funções f e g de (2.1) por vetores, e o efeito do operador \mathcal{A} sobre as funções discretas como um operador discreto, por exemplo, como uma matriz. Geralmente, o processo de discretização resulta num sistema de equações lineares da forma

$$Ax = b, \quad \text{com } A \in \mathbb{R}^{m \times n}, \quad x \in \mathbb{R}^n \quad \text{e} \quad b \in \mathbb{R}^m, \quad (2.4)$$

em que A , x e b são as representações discretas de \mathcal{A} , f e g , respectivamente. Ou, em termos mais gerais, estamos interessados em resolver o problema

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2, \quad (2.5)$$

que sempre possui solução, mesmo que $Ax = b$ seja inconsistente [27]. Como a matriz é originária da discretização de um problema mal posto, a dificuldade deste estágio é que o problema (2.5) (e sua versão na forma $Ax = b$) torna-se muito sensível a pequenas variações no vetor de dados b , que usualmente não é exato.

Observação 2.2. Para o caso de equações integrais, existem diversas maneiras de efetuar a transformação para o caso discreto, sendo os *métodos de quadratura* e os *métodos de Galerkin* alguns deles.

No primeiro caso, a função $f(t)$ pode ser discretizada considerando o seu valor em um número finito de pontos t_j , $j = 1, \dots, n$, no intervalo $[a, b]$, de modo que $x_j = f(t_j)$, e consideramos que o vetor $x \in \mathbb{R}^n$ contém os valores x_j . Agora, a integração em t é aproximada por alguma regra de quadratura utilizando o vetor x , i.e., dados os chamados *pesos da quadratura* w_j (que dependem de cada regra específica utilizada), temos que

$$\int_a^b K(s, t) f(t) dt \approx \sum_{j=1}^n w_j K(s, t_j) x_j, \quad c \leq s \leq d.$$

Agora, para remover o aspecto contínuo em s , consideramos pontos s_i , para $i = 1, \dots, m$, no intervalo $[c, d]$, gerando o lado direito b do sistema com as entradas $b_i = g(s_i)$. Deste modo, aplicando a regra de quadratura para cada um dos pontos $s = s_i$ da equação acima, temos um sistema de equações discreto como em (2.4), em que $A = [a_{ij}] \in \mathbb{R}^{m \times n}$ para $a_{ij} = w_j K(s_i, t_j)$.

Os métodos de Galerkin, por outro lado, partem de uma premissa diferente, considerando dois conjuntos de funções de base $\theta_j(t)$ e $\phi_i(s)$ e expandindo a solução procurada $f(t)$ como uma combinação das funções $\theta_j(t)$, isto é,

$$f(t) \approx \bar{f}(t) = \sum_{j=1}^n x_j \theta_j(t).$$

Então, inserimos esta informação na equação integral (2.2), obtendo

$$\int_a^b K(s, t) \sum_{j=1}^n x_j \theta_j(t) dt = g(s).$$

Em seguida, garantindo que o resíduo é ortogonal para cada uma das funções $\phi_i(s)$, ou seja,

$$\int_c^d \phi_i(s) \left(\int_a^b K(s, t) \sum_{j=1}^n x_j \theta_j(t) dt - g(s) \right) ds = 0,$$

obtemos, para cada $i \in \{1, 2, \dots, m\}$,

$$\int_c^d \int_a^b K(s, t) \phi_i(s) \sum_{j=1}^n x_j \theta_j(t) dt ds = \int_c^d g(s) \phi_i(s) ds.$$

Para formar o sistema propriamente dito, tomamos $x \in \mathbb{R}^n$ como o vetor contendo os coeficientes de cada uma das n funções de base θ_j e, similarmente, o lado direito $b = (b_i) \in \mathbb{R}^m$ e a matriz de coeficientes $A = [a_{ij}] \in \mathbb{R}^{m \times n}$ são tais que

$$b_i = \int_c^d g(s) \phi_i(s) ds \quad \text{e}$$

$$a_{ij} = \int_c^d \int_a^b K(s, t) \phi_i(s) \theta_j(t) dt ds,$$

resultando no sistema linear como em (2.4). Para informações complementares acerca de discretizações de equações integrais, veja Baker [2].

Como uma das ferramentas mais importantes para análise teórica e também para o cálculo de soluções de (2.5), apresentamos a decomposição em valores singulares (SVD) de A .

Teorema 2.3 (SVD). *Seja $A \in \mathbb{R}^{m \times n}$ tal que $\text{posto}(A) = r$. Então, existem matrizes $U = [u_1, \dots, u_m] \in \mathbb{R}^{m \times m}$ e $V = [v_1, \dots, v_n] \in \mathbb{R}^{n \times n}$ ortogonais tais que*

$$A = U \begin{bmatrix} \Sigma & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} V^T,$$

em que $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{R}^{r \times r}$ e os números σ_i (chamados de valores singulares) são ordenados de modo não-crescente $\sigma_1 \geq \dots \geq \sigma_r > 0$.

Em posse da SVD, podemos introduzir a chamada *pseudo-inversa de Moore-Penrose* [28, 36], dada por

$$A^\dagger = V \begin{bmatrix} \Sigma^\dagger & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} U^T, \quad \text{com} \quad \Sigma^\dagger = \Sigma^{-1} = \text{diag}(1/\sigma_1, \dots, 1/\sigma_r), \quad (2.6)$$

que consiste de uma versão generalizada do conceito da matriz inversa. Com efeito, a pseudo-inversa existe e é única para toda matriz $A \in \mathbb{R}^{m \times n}$, além de satisfazer diversas propriedades algébricas [36] nos moldes do que se conhece para a matriz inversa, quando a mesma existe. Vale notar que, quando A é não-singular, então $A^\dagger = A^{-1}$. A matriz pseudo-inversa nos conduz ao conjunto de soluções do problema (2.5), que sabemos ser da forma

$$\{z_{LS} = A^\dagger b + w : w \in \mathcal{N}(A)\}, \quad (2.7)$$

em que os vetores Az_{LS} correspondem à melhor aproximação de b em $\mathcal{R}(A)$ (com respeito à norma $\|\cdot\|_2$). Destas,

$$x_{LS} = A^\dagger b$$

é uma solução particular que possui a propriedade de norma mínima, isto é, para qualquer z_{LS} de (2.7),

$$\|x_{LS}\|_2 \leq \|z_{LS}\|_2.$$

Note que, de posse de x_{LS} , é possível construir todas as outras soluções, que já dependem unicamente do núcleo de A . Além disso, a solução de norma mínima x_{LS} sempre existe e é única, satisfazendo as duas primeiras condições de Hadamard para um problema ser bem-posto. A solução x_{LS} é de interesse por satisfazer os requerimentos de solução única para o problema (2.5), de modo que a maior preocupação recai sobre a questão da estabilidade do problema quando temos variações nos dados de entrada. Portanto, por padronização, quando fazemos uso da notação

$$x_{LS} = \underset{x \in \mathbb{R}^n}{\text{argmin}} \|Ax - b\|_2,$$

estamos nos referindo direta e unicamente a $x_{LS} = A^\dagger b$, como um abuso de notação.

Observação 2.4. É interessante notar que, do ponto de vista geométrico, para qualquer z_{LS} de (2.7), o vetor Az_{LS} corresponde à projeção ortogonal de b em $\mathcal{R}(A)$. De fato, é propriedade bastante conhecida [38] que

$$AA^\dagger = P_{\mathcal{R}(A)},$$

em que $P_{\mathcal{R}(A)}$ representa o projetor ortogonal em $\mathcal{R}(A)$. Esta noção reforça a menção anterior acerca das soluções em (2.7) corresponderem à melhor aproximação de b em $\mathcal{R}(A)$, uma vez que a projeção ortogonal $P_{\mathcal{R}(A)}b$ tem por característica ser um vetor em $\mathcal{R}(A)$ cuja distância euclidiana à b é a menor possível.

2.2 EFEITO DO RUÍDO NO PROBLEMA DISCRETO

Similarmente ao que acontece no caso contínuo, raramente possuímos o verdadeiro vetor b , mas sim algum vetor com imprecisões $\tilde{b} = b + e$, em que e representa o ruído proveniente de erros de medição, de discretização, entre outros. Neste caso, o vetor $\tilde{x}_{LS} = A^\dagger \tilde{b}$ corresponde à solução para o problema com dados imprecisos $Ax = \tilde{b}$. Como A é proveniente de um problema mal posto, os seus valores singulares decaem para zero sem que haja algum salto notório no seu decaimento e, mais ainda, o número de condição $\kappa(A) = \|A\|_2 \|A^\dagger\|_2$ é muito grande. Assim, \tilde{x}_{LS} é, em termos práticos, inútil como aproximação para x_{LS} , uma vez que pequenas variações nos dados de entrada tendem a gerar mudanças catastróficas no vetor de soluções encontrado. Nesta seção, buscamos elucidar a influência do ruído em \tilde{b} através de uma análise com a SVD de A .

Observação 2.5. O número de condição de uma matriz $A \in \mathbb{R}^{m \times n}$ é definido, mais geralmente, como $\kappa(A) = \|A\| \|A^\dagger\|$, para $\|\cdot\|$ qualquer norma matricial submultiplicativa. Esta quantidade foi, originalmente, introduzida para matrizes inversíveis (i.e., $A^\dagger = A^{-1}$) e, neste caso,

$$\kappa(A) = \|A\| \|A^{-1}\| \geq \|AA^{-1}\| = 1.$$

A análise se mantém a mesma que para operadores: se $\kappa(A) \approx 1$, dizemos que a matriz é *bem-condicionada* enquanto que, para $\kappa(A)$ “grande”, que é um conceito subjetivo, pertinente ao problema e ao nível de ruído em questão (veja [27, p. 415]), dizemos que A é *mal condicionada*. Quando

$\|\cdot\| = \|\cdot\|_2$, que é de maior interesse aqui, temos

$$\kappa(A) = \frac{\sigma_1}{\sigma_r},$$

frequentemente denominado *número de condição espectral*. Estas noções motivam a introdução do chamado *raio espectral*, definido para toda matriz quadrada A , dado por

$$\rho(A) = \max\{|\lambda| : \lambda \text{ é autovalor de } A\}.$$

Sabemos que $\rho(A) \leq \|A\|$, para qualquer norma matricial $\|\cdot\|$ [27], fato que gera limitantes superiores para o raio espectral, embora a estimativa seja, em geral, muito bruta. No entanto, quando a 2-norma matricial é utilizada e A é simétrica, segue que $\rho(A) = \|A\|_2$.

Note que podemos escrever a pseudo-inversa (2.6) como a soma

$$A^\dagger = \sum_{i=1}^r \sigma_i^{-1} v_i u_i^T,$$

o que implica diretamente que

$$x_{LS} = \sum_{i=1}^r \frac{u_i^T b}{\sigma_i} v_i,$$

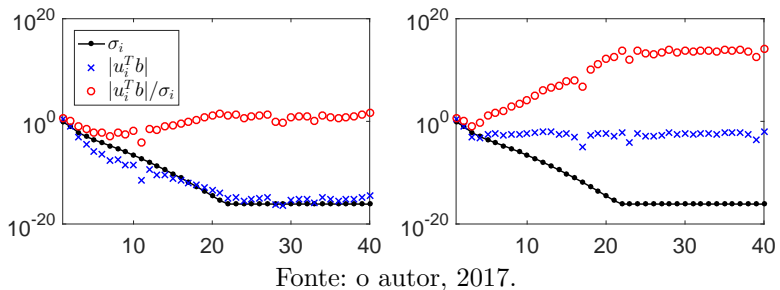
em que os números $u_i^T b$ são denominados *coeficientes de Fourier*. Com dados perturbados \tilde{b} , a solução que possuímos é, com efeito,

$$\tilde{x}_{LS} = \sum_{i=1}^r \frac{u_i^T \tilde{b}}{\sigma_i} v_i = \sum_{i=1}^r \frac{u_i^T b}{\sigma_i} v_i + \sum_{i=1}^r \frac{u_i^T e}{\sigma_i} v_i, \quad (2.8)$$

que corresponde à solução que procuramos acrescida da contribuição do ruído em \tilde{b} . Podemos notar que, para valores singulares pequenos, os coeficientes $\frac{u_i^T e}{\sigma_i}$ são grandes, fazendo com que o último somatório acima seja dominante, tornando a solução encontrada inútil como aproximação para x_{LS} . Hansen [22] definiu este tipo de problema como sendo *problema discreto mal posto*, gerando o paralelo com o que acontece no ambiente contínuo. Neste caso, para podermos construir uma solução com algum significado, as componentes do lado direito $|u_i^T \tilde{b}|$ deveriam (ao menos na média) decair mais rapidamente do que os valores singulares. Em termos grosseiros, esta propriedade é conhecida como *condição discreta de Picard* [23].

Em geral, no entanto, não é o que acontece na prática, como pode ser visto na Figura 2.1. Perceba que, com dados livres de ruído, os coeficientes de Fourier ficam equilibrados, em média, com os valores singulares, evitando que as razões $\frac{|u_i^T b|}{\sigma_i}$ atinjam altas ordens. Tal situação não ocorre no caso de dados com imprecisões, em que os valores singulares tem um decréscimo muito mais expressivo que os coeficientes de Fourier conduzindo, assim, a solução calculada a ser dominada pelo ruído. Vale mencionar que o problema utilizado, *foxgood*, é proveniente do pacote *Regularization Tools* de Hansen [21]. Esta toolbox contém diversos exemplos de problemas discretos mal postos, utilizados neste trabalho para ilustrar resultados, bem como implementações de métodos e outras rotinas auxiliares.

Figura 2.1 – Valores singulares da matriz A do problema *foxgood*, dimensão 40, coeficientes de Fourier e a razão entre eles, para b sem ruído (esquerda) e b com ruído de 1% (direita).



Fonte: o autor, 2017.

2.3 TSVD E MÉTODOS DE PROJEÇÃO

Devido à ordenação dos valores singulares, perceba que os últimos elementos da soma (2.8) tem maior chance de serem afetados pelo ruído do que os primeiros, por estarem relacionados com valores singulares menores. Portanto, se o somatório (2.8) for truncado em algum índice $k < r$, com k próximo de r , podemos amenizar o efeito do erro e no cálculo da solução; porém, se k for pequeno, deixamos de capturar informações importantes do problema. Disso segue claramente que a escolha de k deve estabelecer um balanço entre a quantidade de informação do problema que é capturada e a quantidade de erro introduzida na solução.

Com este intuito, surge o método TSVD [20] (literalmente, *SVD Truncada*), que consiste em gerar soluções da forma

$$x^{(k)} = \sum_{i=1}^k \frac{u_i^T \tilde{b}}{\sigma_i} v_i, \quad (2.9)$$

para $k \leq r$; i.e., informalmente, a TSVD simplesmente não incorpora as componentes associadas aos menores valores singulares de A na soma (2.8). O k que satisfaz os requerimentos de equilíbrio mencionados acima é chamado de *índice de truncamento*. No ambiente contínuo, a teoria que trata dos métodos de resolução de problemas mal postos é conhecida como *teoria de regularização*, o que leva a nomear os métodos envolvidos de *métodos de regularização*. No caso discreto, seguimos o mesmo paralelo. Neste sentido, o índice de truncamento k é, em essência, um parâmetro de regularização a ser determinado: é este índice que controla o balanço entre as informações que queremos do problema e o efeito do ruído, possibilitando a construção das chamadas *soluções regularizadas*, que consistem de aproximações estáveis para x_{LS} .

Na Figura 2.2 apresentamos um exemplo da funcionalidade do algoritmo TSVD, quando comparado com a solução de mínimos quadrados do problema com imprecisões $Ax = \tilde{b}$, que é dada por $\tilde{x}_{LS} = A^\dagger \tilde{b}$. Note que \tilde{x}_{LS} está dominada pelo ruído e não representa em nada uma solução razoável para o problema original $Ax = b$, i.e., livre de perturbação. No entanto, ao utilizarmos o método TSVD, a soma (2.9) é truncada em $k = 7$, gerando uma aproximação de qualidade para a solução procurada. Vale mencionar que determinar o índice de truncamento k não é tarefa simples, demandando o uso de critérios de parada especializados; abordaremos este assunto com mais detalhes futuramente.

O algoritmo TSVD faz parte da classe dos chamados *métodos de projeção*. Inicialmente, relembramos que na prática o vetor de dados é dado na forma $\tilde{b} = b + e$, com e desconhecido. Tendo isto em mente, os métodos de projeção tem por ideia básica gerar boas aproximações para

$$x_{LS} = \operatorname{argmin}_{x \in \mathbb{R}^n} \|Ax - b\|_2^2,$$

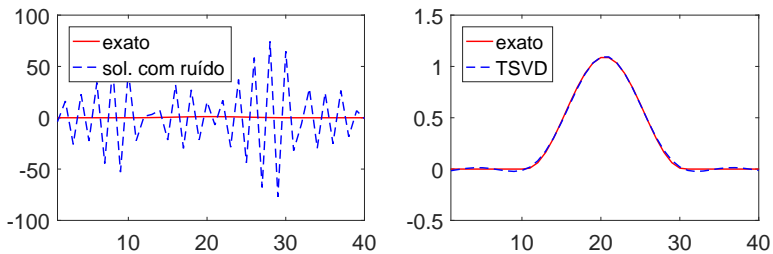
através do cálculo de vetores $x^{(k)}$ de modo que

$$x^{(k)} = \operatorname{argmin}_{x \in \mathcal{V}_k} \|Ax - \tilde{b}\|_2^2,$$

considerando que $\{\mathcal{V}_k\}$ é uma sequência de subespaços de \mathbb{R}^n que gozam das seguintes propriedades:

$$\mathcal{V}_1 \subset \mathcal{V}_2 \subset \cdots \subset \mathcal{V}_k \subset \mathcal{V}_{k+1} \subset \cdots \subset \mathbb{R}^n \quad \text{e} \quad \dim(\mathcal{V}_k) = k.$$

Figura 2.2 – Soluções $\tilde{x}_{LS} = A^\dagger \tilde{b}$ (esquerda) e $x^{(k)}$ gerada por TSVD, com $k = 7$ (direita), para problema phillips, dimensão 40, 1% de ruído nos dados de entrada.



Fonte: o autor, 2017.

No caso da TSVD, note que $\mathcal{V}_k = \text{span}\{v_1, \dots, v_k\}$, de modo que as soluções computadas pertencem ao espaço gerado pelos primeiros k vetores singulares à direita de A . Os espaços de aproximação \mathcal{V}_k dependem intrinsecamente do método em questão; devem ser escolhidos convenientemente, de modo a garantir que boas componentes da solução são capturadas. Vale observar que os métodos de projeção são, também, métodos iterativos, uma vez que geram uma sequência $\{x^{(k)}\}$ de vetores com a propriedade adicional de que $x^{(k)} \in \mathcal{V}_k$, para todo k .

Os métodos de projeção tem grande importância, especialmente quando a dimensão do problema envolvido inviabiliza o cálculo da SVD (ou de alguma outra decomposição matricial), devido ao custo computacional. Outra característica destes métodos é que, em geral, boas aproximações são obtidas com poucas iterações; quando o processo iterativo segue, as novas componentes passam a incorporar mais contribuições do ruído, o que pode desestabilizar a solução.

Na literatura existem diversos métodos de projeção, sendo que alguns mais clássicos podem ser destacados (além da TSVD):

- Método dos Gradientes Conjugados (CGLS) [10, 24];
- Método *Least Squares* QR (LSQR) [33];
- Método dos Resíduos Mínimos (MINRES) [32] (e sua modificação MR-II [17]);
- Método dos Resíduos Mínimos Generalizados (GMRES) [40] (e sua variação RRGMR [12]).

Abordar uma análise destes métodos foge de nosso interesse aqui, mas daremos foco ao algoritmo LSQR, cujas soluções utilizaremos como comparativo para os iterados gerados pelo método proposto neste trabalho.

2.3.1 Método LSQR

Proposto por Paige e Saunders [33] em 1982, o algoritmo LSQR é, atualmente, um dos mais eficientes métodos de projeção. A primeira parte do algoritmo é baseada em explorar a chamada *bidiagonalização de Golub-Kahan* (GKB) [16], que consiste em computar uma decomposição da forma $A = UBV^T$, em que $U \in \mathbb{R}^{m \times (n+1)}$ e $V \in \mathbb{R}^{n \times n}$ possuem colunas ortonormais e $B \in \mathbb{R}^{(n+1) \times n}$ é bidiagonal inferior. Esta fatoração é obtida a partir de um processo iterativo que na k -ésima iteração produz as matrizes

$$\begin{aligned} U_{k+1} &= [u_1, \dots, u_{k+1}] \in \mathbb{R}^{m \times (k+1)}, \\ V_k &= [v_1, \dots, v_k] \in \mathbb{R}^{n \times k} \quad \text{e} \\ B_k &= \begin{bmatrix} \alpha_1 & & & & & \\ \beta_2 & \alpha_2 & & & & \\ & \beta_3 & \ddots & & & \\ & & \ddots & \ddots & & \\ & & & \ddots & \alpha_k & \\ & & & & \beta_{k+1} & \end{bmatrix} \in \mathbb{R}^{(k+1) \times k} \end{aligned}$$

de modo que $AV_k = U_{k+1}B_k$. Usando que $A = UBV^T$, segue diretamente que

$$A^T U = VB^T \quad \text{e} \quad AV = UB. \quad (2.10)$$

Disto e definindo $\beta_1 v_0 \equiv \mathbf{0}$ e $\alpha_{k+1} v_{k+1} \equiv \mathbf{0}$, podemos equacionar as colunas em (2.10) usando a bidiagonalidade de B , i.e.,

$$A^T u_j = \beta_j v_{j-1} + \alpha_j v_j \quad \text{e} \quad (2.11)$$

$$A v_j = \alpha_j u_j + \beta_{j+1} u_{j+1}, \quad \text{para } j = 1, \dots, k. \quad (2.12)$$

Iniciamos o processo escolhendo um vetor $u_1 \in \mathbb{R}^m$ tal que $\|u_1\|_2 = 1$ e tomamos $\alpha_1 = \|A^T u_1\|_2$. Em seguida, construímos recursivamente os vetores $v_1, u_2, v_2, \dots, u_k, v_k, u_{k+1}$ e os elementos respectivos da matriz B_k a partir das equações (2.11) e (2.12), que levam às fórmulas, para $j = 1, \dots, k$,

$$q_j = A^T u_j - \beta_j v_{j-1}, \quad \alpha_j = \|q_j\|_2, \quad v_j = q_j / \alpha_j \quad (2.13)$$

$$p_j = A v_j - \alpha_j u_j, \quad \beta_{j+1} = \|p_j\|_2, \quad u_{j+1} = p_j / \beta_{j+1}. \quad (2.14)$$

Definição 2.6 (Subespaço de Krylov). *Sejam $A \in \mathbb{R}^{n \times n}$ e $b \in \mathbb{R}^n$. O subespaço de Krylov de ordem r gerado por A e b é definido por*

$$\mathcal{K}_r(A, b) = \text{span}\{b, Ab, A^2b, \dots, A^{r-1}b\}.$$

Levando em conta esta definição, podemos estabelecer [10] que os vetores u_j e v_j são tais que

$$u_j \in \mathcal{K}_j(AA^T, u_1) \quad \text{e} \quad v_j \in \mathcal{K}_j(A^T A, A^T u_1).$$

Agora, considerando o sistema $Ax = b$, podemos escolher u_1 como sendo $\frac{b}{\|b\|_2}$, o que nos leva a concluir que os vetores u_j e v_j pertencem aos espaços $\mathcal{K}_j(AA^T, b)$ e $\mathcal{K}_j(A^T A, A^T b)$, respectivamente. Além disso, $\beta_1 u_1 = b$ e $\alpha_1 v_1 = A^T u_1$. Disto e das equações em (2.13) e (2.14), sabemos que no passo k da bidiagonalização temos matrizes U_{k+1} , V_k e B_k satisfazendo

$$\begin{aligned} \beta_1 U_{k+1} e_1 &= b, \\ AV_k &= U_{k+1} B_k \quad \text{e} \\ A^T U_{k+1} &= V_k B_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T, \end{aligned}$$

em que e_i representa o i -ésimo vetor da base canônica de dimensão apropriada.

Neste ponto, a segunda parte do algoritmo LSQR é iniciada. A ideia é construir uma sequência (finita) de vetores $x^{(k)}$, $k \geq 1$, que aproximem a solução do problema de mínimos quadrados

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2$$

com $x^{(k)} \in \mathcal{K}_k(A^T A, A^T b)$. Como vimos acima, $\mathcal{K}_k(A^T A, A^T b) = \text{span}(V_k)$, implicando na existência de $y^{(k)} \in \mathbb{R}^k$ tal que $x^{(k)} = V_k y^{(k)}$.

Vejamos, agora, como se comporta o resíduo $r^{(k)} = Ax^{(k)} - b$. Utilizando que $AV_k = U_{k+1} B_k$ e que $\beta_1 u_1 = b$, segue que

$$\begin{aligned} r^{(k)} &= AV_k y^{(k)} - b \\ &= U_{k+1} B_k y^{(k)} - \beta_1 u_1 \\ &= U_{k+1} B_k y^{(k)} - \beta_1 U_{k+1} e_1 \\ &= U_{k+1} (B_k y^{(k)} - \beta_1 e_1). \end{aligned}$$

Portanto, como U_{k+1} tem colunas ortonormais (i.e., não influenciando na norma $\|\cdot\|_2$), temos que

$$\|r^{(k)}\|_2 = \|B_k y^{(k)} - \beta_1 e_1\|_2.$$

Deste modo, encontrar

$$x^{(k)} = \operatorname{argmin}_{x \in \mathcal{K}_k(A^T A, A^T b)} \|Ax - b\|_2$$

é equivalente a resolver

$$x^{(k)} = V_k y^{(k)}, \quad y^{(k)} = \operatorname{argmin}_{y \in \mathbb{R}^k} \|B_k y - \beta_1 e_1\|_2,$$

em que a estrutura bidiagonal de B_k simplifica o processo. Com efeito, uma maneira de resolver o problema acima é calculando a decomposição QR de B_k via rotações de Givens (que são rápidas de armazenar) e, em seguida, solucionar o sistema utilizando esta decomposição. Esta abordagem possui a vantagem de não precisar computar as QR's em cada iteração, sendo necessário apenas atualizar a QR do passo anterior e, mais ainda, é possível calcular $x^{(k)}$ via uma fórmula de recorrência que utiliza $x^{(k-1)}$. Nem os vetores v_k de V_k precisam ser armazenados, sendo necessário apenas o vetor da iteração corrente. Portanto, o algoritmo como um todo funciona de maneira eficiente, sem depender do armazenamento de muitos dados e resolvendo subproblemas simples. Maiores detalhes sobre o método LSQR, incluindo algoritmo completo, podem ser encontrados em [33].

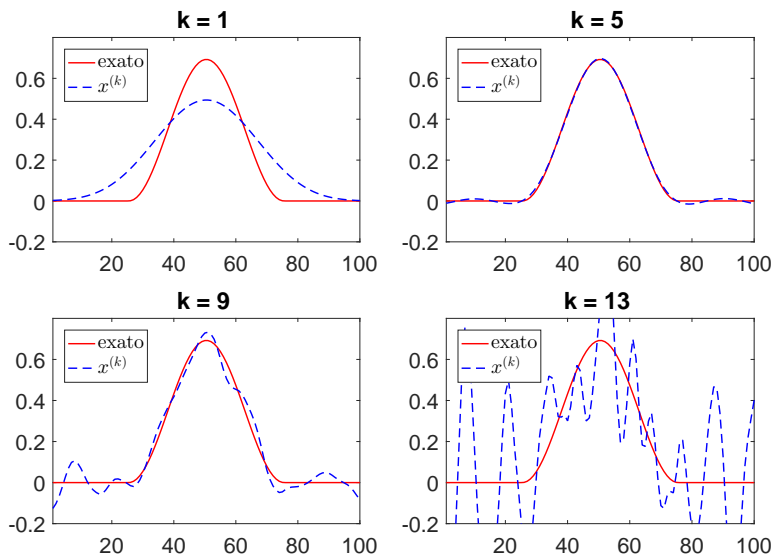
Uma dificuldade inerente ao método é a eventual perda de ortogonalidade nos vetores das matrizes U_k e V_k , problema este recorrente e que surge na implementação prática do algoritmo, devido à aritmética finita. Para evitar isso, GKB precisa ser aliado com alguma estratégia de reortogonalização nos vetores u_i e v_i como, por exemplo, através do processo de Gram-Schmidt [27, p. 307]. Certamente, isto acarreta em custo computacional adicional, que deve ser levado em conta.

Um resultado importante sobre o método LSQR diz respeito ao crescimento monotônico das normas das soluções $x^{(k)}$, ao passo que os resíduos $r^{(k)} = Ax^{(k)} - b$ decrescem da mesma maneira, conforme k cresce. Em outros termos, queremos dizer que [15, p. 1233]

$$\|x^{(k+1)}\|_2 \geq \|x^{(k)}\|_2 \quad \text{e} \quad \|r^{(k+1)}\|_2 \leq \|r^{(k)}\|_2.$$

Finalmente, vale observar que no caso de problemas práticos, os dados do lado direito se apresentam na forma $\tilde{b} = b + e$ e, assim, os iterados $x^{(k)}$ produzidos pelo método LSQR sofrem das mesmas instabilidades presentes nas soluções aproximadas por TSVD. A Figura 2.3 busca ilustrar este fenômeno. Nestas situações, faz-se necessário a escolha de um parâmetro de regularização k como para TSVD, que equilibre a

Figura 2.3 – Algumas soluções $x^{(k)}$ geradas por LSQR, para diferentes valores de k . Problema phillips, dimensão 100, 1% de ruído nos dados de entrada.



Fonte: o autor, 2017.

contribuição do ruído com as informações pertinentes fornecidas pela solução exata.

Para o exemplo da Figura 2.3, o parâmetro de regularização é dado por $k = 5$, iteração esta que melhor equilibra os dados do problema original com o efeito do ruído. Para valores de k menores que este, as soluções aproximadas não incorporam informações do problema em quantidade suficiente, apesar de pouca influência do ruído. Em contrapartida, para k 's maiores que o parâmetro de regularização, as imprecisões presentes no vetor de dados tem maior contribuição, desestabilizando os iterados computados de modo que as soluções geradas são inúteis como aproximação para a solução exata. Este fenômeno de proximidade parcial entre as soluções geradas com ruído e a solução exata x_{LS} procurada é conhecido como *semi-convergência* [31]. Sua nomenclatura é clara, uma vez que as soluções iteradas tendem a se aproximar do vetor procurado (“semi-convergir”) para, então, divergirem devido à contribuição do ruído. Identificar a semi-convergência, no entanto,

não é uma tarefa trivial, muitas vezes associada diretamente com o problema e/ou método em questão. A próxima seção tem por objetivo apresentar algumas ferramentas que buscam detectar o parâmetro de regularização para métodos iterativos, a fim de capturar o iterado com melhor equilíbrio entre as informações do problema original e o ruído associado ao lado direito do sistema.

2.4 CRITÉRIOS DE PARADA PARA MÉTODOS ITERATIVOS

O sucesso do método iterativo quando aplicado a problemas perturbados está intimamente ligado com o uso de algum critério de parada eficiente, capaz de detectar o parâmetro de regularização procurado ou, ao menos, uma boa aproximação deste. Nesta seção, apresentaremos alguns critérios de parada presentes na literatura para métodos iterativos e que serão de essencial importância nos capítulos subsequentes. Faremos uso de três critérios, sendo que somente um deles se baseia no conhecimento da norma do ruído e acoplada a \tilde{b} enquanto os outros dois são regras heurísticas. Certamente, estes não são os únicos critérios existentes, mas apenas alguns representantes de uma vasta gama presente na literatura (veja, por exemplo, [20, 25]).

2.4.1 Princípio da discrepância (DP)

O *princípio de discrepância* (DP) tem como ideia básica a parada na primeira iteração k em que

$$\|r^{(k)}\|_2 = \|Ax^{(k)} - \tilde{b}\|_2 \leq \tau \|e\|_2,$$

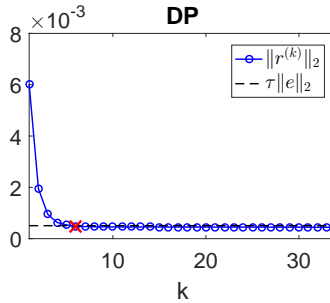
com $\tau \gtrsim 1$. O método é atribuído a Morozov [25, 29] e é um dos principais métodos que se baseia no conhecimento do tamanho do erro e adicionado ao vetor de dados b . Em algumas situações na prática é possível estimar $\|e\|_2$ quando não se conhece o valor da perturbação, tornando o critério ainda aplicável.

De maneira intuitiva, a escolha do parâmetro de regularização k por DP infere que a solução regularizada deve possuir resíduo associado da ordem do ruído nos dados. Uma vez que não temos acesso ao sistema livre de perturbações, é razoável pensar que a solução computada não pode ser melhor do que algo que depende do tamanho do ruído.

Na Figura 2.4, apresentamos um exemplo de aplicação de DP, em que os iterados $x^{(k)}$ são gerados através da TSVD. Note que o parâmetro k escolhido é o primeiro que possui resíduo associado abaixo da linha tracejada, como determina o critério.

O princípio da discrepância é um dos poucos critérios de parada existentes na literatura que possui análise de erro [25, 29], garantindo que, caso $\|e\|_2 \rightarrow 0$, então a solução regularizada computada tende a x_{LS} (solução do problema livre de ruído).

Figura 2.4 – Princípio da discrepância aplicado ao problema *deriv2*, dimensão 500, com ruído de 1% e $\tau = 1.1$, com (\times) marcando o parâmetro de regularização escolhido.



Fonte: o autor, 2017.

2.4.2 L-curve

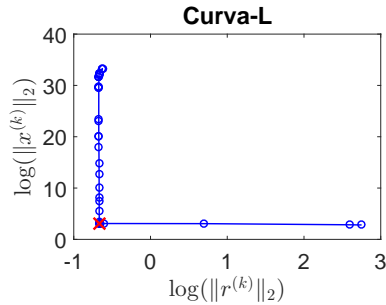
A *L-curve* (ou *curva-L*) consiste, de maneira grosseira, no gráfico gerado pela norma da solução e pela norma do resíduo correspondente. Sua ideia geral foi formalizada por Hansen [19, 25] e, aqui, consiste na curva discreta definida pelos pontos

$$\left(\log(\|r^{(k)}\|_2), \log(\|x^{(k)}\|_2) \right) \quad (2.15)$$

para $k = 1, \dots, k_{max}$, com k_{max} natural definido. Na Figura 2.5 abaixo, temos um exemplo de L-curve para o problema *shaw*, em que consideramos dimensão $n = 500$, perturbação de 1% nos dados iniciais e $k_{max} = 35$, com iterados gerados pela TSVD. A forma da curva deixa clara a escolha para a sua nomeação.

Uma característica deste critério de parada que não é compartilhada por DP é a necessidade de comportamento monotônico na norma dos iterados e dos resíduos. Mais especificamente, L-curve assume que o método iterativo respeita que $\|x^{(k+1)}\|_2 \geq \|x^{(k)}\|_2$ e $\|r^{(k+1)}\|_2 \leq \|r^{(k)}\|_2$. Sem este comportamento, não há garantia que L-curve vá selecionar um bom parâmetro de regularização.

Figura 2.5 – Critério da L-curve aplicado ao problema *shaw*, dimensão 500, com ruído de 1%, com (×) marcando o parâmetro de regularização escolhido.



Fonte: o autor, 2017.

Intuitivamente, a escolha do parâmetro de regularização k se dá através da determinação do ponto de maior curvatura da curva definida pelos pontos de (2.15), isto é, buscamos k tal que o ponto $(\log(\|r^{(k)}\|_2), \log(\|x^{(k)}\|_2))$ esteja próximo ao “canto” da curva-L, pois procura minimizar tanto $\|r^{(k)}\|_2$ quanto controlar a ordem de $\|x^{(k)}\|_2$. Aumentando ou reduzindo k a partir deste ponto, temos aumento em alguma das normas, sem reduzir a outra significativamente, o que leva à escolha pela parada no canto. A pergunta é: como escolher tal k ? Como complicador adicional, é comum haver um acúmulo de iterações nas proximidades da esquina da curva (veja a Figura 2.5), dificultando a busca por k . Além disso, nem sempre o formato em “L” da curva será tão claro e que permita uma boa ideia (visualmente) quanto à iteração de parada.

A rotina *corner*, que faz parte da toolbox *Regularization Tools* [21], tem por objetivo identificar (ou aproximar) o ponto de maior curvatura da curva-L, a partir dos dados $\|r^{(k)}\|_2$ e $\|x^{(k)}\|_2$. Em termos grosseiros, a ideia geral do código é construir um *spline* com os dados inseridos e, a partir disto, estimar o canto da curva-L.

Uma dificuldade inerente deste critério é a necessidade de efetuar um grande número de iterações para, então, poder estimar o parâmetro k . No exemplo da Figura 2.5, efetuamos 35 iterações da TSVD para determinarmos k em torno da 6ª iteração. Este empecilho não ocorre com DP, em que a parada é determinada na iteração corrente, sem depender de pré-computar diversas iterações. Vale também comentar que o critério da curva-L não possui análise de erro (como acontece

com DP), o que o torna um critério heurístico, baseado na “esperança” (suportada pelo comportamento típico das quantidades envolvidas) de que o iterado escolhido será de boa qualidade. Mesmo assim, a curva-L consiste de um dos critérios mais populares e tem sido utilizado com sucesso em diversos problemas.

2.4.3 Regra do produto mínimo (MPR)

Uma outra abordagem foi apresentada recentemente por Bazán, Borges e Cunha [6, 11], na chamada *regra do produto mínimo* (MPR). Nos trabalhos, é apresentada uma maneira heurística (tal qual o critério da curva-L) para a busca da iteração de parada k , que se baseia na ideia de minimizar a função discreta

$$\Psi_k = \|r^{(k)}\|_2 \|x^{(k)}\|_2,$$

isto é, buscamos \hat{k} tal que

$$\hat{k} = \operatorname{argmin} \Psi_k.$$

A regra do produto mínimo pode ser vista como uma contrapartida discreta da regra de Regińska [39], a qual procura por um canto da curva-L contínua definida para o problema de Tikhonov. Para maiores informações nesta estratégia, o leitor é direcionado para Bazán [4] e Bazán e Francisco [7].

Do ponto de vista prático, MPR pode ser implementado através do monitoramento das diferenças

$$\nabla \Psi_k = \Psi_{k+1} - \Psi_k, \quad \text{para } k \geq 1,$$

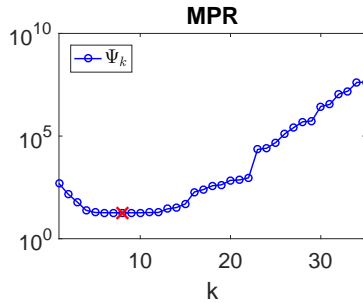
e selecionando o primeira índice k tal que $\nabla \Psi_{k-1} < 0$ e $\nabla \Psi_k > 0$, tornando o uso do critério facilmente introduzível no processo iterativo. Esta maneira de escolher o parâmetro de regularização implica que, no máximo, $k + 1$ iterações serão necessárias. Esta é uma diferença essencial do que acontece com o critério dado pela curva-L, que demanda computação prévia de um grande número de iterados. Deste modo, MPR verifica na iteração corrente pelo mínimo de Ψ_k , através das quantidades $\nabla \Psi_k$, fazendo uso de dados do iterado atual para saber se o anterior minimiza Ψ_k .

Na Figura 2.6, apresentamos a curva discreta Ψ_k para o problema *gravity*, dimensão 500, com iterados $x^{(k)}$ gerados através da TSVD. Enfatizamos que tanto MPR (tal qual L-curve) assume que o método

iterativo em questão apresenta propriedade de monotonia nas normas dos iterados e dos resíduos, isto é,

$$\|x^{(k+1)}\|_2 \geq \|x^{(k)}\|_2 \quad \text{e} \quad \|r^{(k+1)}\|_2 \leq \|r^{(k)}\|_2.$$

Figura 2.6 – MPR aplicado ao problema `gravity`, dimensão 500, com ruído de 1%, com (×) marcando o parâmetro de regularização escolhido.



Fonte: o autor, 2017.

2.5 ASPECTOS DA REGULARIZAÇÃO DE TIKHONOV

Além dos métodos de projeção, uma outra classe importante são os chamados *métodos de penalidade*, sendo a regularização de Tikhonov um dos mais conhecidos. Como vimos para TSVD, para dados com ruído a solução $\tilde{x}_{LS} = A^\dagger \tilde{b}$ tende a ser de alta ordem em norma, uma vez que os valores singulares decaem mais rapidamente que os coeficientes de Fourier associados. Em termos intuitivos, os métodos de penalidade visam minimizar o resíduo ao mesmo tempo que controlam a norma da solução aproximada. Para a abordagem de Tikhonov [45], no caso mais simples, buscamos computar aproximações para a solução do problema original (2.5) através da formulação

$$x_\lambda = \operatorname{argmin}_{x \in \mathbb{R}^n} \left\{ \|Ax - \tilde{b}\|_2^2 + \lambda^2 \|x\|_2^2 \right\}, \quad (2.16)$$

em que λ é o parâmetro de regularização (que é contínuo, diferentemente dos índices discretos k para os métodos iterativos). Este parâmetro controla a quantidade de regularização imposta ao problema e deve ser escolhido com cuidado. Por exemplo, considerando situações extremas,

caso $\lambda = 0$, estamos no ambiente usual do problema de mínimos quadrados (i.e., sem impor regularização), enquanto que $\lambda = \infty$ implica que o minimizador é $x = \mathbf{0}$. Desta maneira, a grande dificuldade da regularização de Tikhonov se encontra em determinar λ de modo que x_λ de (2.16) aproxime bem a solução x_{LS} do problema original (2.5).

Num escopo mais geral para problemas discretos, a estratégia de regularização de Tikhonov toma a forma

$$x_{L,\lambda} = \operatorname{argmin}_{x \in \mathbb{R}^n} \left\{ \|Ax - \tilde{b}\|_2^2 + \lambda^2 \|L(x - x_0)\|_2^2 \right\}, \quad (2.17)$$

em que, usualmente, x_0 é uma aproximação inicial da solução de (2.5) (caso não esteja disponível, tomamos $x_0 = \mathbf{0}$) e a matriz $L \in \mathbb{R}^{p \times n}$ costuma introduzir dados adicionais do problema original. Uma situação recorrente é L ser matriz da primeira ou segunda derivadas discreta, o que costuma ter efeito suavizante nos problemas, em especial para equações integrais, uma vez que esperamos (em muitos casos) diferenciabilidade da solução, ao menos. Por exemplo, é usual considerar as matrizes

$$L_1 = \begin{bmatrix} -1 & 1 & & & \\ & \ddots & \ddots & & \\ & & & -1 & 1 \end{bmatrix} \in \mathbb{R}^{(n-1) \times n},$$

que representa a discretização pelo método de diferenças finitas para o operador diferencial de primeira ordem, e

$$L_2 = \begin{bmatrix} 1 & -2 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & & 1 & -2 & 1 \end{bmatrix} \in \mathbb{R}^{(n-2) \times n},$$

que, similarmente, consiste de uma aproximação por diferenças finitas centrais para o operador diferencial de segunda ordem.

Por questões de nomenclatura, caso $L = I$, dizemos que (2.17) é um *problema de Tikhonov na forma padrão*. Para os outros casos, isto é, para $L \neq I$, dizemos que (2.17) é um *problema de Tikhonov na forma geral*.

Observação 2.7. Perceba que podemos considerar $x_0 = \mathbf{0}$ em (2.17), uma vez que é sempre possível realizar a mudança de variável $\hat{x} = x - x_0$, resultando no problema

$$\hat{x}_{L,\lambda} = \operatorname{argmin}_{\hat{x} \in \mathbb{R}^n} \left\{ \|A\hat{x} - \tilde{b}\|_2^2 + \lambda^2 \|L\hat{x}\|_2^2 \right\},$$

com $\hat{b} = \tilde{b} - Ax_0$. Em seguida, para retornar ao problema original, basta tomar $x_{L,\lambda} = \hat{x}_{L,\lambda} + x_0$.

Deste modo, pensando que $x_0 = \mathbf{0}$, o problema (2.17) pode ser escrito em duas formas equivalentes:

$$\min_{x \in \mathbb{R}^n} \left\| \begin{bmatrix} A \\ \lambda L \end{bmatrix} x - \begin{bmatrix} \tilde{b} \\ \mathbf{0} \end{bmatrix} \right\|_2 \quad \text{e} \quad (A^T A + \lambda^2 L^T L) x = A^T \tilde{b}, \quad (2.18)$$

em que esta última representa as assim chamadas *equações normais regularizadas*. Para garantirmos unicidade na solução das equações acima, pedimos que $\mathcal{N}(A) \cap \mathcal{N}(L) = \{\mathbf{0}\}$. Agora, retornando ao caso mais simples (2.16) e usando a SVD de A , segue que

$$x_\lambda = (A^T A + \lambda^2 I)^{-1} A^T \tilde{b} = \sum_{i=1}^r \frac{\sigma_i^2}{\sigma_i^2 + \lambda^2} \frac{u_i^T \tilde{b}}{\sigma_i} v_i, \quad (2.19)$$

e podemos notar diretamente o efeito regularizante de λ na solução computada: para os primeiros índices i , pela ordenação dos valores singulares, temos que $\frac{\sigma_i^2}{\sigma_i^2 + \lambda^2} \approx 1$, de modo que as informações contidas nestes valores singulares são mantidas no somatório; no entanto, para i “grande”, devido ao rápido decaimento dos valores singulares σ_i de A , temos $\frac{\sigma_i^2}{\sigma_i^2 + \lambda^2} \approx 0$, removendo o efeito dos pequenos valores singulares de A (associados com os maiores coeficientes de Fourier em módulo) do somatório (2.19). Novamente, é necessário equilíbrio, pois se λ é tal que $\lambda^2 \approx 0$, recaímos em (2.8), computando a solução \tilde{x}_{LS} dominada pelo ruído; por outro lado, se λ é grande, temos tendência a perder as informações que os valores singulares fornecem do problema, uma vez que $\frac{\sigma_i^2}{\sigma_i^2 + \lambda^2}$ pode ser virtualmente nulo mesmo para os i 's iniciais. Uma análise semelhante é válida para $L \neq I$, caso presente em (2.18).

As soluções geradas tanto por TSVD quanto por Tikhonov podem ser expressas como soluções SVD filtradas dadas por

$$x_\alpha = \sum_{i=1}^r \phi_{\alpha,i} \frac{u_i^T \tilde{b}}{\sigma_i} v_i = V_r \Phi_\alpha \Sigma^\dagger U_r^T \tilde{b}, \quad \Phi_\alpha = \text{diag}(\phi_{\alpha,1}, \dots, \phi_{\alpha,r}),$$

em que x_α é a solução regularizada, para α algum parâmetro de regularização, e os números $\phi_{\alpha,i}$ são conhecidos como *fatores de filtro*. Para TSVD e Tikhonov, os fatores de filtro são dados, respectivamente, por

$$\phi_{k,i} = \begin{cases} 1, & \text{se } i \leq k \\ 0, & \text{caso contrário} \end{cases} \quad \text{e} \quad \phi_{\lambda,i} = \frac{\sigma_i^2}{\sigma_i^2 + \lambda^2}.$$

Sua denominação é autoexplicativa, uma vez que estes fatores buscam “filtrar” o efeito do ruído na solução regularizada. Para λ escolhido apropriadamente, o efeito dos filtros de Tikhonov é similar ao truncamento da expansão em vetores singulares conduzida pela TSVD. Enfatizamos que determinar tal λ demanda o uso de critérios de parada capazes de lidar com λ como parâmetro contínuo. Na literatura, existem muitos métodos bem estabelecidos com este objetivo, mas não os abordaremos pois fogem do escopo deste trabalho. Observamos, no entanto, que os critérios de parada apresentados na seção anterior (DP, L-curve e MPR) possuem versões contínuas para a determinação do parâmetro de Tikhonov; algumas referências neste sentido são [4, 7, 20, 25, 29], tanto para estes critérios quanto para outros.

3 O MÉTODO DE NEWTON

Dado um sistema de equações lineares $Ax = b$, com $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$ e $b \in \mathbb{R}^m$, estamos interessados em calcular

$$x_{LS} = \operatorname{argmin}_{x \in \mathbb{R}^n} \|Ax - b\|_2,$$

ou, de fato, aproximações estáveis para este vetor, como explanado no capítulo anterior. No entanto, computar $x_{LS} = A^\dagger b$ demanda o conhecimento de efetuar operações com A^\dagger , cujo custo para o seu cálculo é computacionalmente comparável ao da matriz inversa A^{-1} (quando a mesma existe). Atacar o problema por este caminho, portanto, não é recomendável e, muitas vezes, simplesmente inviável, especialmente quando a matriz A envolvida é de grande porte.

Com o intuito de contornar tal dificuldade, partimos para o uso de métodos iterativos para a pseudo-inversa, i.e., métodos que constroem uma sequência de matrizes $\{X_k\}$ de modo que $X_k \rightarrow A^\dagger$ quando $k \rightarrow \infty$. Note que, de posse de algum iterado X_k , conseguimos uma aproximação natural para x_{LS} através do produto $X_k b$. Uma outra alternativa é construir uma sequência de vetores da forma

$$x^{(k)} = X_k b, \tag{3.1}$$

sem computar as matrizes X_k diretamente, mas tendo em mente que buscamos, de fato, calcular aproximações do “efeito” de tais matrizes quando aplicadas ao vetor b . Do ponto de vista numérico, é evidente que fazer uso da sequência vetorial é mais econômico que computar a sequência matricial para, então, construir a aproximação $X_k b$. Neste trabalho, apresentamos uma família de métodos iterativos para computar as matrizes X_k com convergência de ordem $p \geq 2$, com foco especial no caso $p = 2$, que possui um paralelo com o método de Newton.

O objetivo deste capítulo é apresentar uma maneira de incorporar o efeito das matrizes X_k na iteração vetorial (3.1), bem como resultados teóricos (de convergência e monotonia das normas das soluções e dos resíduos, por exemplo) e aspectos numéricos relacionados (de implementação computacional e custo operacional), para a situação em que a sequência $\{X_k\}$ é gerada através do método de Newton.

3.1 MÉTODO ITERATIVO PARA A PSEUDO-INVERSA

Ao longo dos anos, diversos métodos iterativos para o cálculo da inversa de Moore-Penrose A^\dagger foram desenvolvidos e implementados. Entre eles, uma classe específica tem atraído bastante atenção, pela sua alta ordem de convergência, que apresentaremos a seguir. Tomemos, antes, alguns cuidados com relação à notação.

Definição 3.1. *Sejam $A, B \in \mathbb{R}^{m \times n}$. Chamamos de imagem do par (A, B) e núcleo do par (A, B) , respectivamente,*

$$\begin{aligned} \mathcal{R}(A, B) &= \{Y \in \mathbb{R}^{m \times n} : Y = AXB, \text{ para } X \in \mathbb{R}^{n \times m}\} \quad e \\ \mathcal{N}(A, B) &= \{X \in \mathbb{R}^{n \times m} : AXB = \mathbf{0}\}. \end{aligned}$$

Observação 3.2. Note que os conjuntos $\mathcal{R}(A, B)$ e $\mathcal{N}(A, B)$ podem ser vistos como extensões dos conjuntos usuais $\mathcal{R}(\cdot)$ e $\mathcal{N}(\cdot)$. Com efeito, é possível mostrar [9, p. 110] que, com o produto interno no espaço de matrizes $\mathbb{R}^{m \times n}$ definido por

$$\langle A, B \rangle = \text{trace}(B^T A) = \sum_{i=1}^m \sum_{j=1}^n a_{ij} b_{ij},$$

os conjuntos $\mathcal{R}(A, B)$ e $\mathcal{N}(A^T, B^T)$ são subespaços complementares de $\mathbb{R}^{m \times n}$, para cada par de matrizes $A, B \in \mathbb{R}^{m \times n}$. Note que este resultado é análogo à sua versão usual, que afirma que $\mathcal{R}(A)$ e $\mathcal{N}(A^T)$ são complementares.

Para os métodos iterativos, se $X_k \rightarrow A^\dagger$, introduzimos a sequência de resíduos R_k associados às matrizes X_k definidos por

$$R_k = P_{\mathcal{R}(A)} - AX_k, \quad k = 0, 1, 2, \dots,$$

em geral utilizados para determinar a ordem de convergência dos métodos, do ponto de vista teórico. Lembrando que $P_{\mathcal{R}(A)} = AA^\dagger$, é claro que R_k converge para $\mathbf{0}$, dado que a sequência X_k converge para A^\dagger .

O resultado seguinte apresenta condições e ordem de convergência para a família de métodos iterativos matriciais que faremos uso neste trabalho.

Teorema 3.3. [9, p. 273] *Seja $\mathbf{0} \neq A \in \mathbb{R}^{m \times n}$ e considere que a aproximação inicial $X_0 \in \mathbb{R}^{n \times m}$ e seu resíduo $R_0 = P_{\mathcal{R}(A)} - AX_0$ satisfazem $X_0 \in \mathcal{R}(A^T, A^T)$ e $\rho(R_0) < 1$, respectivamente. Então, para qualquer inteiro $p \geq 2$ a sequência*

$$X_{k+1} = X_k \left(I + T_k + T_k^2 + \dots + T_k^{p-1} \right), \quad k = 0, 1, 2, \dots, \quad (3.2)$$

em que $T_k = I - AX_k$, converge para A^\dagger quando $k \rightarrow \infty$ e a sequência de resíduos correspondente satisfaz

$$\|R_{k+1}\| \leq \|R_k\|^p, \quad k = 0, 1, 2, \dots, \quad (3.3)$$

para qualquer norma matricial submultiplicativa $\|\cdot\|$.

Note que a desigualdade (3.3) acima garante que o método em questão possui ordem de convergência p , para qualquer inteiro $p \geq 2$. Esta constatação implica que podemos gerar boas aproximações para A^\dagger rapidamente, desde que p seja suficientemente grande. Por outro lado, o custo computacional cresce com p , algo que precisa ser levado em conta. Enfatizamos que, neste trabalho, o foco principal é o método iterativo para o caso $p = 2$, que possui convergência quadrática ao mesmo tempo que minimiza o custo operacional, quando comparado com os métodos gerados para $p > 2$.

Com relação às condições para convergência presentes no Teorema 3.3, exigimos que $X_0 \in \mathcal{R}(A^T, A^T)$ e $\rho(R_0) < 1$. Para a primeira condição, uma escolha bastante usual é tomar $X_0 = \beta A^T$, para $\beta \in \mathbb{R}$, pois, como A^\dagger satisfaz as equações de Penrose [36], temos:

$$\beta A^T = \beta(AA^\dagger A)^T = A^T \beta (A^\dagger)^T A^T \in \mathcal{R}(A^T, A^T).$$

Observação 3.4. Encontrar aproximações iniciais no espaço $\mathcal{R}(A^T, A^T)$ não é tarefa trivial. Algumas outras possibilidades para o iterando X_0 são apresentadas por Zoblec [47] (no contexto de operadores lineares em espaços de Hilbert), quando a matriz pode ser dividida de maneira específica, como $A = M + N$ ou $A = D + S + Q$, em que as matrizes envolvidas devem satisfazer determinadas propriedades. No artigo, no entanto, computar X_0 envolve o cálculo de pseudo-inversas, algo que buscamos evitar. Na maioria das referências, a escolha $X_0 = \beta A^T$ é uma unanimidade [8, 35, 37, 38, 44].

O próximo passo é calibrar β de modo que a segunda condição, $\rho(R_0) < 1$, também seja satisfeita.

Proposição 3.5. [9, p. 276] *Sejam $\mathbf{0} \neq A \in \mathbb{R}^{m \times n}$, $\beta \in \mathbb{R}$ e $R_0 = P_{\mathcal{R}(A)} - \beta AA^T$. Então, $\rho(R_0) < 1$ se, e somente se,*

$$0 < \beta < \frac{2}{\rho(AA^T)}. \quad (3.4)$$

Concluindo, tomando $X_0 = \beta A^T$ e escolhendo β como em (3.4), garantimos que o método apresentado no Teorema 3.3 converge. Deste

modo, consideramos X_0 dado desta maneira como “escolha padrão” para o iterando inicial, considerada deste ponto em diante. Determinar β satisfazendo (3.4), entretanto, é uma operação delicada, uma vez que calcular $\rho(AA^T)$ demanda custo computacional elevado e, muitas vezes, é inviável. Tendo isto em mente, abordamos algumas maneiras de escolher a constante β na subseção seguinte.

3.1.1 Cálculo de β

Inferir algum valor para β consiste, primariamente, em encontrar bons limitantes superiores para $\rho(AA^T)$, já que o cálculo de autovalores é tarefa complicada na prática. Caso a matriz A esteja disponível explicitamente, algumas possibilidades surgem através do uso de propriedades das normas matriciais. Por exemplo, lembrando que $\rho(\cdot) = \|\cdot\|_2$ (para matrizes simétricas) e que $\|\cdot\|_2 \leq \|\cdot\|_F$ [16], temos

$$\rho(AA^T) = \|AA^T\|_2 \leq \|A\|_2 \|A^T\|_2 = \|A\|_2^2 \leq \|A\|_F^2,$$

o que implica que $\beta = 1/\|A\|_F^2$ satisfaz (3.4) trivialmente:

$$0 < \frac{1}{\|A\|_F^2} < \frac{2}{\rho(AA^T)}.$$

Outra possibilidade, utilizando do fato de que $\|\cdot\|_2 \leq \sqrt{\|\cdot\|_1 \|\cdot\|_\infty}$ [16], leva a escolher

$$\beta = \frac{1}{\|A\|_1 \|A\|_\infty}.$$

Estes são somente alguns dos valores possíveis para β , o que leva ao questionamento: qual a influência de tal constante nas iterações? Isto é, qualquer β arbitrário no intervalo de interesse (3.4) tem o mesmo efeito sobre o método iterativo? Como resposta, Ben-Israel e Cohen [8] mostraram que resíduo $\|R_k\|_2$ é minimizado quando

$$\beta = \beta_{op} = \frac{2}{\sigma_1^2 + \sigma_r^2}.$$

Como $\sigma_1^2 = \rho(AA^T)$, veja que computar este β é tarefa ainda mais custosa que aproximar diretamente o raio espectral de AA^T , que já estamos evitando. Por outro lado, para problemas discretos mal postos, $\sigma_r \approx 0$, o que leva à constatação de que devemos buscar uma boa aproximação para $\rho(AA^T)$ simplesmente, uma vez que, neste caso,

$$\sigma_1^2 + \sigma_r^2 \approx \sigma_1^2 = \rho(AA^T).$$

Uma outra abordagem permite encontrar aproximações para o raio espectral $\rho(AA^T)$ mesmo no caso em que A não é dada explicitamente, sendo somente acessada através de produtos matriz-vetor. Para tanto, faremos uso da chamada *bidiagonalização de Lanczos* [16] (também conhecida como *bidiagonalização de Golub-Kahan-Lanczos* [1]), que tem por objetivo encontrar matrizes

$$\begin{aligned} U &= [u_1, \dots, u_m], & U^T U &= I_m, & \text{e} \\ V &= [v_1, \dots, v_n], & V^T V &= I_n, \end{aligned}$$

tais que $U^T A V = B$ representa a bidiagonalização de $A \in \mathbb{R}^{m \times n}$, em que $B \in \mathbb{R}^{m \times n}$ é da forma

$$B = \begin{bmatrix} \alpha_1 & \beta_1 & & & & & \\ & \alpha_2 & \ddots & & & & \\ & & \ddots & \ddots & & & \\ & & & \ddots & \ddots & & \\ & & & & \ddots & & \\ & & & & & \beta_{n-1} & \\ & & & & & & \alpha_n \end{bmatrix}.$$

Note que este procedimento é similar ao processo GKB, apresentado na Subseção 2.3.1. Com efeito, a partir de operações análogas às descritas para GKB, podemos elaborar um processo iterativo equacionando as colunas de $AV = UB$ e $A^T U = VB^T$, que descrevemos no Algoritmo 3.1.

Algoritmo 3.1 Bidiagonalização de Lanczos

- 1: Escolha $v_1 \in \mathbb{R}^n$ tal que $\|v_1\|_2 = 1$ e tome $\beta_0 = 0$
 - 2: **Para** $k = 1, 2, \dots$ **faça**
 - 3: $u_k \leftarrow Av_k - \beta_{k-1}u_{k-1}$
 - 4: $\alpha_k \leftarrow \|u_k\|_2$
 - 5: $u_k \leftarrow u_k/\alpha_k$
 - 6: $v_{k+1} \leftarrow A^T u_k - \alpha_k v_k$
 - 7: $\beta_k \leftarrow \|v_{k+1}\|_2$
 - 8: $v_{k+1} \leftarrow v_{k+1}/\beta_k$
 - 9: **Fim Para**
-

O método de Lanczos descrito aqui é predecessor a GKB, e ambos foram inspirados pelo processo de tridiagonalização de Lanczos [1, 16], que visa transformar matrizes Hermitianas em tridiagonais, a fim de (entre outras funcionalidades) aproximar os autovalores da mesma.

Do ponto de vista prático, o processo de Lanczos é finito e, para uma bidiagonalização completa, a parada do algoritmo é feita através do monitoramento de β_k , como observado por Golub e Van Loan [16]: as iterações são mantidas somente enquanto $\beta_k \neq 0$. Aqui, no entanto, não estamos interessados em calcular B por completo, mas sim fazer uso das propriedades de aproximação dos autovalores de A que tal matriz possui. De $U^T A U = B$, segue diretamente que $U^T A A^T U = B B^T$, implicando em

$$\rho(AA^T) = \rho(BB^T),$$

uma vez que U não influencia na 2-norma matricial, pelas colunas ortonormais. Portanto, quando consideramos a matriz gerada até a k -ésima iteração, dada por

$$B_k = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ & \alpha_2 & \ddots & & \\ & & \ddots & \beta_{k-1} & \\ & & & & \alpha_k \end{bmatrix} \in \mathbb{R}^{k \times k},$$

não é surpresa que $\rho(B_k B_k^T)$ aproxime o valor de $\rho(AA^T)$. Com efeito, é fato conhecido na literatura e na prática que mesmo poucas iterações são suficientes para que $\rho(B_k B_k^T) \approx \rho(AA^T)$. Esta propriedade é reflexo do processo de tridiagonalização de Lanczos, para o qual o Algoritmo 3.1 estabelece um paralelo (mais informações em [1, 16]). Na Tabela 3.1, ilustramos alguns destes resultados de boa aproximação.

Tabela 3.1 – Comparativo entre o raio espectral de AA^T e de $B_k B_k^T$, para $k = 5$ e matrizes de ordem 1000.

	foxgood	phillips	heat	shaw	gravity	baart
$\rho(AA^T)$	0.6575	33.6741	0.1261	8.9599	41.7212	10.4244
$\rho(B_k B_k^T)$	0.6575	32.7506	0.1261	8.9599	41.7212	10.4244

Fonte: o autor, 2017.

Note que, em geral, o raio espectral de $B_k B_k^T$ praticamente coincide com o de AA^T , mesmo com somente $k = 5$ iterações do algoritmo de Lanczos. A vantagem desta estratégia se encontra no fato de que computar a 2-norma matricial da matriz tridiagonal simétrica $B_k B_k^T$ de ordem k é computacionalmente mais amigável que operar com $AA^T \in \mathbb{R}^{m \times m}$, ainda mais se estamos tratando com problemas de grande porte.

Assim, lembrando que a aproximação do raio espectral de AA^T através das matrizes B_k se dá inferiormente, isto é,

$$\|B_k B_k^T\|_2 = \rho(B_k B_k^T) \lesssim \rho(AA^T),$$

podemos tomar β como

$$\beta = \frac{1}{K \|B_k B_k^T\|_2},$$

em que K é constante ligeiramente maior que 1 ($K = 1.2$ ou $K = 1.5$, por exemplo), assegurando que o número $K \|B_k B_k^T\|_2$ é, de fato, limitante superior para o raio espectral de AA^T . Portanto, via algoritmo de Lanczos, conseguimos computar β satisfazendo os requerimentos de (3.4) com baixo custo operacional e, ao mesmo tempo, podendo tratar com matrizes dadas implicitamente.

3.2 MÉTODO DE NEWTON MATRICIAL

Considerando $p = 2$ no método dado em (3.2), a iteração toma a forma

$$X_{k+1} = X_k(2I - AX_k), \quad (3.5)$$

que apresenta convergência de boa qualidade (quadrática) ao mesmo tempo que demanda o menor número de operações matriciais se comparado com os métodos da família (3.2). Deste ponto em diante, este é o método em foco, para o qual construiremos a seqüência $x^{(k)} = X_k b$.

É interessante, no entanto, notar a relação entre o método de segunda ordem (3.5) e o método de Newton. Caso estejamos buscando o recíproco de um número real $a \neq 0$, podemos tentar computar uma raiz (que é única) para a função

$$f(x) = a - \frac{1}{x}.$$

Aplicando o método de Newton na função acima, construímos os iterados

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} = x_k - \frac{a - 1/x_k}{1/x_k^2} = x_k(2 - ax_k), \quad (3.6)$$

e é facilmente verificável que $\{x_k\}$ converge sempre que o iterando inicial satisfizer

$$0 < x_0 < \frac{2}{a}, \quad \text{se } a > 0, \quad \text{ou} \quad \frac{2}{a} < x_0 < 0, \quad \text{se } a < 0.$$

Em ambos os casos ($a > 0$ ou $a < 0$), se tomarmos $x_0 = \beta a$, então a condição acima implica que

$$0 < \beta < \frac{2}{a^2}$$

para obtermos convergência. Tanto esta desigualdade quanto a iteração (3.6) são paralelos exatos para o processo iterativo dado em (3.5) e para a escolha da aproximação inicial $X_0 = \beta A^T$, com β como em (3.4). Sabemos que $x_k \rightarrow x^*$ quadraticamente (propriedade do método de Newton), em que o limite é tal que $f(x^*) = 0$, isto é, $x^* = a^{-1}$. Esta iteração pode ser generalizada para buscar a inversa de uma matriz não-singular M pela função matricial [34]

$$f(X) = M - X^{-1}.$$

Posteriormente, o caso mais geral é apresentado pelo método (3.5), que converge para a pseudo-inversa de $A \in \mathbb{R}^{m \times n}$. Vale observar que, caso A seja não-singular, então $A^\dagger = A^{-1}$, de modo que este caso é contemplado, sem perda alguma, pelo método matricial quadrático. Por estas conexões, a iteração em (3.5) será referenciada, deste ponto em diante, como *método de Newton (matricial)*.

Iniciamos os trabalhos desta seção com o seguinte lema, que apresenta algumas propriedades do método de Newton, necessárias mais à frente.

Lema 3.6. *A sequência X_k definida em (3.5) satisfaz as seguintes propriedades:*

- (i) $I - AX_{k+1} = (I - AX_k)^2$, para todo $k \geq 0$.
- (ii) AX_k e $X_k A$ são Hermitianas, para todo $k \geq 0$.
- (iii) $X_k A A^\dagger = X_k$ e $A^\dagger A X_k = X_k$, para todo $k \geq 0$.
- (iv) $\|X_k\|_2 \leq \|X_{k+1}\|_2$, para todo $k \geq 1$, e $\|X_k\|_2$ converge por baixo para $\|A^\dagger\|_2$ quando $k \rightarrow \infty$.

Demonstração. Destes itens, apenas (iv) apresenta maior dificuldade; os outros são elementares e bastante conhecidos.

$$(i) \quad I - AX_{k+1} = I - AX_k(2I - AX_k) = I - AX_k - AX_k(I - AX_k) = (I - AX_k)(I - AX_k) = (I - AX_k)^2.$$

- (ii) Para $X_0 = \beta A^T$, é claro que $AX_0 = (AX_0)^T$. Tomando a hipótese de indução $AX_k = (AX_k)^T$, temos que $(AX_{k+1})^T = (AX_k(2I - AX_k))^T = (2I - AX_k)AX_k = AX_k(2I - AX_k) = AX_{k+1}$. Analogamente, temos que $X_k A$ é Hermitiana.
- (iii) Como AA^\dagger é Hermitiana e $AA^\dagger A = A$ [36], temos que $X_0 AA^\dagger = \beta A^T (AA^\dagger)^T = \beta (AA^\dagger A)^T = \beta A^T = X_0$. Agora, considerando a hipótese de indução $X_k AA^\dagger = X_k$, veremos que vale o caso $k + 1$. De fato, $X_{k+1} AA^\dagger = 2X_k AA^\dagger - X_k AX_k AA^\dagger = 2X_k - X_k AX_k = X_{k+1}$. Pelo mesmo raciocínio, segue que $A^\dagger AX_k = X_k$.
- (iv) Usando a SVD de A de (2.3) e $X_0 = \beta A^T$, temos que

$$X_0 = V \begin{bmatrix} \beta \Sigma & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} U^T = V \begin{bmatrix} \Omega_0 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} U^T,$$

em que $\Omega_0 = \text{diag}(\beta\sigma_1, \dots, \beta\sigma_r)$. Agora, tendo em vista a hipótese de indução

$$X_k = V \begin{bmatrix} \Omega_k & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} U^T, \quad (3.7)$$

em que $\Omega_k = \text{diag}(\omega_1^{(k)}, \dots, \omega_r^{(k)})$, com $\omega_i^{(0)} = \beta\sigma_i$, segue que

$$\begin{aligned} X_{k+1} &= 2X_k - X_k AX_k \\ &= V \begin{bmatrix} 2\Omega_k - \Omega_k \Sigma \Omega_k & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} U^T, \end{aligned} \quad (3.8)$$

o que implica diretamente que

$$\omega_i^{(k+1)} = 2\omega_i^{(k)} - \sigma_i \left(\omega_i^{(k)} \right)^2. \quad (3.9)$$

É fato conhecido [44] que a sequência acima converge para $1/\sigma_i$ dado que

$$0 < \omega_i^{(0)} < 2/\sigma_i, \quad (3.10)$$

que é válida pelo fato de β satisfazer (3.4). Por outro lado, seguindo um raciocínio parecido com o item (i) deste lema, a equação (3.9) implica que

$$1 - \sigma_i \omega_i^{(k+1)} = \left(1 - \sigma_i \omega_i^{(k)} \right)^2, \quad \text{para todo } k \geq 0. \quad (3.11)$$

Disto e de (3.9), segue que

$$\omega_i^{(k+1)} - \omega_i^{(k)} = \omega_i^{(k)} \left(1 - \sigma_i \omega_i^{(k)} \right) \geq 0, \quad \text{para todo } k \geq 1, \quad (3.12)$$

o que prova que $\omega_i^{(k)}$ converge monotonicamente por baixo para $1/\sigma_i$. Em particular, para $i = r$, de (3.12) temos que

$$\|X_k\|_2 = \omega_r^{(k)} \leq \omega_r^{(k+1)} = \|X_{k+1}\|_2 \quad (3.13)$$

e, mais ainda,

$$\lim_{k \rightarrow \infty} \omega_r^{(k)} = 1/\sigma_r = \|A^\dagger\|_2.$$

Assim, completamos a demonstração. \square

Uma propriedade interessante do método pode ser extraída diretamente deste lema, a partir do item (iii) e lembrando que $A^\dagger = A^\dagger A A^\dagger$ [36]:

$$\begin{aligned} A^\dagger - X_{k+1} &= A^\dagger - X_k - X_k + X_k A X_k \\ &= A^\dagger A A^\dagger - A^\dagger A X_k - X_k A A^\dagger + X_k A X_k \\ &= A^\dagger A (A^\dagger - X_k) - X_k A (A^\dagger - X_k) \\ &= (A^\dagger - X_k) A (A^\dagger - X_k). \end{aligned}$$

Veja que esta relação reafirma a convergência quadrática já garantida pelo Teorema 3.3 para o método de Newton, pois

$$\|A^\dagger - X_{k+1}\|_2 \leq \|A\|_2 \|A^\dagger - X_k\|_2^2.$$

Agora, veja que o item (iv) do lema afirma a convergência quadrática dos $\omega_i^{(k)}$. Apesar disso, é bem sabido que a convergência dos $\omega_i^{(k)}$ pode ser lenta mesmo para matrizes relativamente bem condicionadas, como discutido por Söderström e Stewart [44]. Com efeito, $\omega_i^{(k)}$ não pode convergir antes de $2 \log_2(\sigma_1/\sigma_i)$ iterações [44], e veja que este número pode atingir alta ordem caso os σ_i 's sejam pequenos.

A boa notícia é que, para problemas discretos mal postos, frequentemente os maiores valores singulares de A são suficientes para construir soluções aproximadas de qualidade para o problema de mínimos quadrados associado a $Ax = b$ [20]. Ou seja, na prática não precisamos aguardar a convergência total para termos boas aproximações para o problema. Mas, para isto acontecer, precisamos garantir que os $\omega_i^{(k)}$'s associados aos maiores valores singulares convirjam antes dos demais. Esta análise da ordenação dos $\omega_i^{(k)}$'s é o foco do próximo resultado. Considere, assim, a função

$$f(\omega) = 2\omega - \sigma_i \omega^2$$

e o iterando inicial dado por $\omega^{(0)} = \beta\sigma_i$ escolhido de acordo com (3.10), o que deixa claro que a ordenação com que ocorre a convergência depende de β . Agora, para verificar como isso ocorre, assumamos que para algum i temos

$$\frac{\sigma_1}{\sqrt{2}} < \sigma_i < \sigma_1,$$

o que implica que $1/\sigma_i^2 < 2/\sigma_1^2$. Portanto, tomando $\beta = 1/\sigma_i^2$, garantimos convergência das matrizes X_k e, ao mesmo tempo,

$$\omega_i^{(0)} = \beta\sigma_i = \frac{1}{\sigma_i}.$$

Ou seja, neste caso, $1/\sigma_i$ é automaticamente capturado pelo chute inicial e o recíproco dos valores singulares próximos a σ_i deveriam convergir rapidamente também, nas primeiras iterações.

O teorema seguinte formaliza esta análise, afirmando que se β for escolhido corretamente, então a sequência $\omega_i^{(k)}$ converge para $1/\sigma_i$ de tal maneira que os recíprocos associados com maiores valores singulares de A são capturados primeiro.

Teorema 3.7. *Seja $e_i^{(k)} = \frac{1}{\sigma_i} - \omega_i^{(k)}$ e assumamos que $\beta < \frac{1}{\sigma_i^2}$. Então, para todo $k > 0$, temos que*

$$e_i^{(k)} \leq e_{i+1}^{(k)}, \quad \text{para } i = 1, 2, \dots, r-1.$$

Demonstração. A hipótese sobre β implica que $0 < \omega_i^{(0)} = \beta\sigma_i < \sigma_i/\sigma_1^2 \leq 1/\sigma_i$. Isto, juntamente com

$$\omega_1^{(0)} \geq \omega_2^{(0)} \geq \dots \geq \omega_r^{(0)},$$

implicam que

$$0 < 1 - \sigma_i\omega_i^{(0)} \leq 1 - \sigma_{i+1}\omega_{i+1}^{(0)} < 1, \quad i = 1, 2, \dots, r-1. \quad (3.14)$$

Agora, de (3.11) aplicada repetidamente, segue que $\sigma_i\omega_i^{(k)} = 1 - (1 - \sigma_i\omega_i^{(0)})^{2^k}$. Como $\omega_i^{(k+1)} = \left[1 + (1 - \sigma_i\omega_i^{(k)})\right]\omega_i^{(k)}$, veja que

$$\omega_i^{(1)} = \left[1 + (1 - \sigma_i\omega_i^{(0)})^{2^1-1}\right]\omega_i^{(0)}.$$

Considerando a hipótese de indução

$$\omega_i^{(k)} = \sum_{\ell=0}^{2^k-1} (1 - \sigma_i\omega_i^{(0)})^\ell \omega_i^{(0)}, \quad (3.15)$$

segue que

$$\begin{aligned}\omega_i^{(k+1)} &= \left[1 + \left(1 - \sigma_i \omega_i^{(k)}\right)\right] \omega_i^{(k)} \\ &= \left[1 - \left(1 + \sigma_i \omega_i^{(0)}\right)^{2^k}\right] \sum_{\ell=0}^{2^k-1} \left(1 - \sigma_i \omega_i^{(0)}\right)^\ell \omega_i^{(0)} \\ &= \sum_{\ell=0}^{2^{k+1}-1} \left(1 - \sigma_i \omega_i^{(0)}\right)^\ell \omega_i^{(0)},\end{aligned}$$

de modo que a equação (3.15) é válida. Por outro lado, levando em conta que a série real

$$\frac{1}{1-c} = \sum_{\ell=0}^{\infty} c^\ell$$

é convergente sempre que $|c| < 1$, tomando $c = \left(1 - \sigma_i \omega_i^{(0)}\right)$ obtemos

$$\frac{1}{\sigma_i} = \sum_{\ell=0}^{\infty} \left(1 - \sigma_i \omega_i^{(0)}\right)^\ell \omega_i^{(0)}.$$

Disto e de (3.15), temos

$$\begin{aligned}e_i^{(k)} &= \frac{1}{\sigma_i} - \omega_i^{(k)} = \sum_{\ell=2^k}^{\infty} \left(1 - \sigma_i \omega_i^{(0)}\right)^\ell \omega_i^{(0)} \\ &= \left(1 - \sigma_i \omega_i^{(0)}\right)^{2^k} \sum_{\ell=0}^{\infty} \left(1 - \sigma_i \omega_i^{(0)}\right)^\ell \omega_i^{(0)} \\ &= \frac{\left(1 - \sigma_i \omega_i^{(0)}\right)^{2^k}}{\sigma_i}.\end{aligned}\tag{3.16}$$

Desde que $\sigma_i \geq \sigma_{i+1}$, usando (3.14) temos que

$$e_i^{(k)} = \frac{\left(1 - \sigma_i \omega_i^{(0)}\right)^{2^k}}{\sigma_i} \leq \frac{\left(1 - \sigma_{i+1} \omega_{i+1}^{(0)}\right)^{2^k}}{\sigma_{i+1}} = e_{i+1}^{(k)},$$

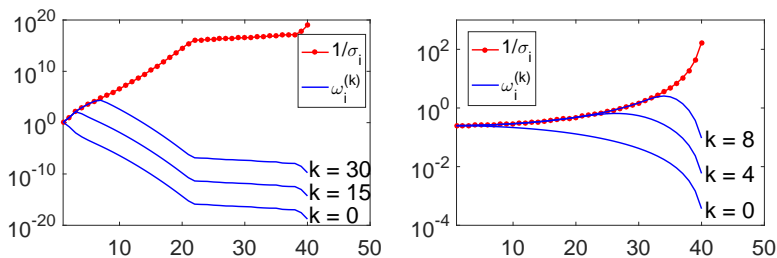
o que completa a demonstração. \square

A conclusão mais importante do teorema é que, caso escolhamos $\beta < 1/\sigma_1^2$, as informações relacionadas aos maiores valores singulares de

A convergem antes, com a propriedade de que $\omega_i^{(k)}$ converge para $1/\sigma_i$ preservando a ordem natural dos valores singulares.

A velocidade de convergência, no entanto, depende também do condicionamento da matriz envolvida. Por exemplo, considere a matriz proveniente do problema `foxgood` e também $T = \text{tridiag}(-1, 2, -1)$, que surge frequentemente na discretização do operador diferencial de segunda ordem via método de diferenças finitas centrais, ambas com dimensão $n = 40$. A matriz de `foxgood` possui número de condicionamento da ordem de 10^{18} , considerado severo, ainda mais que a dimensão do problema é pequena. Já T é tal que $\kappa(T) \approx 10^3$, deixando claro que seu mal condicionamento é leve. O comportamento dos recíprocos dos valores singulares e dos $\omega_i^{(k)}$'s para estas duas matrizes é ilustrado na Figura 3.1. O decaimento dos valores singulares para `foxgood` ocorre com extrema rapidez, gerando dificuldades à velocidade de convergência, de modo que um número bastante elevado de iterações são necessárias para capturar a mesma quantidade de valores singulares que T exhibe em poucas iterações. Note que, para T , os $\omega_i^{(k)}$'s praticamente capturam todo o espectro já nas iterações iniciais. Vale observar que os recíprocos $1/\sigma_i$ de `foxgood` são, aproximadamente, 10 ordens de grandeza maiores que os mesmos para a matriz T .

Figura 3.1 – Comportamento dos recíprocos dos valores singulares e das seqüências $\omega_i^{(k)}$ para as matrizes de `foxgood` (esquerda) e $T = \text{tridiag}(-1, 2, -1)$ (direita), ambas com dimensão 40 e $\beta = 1/(\sigma_1^2 + \sigma_r^2)$.



Fonte: o autor, 2017.

3.3 O MÉTODO ITERATIVO VETORIAL

Como mencionado anteriormente, nosso interesse na seqüência matricial gerada pelo método de Newton (3.5) se encontra na possibilidade

de construir uma sequência de vetores da forma

$$x^{(k)} = X_k b \in \mathbb{R}^n, \text{ para } k \geq 0. \quad (3.17)$$

Note que, como $x^{(k)} - x_{LS} = (X_k - A^\dagger)b$, segue claramente que $x^{(k)} \rightarrow x_{LS}$, uma vez que a sequência matricial converge para a pseudo-inversa. Portanto, $\{x^{(k)}\}$ gera vetores que tendem a aproximar a solução de norma mínima do problema de mínimos quadrados associado ao sistema $Ax = b$, solução esta de interesse tanto teórico quanto prático.

O primeiro resultado desta seção afirma que a sequência gerada por (3.17) possui segunda ordem de convergência, assim como a versão matricial do método de Newton que a precede, propriedade esperada naturalmente pela forma como a sequência vetorial é construída.

Teorema 3.8. *Seja $x_{LS} = A^\dagger b$. Então, para todo $k \geq 1$, é válido que*

$$\|x_{LS} - x^{(k+1)}\|_2 \leq C \|x_{LS} - x^{(k)}\|_2^2,$$

em que C é uma constante não negativa que depende de x_{LS} .

Demonstração. Levando em conta que (3.11) é equivalente a

$$\frac{1}{\sigma_i} - \omega_i^{(k+1)} = \sigma_i \left(\frac{1}{\sigma_i} - \omega_i^{(k)} \right)^2, \text{ para } k \geq 0,$$

e que a SVD de A implica que

$$x_{LS} = V \begin{bmatrix} \Sigma^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} U^T b,$$

usando a equação (3.8) temos que

$$\begin{aligned} x_{LS} - x^{(k+1)} &= V \begin{bmatrix} \Sigma^{-1} - \Omega_{k+1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} U^T b \\ &= V \begin{bmatrix} \Sigma (\Sigma^{-1} - \Omega_k)^2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} U^T b \\ &= V_r \Sigma (\Sigma^{-1} - \Omega_k)^2 U_r^T b, \end{aligned}$$

em que $V_r = [v_1, \dots, v_r]$ e $U_r = [u_1, \dots, u_r]$, para $r = \text{posto}(A)$. Agora, defina ξ como a matriz diagonal $r \times r$ cujas entradas são:

$$\xi_{ii} = \begin{cases} |u_i^T b|, & \text{se } |u_i^T b| \neq 0 \\ 1, & \text{caso contrário} \end{cases}.$$

Claramente, ξ é não-singular. Disto e da equação acima, segue que

$$\begin{aligned} \|x_{LS} - x^{(k+1)}\|_2 &= \|\Sigma (\Sigma^{-1} - \Omega_k)^2 U_r^T b\|_2 \\ &= \|\Sigma \xi^{-1} (\Sigma^{-1} - \Omega_k)^2 \xi U_r^T b\|_2 \\ &\leq \|\Sigma \xi^{-1}\|_2 \|(\Sigma^{-1} - \Omega_k)^2 \xi U_r^T b\|_2 \\ &\leq \|\Sigma \xi^{-1}\|_2 \|(\Sigma^{-1} - \Omega_k)^2 \xi U_r^T b\|_1. \end{aligned}$$

Tomando $C = \|\Sigma \xi^{-1}\|_2$, finalizamos a demonstração ao notar que

$$\begin{aligned} \|(\Sigma^{-1} - \Omega_k)^2 \xi U_r^T b\|_1 &= \sum_{i=1}^r \left(\sigma_i^{-1} - \omega_i^{(k)} \right)^2 |u_i^T b|^2 \\ &= \|x_{LS} - x^{(k)}\|_2^2. \end{aligned}$$

□

Agora, pelo Lema 3.6 e pelo Teorema 3.3, sabemos que a versão matricial do método de Newton possui a propriedade de monotonia na norma dos iterados e dos resíduos, sendo o primeiro não-decrescente e o segundo não-crescente, i.e., $\|X_k\|_2 \leq \|X_{k+1}\|_2$ e $\|R_{k+1}\|_2 \leq \|R_k\|_2$. O resultado seguinte afirma que o mesmo ocorre com a sequência vetorial (3.17). Lembramos que esta informação é crucial para o uso de critérios de parada heurísticos, tais como L-curve e MPR.

Teorema 3.9. *Sejam $x^{(k)}$ sequência de vetores gerada pelo método de Newton e $r^{(k)} = b - Ax^{(k)}$ sequência dos resíduos correspondentes. Então, as seguintes propriedades são válidas:*

- (i) $\|x^{(k+1)}\|_2 \geq \|x^{(k)}\|_2$, para todo $k \geq 1$.
- (ii) $\|r^{(k+1)}\|_2 \leq \|r^{(k)}\|_2$, para todo $k \geq 0$.

Demonstração. (i) Usando a equação (3.7), segue que

$$x^{(k)} = X_k b = \sum_{i=1}^r \omega_i^{(k)} (u_i^T b) v_i. \quad (3.18)$$

Portanto, disto e de (3.13), temos

$$\|x^{(k+1)}\|_2^2 = \sum_{i=1}^r \left(\omega_i^{(k+1)} \right)^2 |u_i^T b|^2 \geq \sum_{i=1}^r \left(\omega_i^{(k)} \right)^2 |u_i^T b|^2 = \|x^{(k)}\|_2^2.$$

(ii) Observe que, da escolha de β no intervalo $0 < \beta < 2/\sigma_1^2$, segue que

$$0 < \beta\sigma_i^2 < 2\frac{\sigma_i^2}{\sigma_1^2} \leq 2,$$

inferindo que $-1 < 1 - \beta\sigma_i^2 < 1$. Como $\sigma_i\omega_i^{(0)} = \beta\sigma_i^2$, temos que $-1 < 1 - \sigma_i\omega_i^{(0)} < 1$. Disto e da equação (3.11), concluímos que

$$1 - \sigma_i\omega_i^{(k)} = \left(1 - \sigma_i\omega_i^{(k-1)}\right)^2 = \dots = \left(1 - \sigma_i\omega_i^{(0)}\right)^{2^k} < 1, \quad (3.19)$$

para todo $k \geq 1$. Utilizando este fato e novamente a equação (3.11), segue que

$$\left(1 - \sigma_i\omega_i^{(k+1)}\right)^2 = \left(1 - \sigma_i\omega_i^{(k)}\right)^4 \leq \left(1 - \sigma_i\omega_i^{(k)}\right)^2, \quad (3.20)$$

válido para todo $k \geq 0$. Agora, pela SVD de A e por (3.7), temos que

$$\begin{aligned} \|r^{(k+1)}\|_2^2 &= \|b - Ax^{(k+1)}\|_2^2 = \|(I - \Sigma\Omega_{k+1})U^T b\|_2^2 \\ &= \sum_{i=1}^r \left(1 - \sigma_i\omega_i^{(k+1)}\right)^2 |u_i^T b|^2 + \|b^\perp\|_2^2, \end{aligned}$$

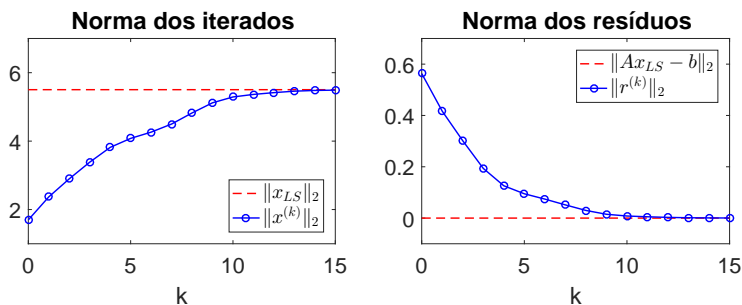
em que b^\perp representa a componente de b que não pertence ao espaço coluna de A . Assim, disto e (3.20), segue que

$$\begin{aligned} \|r^{(k+1)}\|_2^2 &= \sum_{i=1}^r \left(1 - \sigma_i\omega_i^{(k+1)}\right)^2 |u_i^T b|^2 + \|b^\perp\|_2^2 \\ &\leq \sum_{i=1}^r \left(1 - \sigma_i\omega_i^{(k)}\right)^2 |u_i^T b|^2 + \|b^\perp\|_2^2 \\ &= \|r^{(k)}\|_2^2, \end{aligned}$$

o que completa a demonstração. □

Como ilustração do resultado do teorema acima, apresentamos a Figura 3.2, uma confirmação visual à afirmação do teorema, acerca da monotonia da norma dos iterados e dos resíduos, para o problema-teste heat, com dimensão 500.

Figura 3.2 – Comportamento de $\|x^{(k)}\|_2$ (à esquerda) e dos resíduos $\|b - Ax^{(k)}\|_2$ (à direita), problema heat, dimensão 500.



Fonte: o autor, 2017.

Observação 3.10. O item (i) do Teorema 3.9 pede que $k \geq 1$, que é consequência da desigualdade (3.12) também ser válida somente para $k \geq 1$. Com efeito, podemos construir facilmente um contra-exemplo verificando que é possível ocorrer $\|x^{(0)}\|_2 > \|x^{(1)}\|_2$. Tomando, e.g., o sistema $Ax = b$, com $A = I_n$ e $b = e_1$, para e_i o i -ésimo vetor da base canônica de \mathbb{R}^n , é fácil verificar que

$$x_0 = \beta e_1 \quad \text{e} \quad x_1 = \beta(2 - \beta)e_1.$$

Agora, basta calibrar β corretamente. Para convergência, precisamos que $0 < \beta < 2/\rho(I_n I_n^T) = 2$. Disto, é direto verificar que qualquer β no intervalo $(1, 2)$ satisfaz

$$\|x^{(0)}\|_2 = \beta > \beta(2 - \beta) = \|x^{(1)}\|_2,$$

o que conclui o exemplo.

Para finalizar a seção, apresentamos um outro resultado de convergência, com estimativas para o erro relativo para algumas escolhas de β no iterando inicial.

Teorema 3.11. *Seja $x_{LS} = A^\dagger b$. Então, a norma do erro $\|x_{LS} - x^{(k+1)}\|_2$ decresce monotonicamente com k , ou seja, para $k \geq 1$, temos que*

$$\|x_{LS} - x^{(k+1)}\|_2 \leq \|x_{LS} - x^{(k)}\|_2.$$

Mais ainda, são válidas as estimativas:

$$\frac{\|x_{LS} - x^{(k)}\|_2}{\|x_{LS}\|_2} \leq \begin{cases} \left(\frac{\kappa(A)^2 - 1}{\kappa(A)^2}\right)^{2^k}, & \text{se } \beta > \frac{1}{\sigma_1^2} \\ \left(\frac{\kappa(A)^2 - 1}{\kappa(A)^2 + 1}\right)^{2^k}, & \text{se } \beta = \frac{2}{\sigma_1^2 + \sigma_r^2} \end{cases}. \quad (3.21)$$

Demonstração. Observe que a equação (3.11) implica que

$$\frac{1}{\sigma_i} - \omega_i^{(k+1)} = \left(1 - \sigma_i \omega_i^{(k)}\right) \left(\frac{1}{\sigma_i} - \omega_i^{(k)}\right) \leq \frac{1}{\sigma_i} - \omega_i^{(k)}, \text{ para } k \geq 1,$$

em que a última desigualdade segue de (3.19). Disto e de (3.7), temos que

$$\begin{aligned} \|x_{LS} - x^{(k+1)}\|_2^2 &= \sum_{i=1}^r \left(\frac{1}{\sigma_i} - \omega_i^{(k+1)}\right)^2 |u_i^T b|^2 \\ &\leq \sum_{i=1}^r \left(\frac{1}{\sigma_i} - \omega_i^{(k)}\right)^2 |u_i^T b|^2 \\ &= \|x_{LS} - x^{(k)}\|_2^2, \end{aligned}$$

como enunciado. Agora, através de um argumento similar e utilizando (3.16), segue que

$$\|x_{LS} - x^{(k)}\|_2^2 = \sum_{i=1}^r \frac{\left(1 - \sigma_i \omega_i^{(0)}\right)^{2(2^k)}}{\sigma_i^2} |u_i^T b|^2 \leq \left(1 - \sigma_r \omega_r^{(0)}\right)^{2(2^k)} \|x_{LS}\|_2^2,$$

ou seja,

$$\frac{\|x_{LS} - x^{(k)}\|_2}{\|x_{LS}\|_2} \leq \left(1 - \sigma_r \omega_r^{(0)}\right)^{2^k}.$$

Como $\omega_r^{(0)} = \beta \sigma_r$, as estimativas em (3.21) são consequência das escolhas de β . Por exemplo, para $\beta > 1/\sigma_1^2$, temos

$$1 - \sigma_r \omega_r^{(0)} \leq 1 - \frac{\sigma_r^2}{\sigma_1^2} = \frac{\kappa(A)^2 - 1}{\kappa(A)^2}.$$

Já para o segundo valor de β ,

$$1 - \sigma_r \omega_r^{(0)} = \frac{\sigma_1^2 + \sigma_r^2 - 2\sigma_r^2}{\sigma_1^2 + \sigma_r^2} = \frac{\kappa(A)^2 - 1}{\kappa(A)^2 + 1},$$

o que finaliza a demonstração. \square

3.4 ASPECTOS COMPUTACIONAIS

Até o momento, tratamos de diversas propriedades relativas à sequência de vetores em (3.17), dado por $x^{(k)} = X_k b$, com $X_{k+1} = X_k(2I - AX_k)$, em que $X_0 = \beta A^T$, mas sem de fato considerar alguma implementação computacional. A sequência vetorial pode ser construída, da maneira mais direta, a partir do conhecimento de X_k e, em seguida, efetuando o produto $X_k b$. Infelizmente, esta estratégia nada acrescenta em ganho de operações.

No entanto, o intuito de utilizar a sequência vetorial é a possibilidade de lidar com a construção de vetores em cada iteração, e não de matrizes, como na estratégia direta. Assim, esperamos reduzir o esforço computacional do método de Newton, ao mesmo tempo que gozamos das suas propriedades de boa convergência.

Inicialmente, note que a iteração de Newton pode ser reescrita como

$$X_{k+1} = X_k(2I - AX_k) = X_k + X_k(I - AX_k) = X_k + (I - X_k A)X_k. \quad (3.22)$$

Caso assumimos a notação $U_k = (I - X_k A)$, por uma abordagem semelhante à utilizada no Lema 3.6, item (i), verificamos que

$$U_{k+1} = U_k^2 = U_{k-1}^{2^2} = \dots = U_0^{2^{k+1}}.$$

Portanto, disto e (3.22), temos que a sequência em (3.17) pode ser apresentada através da recorrência

$$x^{(k+1)} = X_{k+1} b = X_k b + U_k X_k b = x^{(k)} + U_0^{2^k} x^{(k)}, \quad (3.23)$$

em que $x^{(0)} = \beta A^T b$ e $U_0 = (I - \beta A^T A) \in \mathbb{R}^{n \times n}$. A iteração em (3.23) é o que consideramos como o *método de Newton (vetorial)*, o qual sugere dois tipos de implementação: uma explícita, que consiste em um produto matriz-matriz por iteração; e uma implícita, que lida com 2^k produtos matriz-vetor envolvendo U_0 na k -ésima iteração. Ambos são apresentados nos Algoritmos 3.2 e 3.3 a seguir.

Do ponto de vista operacional, o Algoritmo 3.2 efetua $\mathcal{O}(n^3)$ operações por iteração, resultado da necessidade de computar um produto de matrizes quadradas. Por outro lado, o algoritmo seguinte efetua algo em torno de $\mathcal{O}(2^k n^2)$ operações na k -ésima iteração (devido aos 2^k produtos matriz-vetor com U_0) e, pelo crescimento exponencial de 2^k , estas iterações podem superar n^3 operações com certa facilidade. Por outro lado, se os produtos matriz-vetor puderem ser efetuados eficientemente

Algoritmo 3.2 Método de Newton explícito

Entrada: A, b **Saída:** Aproximação $x^{(*)}$ de x_{LS}

- 1: Compute β tal que $0 < \beta < \frac{2}{\rho(AA^T)}$ ▷ Subseção 3.1.1
 - 2: $U_0 \leftarrow I - \beta A^T A$
 - 3: $x^{(0)} \leftarrow \beta A^T b$
 - 4: **Para** $k = 0, 1, 2, \dots$ **faça**
 - 5: $x^{(k+1)} \leftarrow x^{(k)} + U_0 x^{(k)}$
 - 6: **Se** (critério de parada satisfeito) **então**
 - 7: $x^{(*)} \leftarrow x^{(k+1)}$
 - 8: **Fim Se**
 - 9: $U_0 \leftarrow U_0^2$
 - 10: **Fim Para**
-

Algoritmo 3.3 Método de Newton implícito

Entrada: A, b **Saída:** Aproximação $x^{(*)}$ de x_{LS}

- 1: Compute β tal que $0 < \beta < \frac{2}{\rho(AA^T)}$ ▷ Subseção 3.1.1
 - 2: $x^{(0)} \leftarrow \beta A^T b$
 - 3: **Para** $k = 0, 1, 2, \dots$ **faça**
 - 4: $y \leftarrow x^{(k)}$
 - 5: **Para** $i = 1, 2, \dots, 2^k$ **faça** ▷ *loop* para computar $U_0^{2^k} x^{(k)}$
 - 6: $y \leftarrow y - \beta A^T (Ay)$
 - 7: **Fim Para**
 - 8: $x^{(k+1)} \leftarrow x^{(k)} + y$
 - 9: **Se** (critério de parada satisfeito) **então**
 - 10: $x^{(*)} \leftarrow x^{(k+1)}$
 - 11: **Fim Se**
 - 12: **Fim Para**
-

(e.g., quando as matrizes envolvidas são esparsas), o Algoritmo 3.3 pode ser vantajoso. Além disso, caso a matriz A não esteja disponível explicitamente, construir (e armazenar) U_0 se torna uma tarefa impraticável, levando ao uso do segundo algoritmo somente. Nesta situação, A só pode ser acessada via produtos da forma Az ou $A^T z$; para problemas de grande porte, esta é uma realidade, acentuada com o uso de computação paralela.

Uma vantagem do método de Newton é o fato de lidarmos, em sua totalidade, somente com operações “brutas”, aqui expressas por

cálculos envolvendo matrizes e vetores, menos sujeitas a instabilidades numéricas pela aritmética finita. Para o algoritmo LSQR, por exemplo, a ortogonalidade dos vetores construídos em cada iteração é frequentemente perdida devido a esta dificuldade; neste caso, algum processo de reortogonalização precisa ser acoplado ao método para garantir sua completa funcionalidade. No caso do método iterativo proposto aqui, cuidados desta ordem não precisam ser tomados.

4 ITERAÇÕES DE NEWTON REGULARIZADAS

Dado um sistema de equações lineares na forma $Ax = b$, verificamos que a sequência vetorial dada em (3.17) converge para a solução procurada $x_{LS} = A^\dagger b$. Na prática, no entanto, o vetor de dados b normalmente não é exato, sendo dado por um vetor com perturbações $\tilde{b} = b + e$, para e vetor de imprecisões desconhecido. Como elaborado no Capítulo 2, por estarmos lidando com problemas sensíveis à perturbações, pequenas variações nos dados de entrada levam a grandes diferenças na solução computada. Deste modo, ao aplicarmos a sequência (3.17) em $Ax = \tilde{b}$, não temos garantia que a recorrência

$$\tilde{x}^{(k)} = X_k \tilde{b}$$

convirja para a solução de interesse $x_{LS} = A^\dagger b$, especialmente considerando A mal condicionada. Portanto, em poucas iterações, as imprecisões em \tilde{b} tendem a deturpar completamente a sequência, esta convergindo para $\tilde{x}_{LS} = A^\dagger \tilde{b}$, que em geral não serve de aproximação para x_{LS} , por estar dominada pelo ruído contido nos dados de entrada.

Com efeito, de (3.16), temos que

$$\omega_i^{(k)} = \frac{1 - \left(1 - \sigma_i \omega_i^{(0)}\right)^{2^k}}{\sigma_i},$$

implicando que podemos escrever os iterados $\tilde{x}^{(k)}$ diretamente, a partir de (3.18), como

$$\begin{aligned} \tilde{x}^{(k)} &= \sum_{i=1}^r \left[1 - \left(1 - \sigma_i \omega_i^{(0)}\right)^{2^k} \right] \frac{u_i^T \tilde{b}}{\sigma_i} v_i \\ &= x^{(k)} + \sum_{i=1}^r \left[1 - \left(1 - \sigma_i \omega_i^{(0)}\right)^{2^k} \right] \frac{u_i^T e}{\sigma_i} v_i, \end{aligned}$$

em que os números $f_i = 1 - \left(1 - \sigma_i \omega_i^{(0)}\right)^{2^k}$ são os *fatores de filtro* para o método iterativo baseado na iteração de Newton. Veja que, para k pequeno, os fatores de filtro $f_i \approx 0$ e, portanto, $\tilde{x}^{(k)}$ tende a se aproximar de $x^{(k)}$. Por outro lado, quando $f_i \approx 1$, algo que ocorre relativamente rápido considerando o crescimento exponencial de 2^k ,

então o k -ésimo iterado começa a ser dominado pelo ruído, momento que surge a propriedade de semi-convergência [31]. Em outros termos, quando estamos lidando com problemas com o lado direito perturbado, o índice k atua como parâmetro de regularização. Seu objetivo é escolher alguma solução de qualidade para o problema original $Ax = b$ através, somente, dos iterados $\tilde{x}^{(k)}$, que são gerados a partir de dados com ruído.

Uma maneira de visualizar o papel da semi-convergência e da boa escolha de k é analisando o comportamento de $\|x_{LS} - \tilde{x}^{(k)}\|_2$ como uma função de k . Note que esta cota pode ser dividida em duas partes:

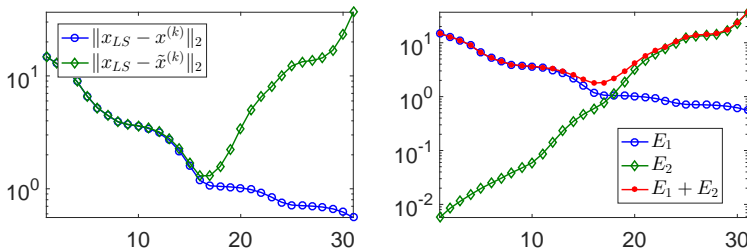
$$\|x_{LS} - \tilde{x}^{(k)}\|_2 \leq \|x_{LS} - x^{(k)}\|_2 + \|x^{(k)} - \tilde{x}^{(k)}\|_2, \quad (4.1)$$

em que a primeira é denominada *erro de regularização* e a segunda é o *erro de ruído* (ou *erro de amplificação*). Pela convergência do método, o erro de regularização decresce com k , como resultado do Teorema 3.8. Por outro lado, o erro de ruído pode ser escrito como

$$\|x^{(k)} - \tilde{x}^{(k)}\|_2^2 = \sum_{i=1}^r \left[\frac{1 - \left(1 - \sigma_i \omega_i^{(0)}\right)^{2k}}{\sigma_i} \right]^2 |u_i^T e|^2,$$

que claramente cresce com k : basta perceber que a quantidade em colchetes é crescente com k , para σ_i fixo. O comportamento típico de ambos os erros e também de $\|x_{LS} - \tilde{x}^{(k)}\|_2$ quando comparado com $\|x_{LS} - x^{(k)}\|_2$ pode ser visto na Figura 4.1 (direita).

Figura 4.1 – Comportamento dos erros com e sem perturbação (esquerda) e da estimativa (4.1) (direita), em que E_1 e E_2 representam os erros de regularização e de ruído, respectivamente. Problema shaw, dimensão 500, com perturbação de 1% nos dados de entrada.



Fonte: o autor, 2017.

Informações importantes podem ser extraídas da Figura 4.1. A figura da esquerda é clara em expressar que, quando temos ruído nos dados, os iterados $\tilde{x}^{(k)}$ tendem a se distanciar de x_{LS} a partir de alguma certa iteração k ; para os iterados anteriores a este índice, no entanto, o erro gerado pela sequência com imprecisões e pela exata são basicamente os mesmos, sugerindo que os iterados computados $\tilde{x}^{(k)}$ e $x^{(k)}$ estão próximos também. Já a figura da direita informa como se dá a introdução do ruído no processo iterativo. Caso paremos as iterações muito cedo, isto é, para k pequeno, apesar de termos pouca influência do ruído, pouca informação do problema em si é introduzida nos iterados. Por outro lado, se escolhermos k grande, o ruído corrompe totalmente a solução capturada. Portanto, precisamos encontrar um equilíbrio entre as duas quantidades, de modo a minimizar a influência do ruído ao mesmo tempo que componentes importantes da solução exata são capturados. Este “balanço” é representado pela propriedade de semi-convergência e, intuitivamente, infere que deveríamos procurar um k nas proximidades do minimizador da curva discreta $E_1 + E_2$, ponto este que equilibra o efeito do ruído com a quantidade de informações fornecidas pelo problema original. É neste estágio que entram os critérios de parada discutidos na Seção 2.4, cujo objetivo é tentar estimar o parâmetro de regularização k , o que normalmente não é tarefa fácil.

4.1 ESTIMATIVA TEÓRICA

Quando o princípio da discrepância de Morozov (DP) [29] é utilizado como critério de parada, então é possível deduzir uma estimativa teórica para $\|\tilde{x}^{(k)} - x_{LS}\|_2$, considerando que temos conhecimento do erro adicionado ao lado direito do problema e que o mesmo satisfaz

$$\|\tilde{b} - b\|_2 = \|e\|_2 \leq \tau\delta,$$

para $\tau \gtrsim 1$ e $\delta = \|e\|_2$ fixos. Como sabemos, DP escolhe o parâmetro de regularização pedindo que a norma do resíduo $\|\tilde{r}^{(k)}\|_2$ seja da mesma ordem que a discrepância nos dados de entrada. Matematicamente, isto significa que devemos tomar como parâmetro de regularização o primeiro inteiro $k = k(\delta)$ tal que

$$\|\tilde{r}^{(k)}\|_2 = \|\tilde{b} - A\tilde{x}^{(k)}\|_2 \leq \tau\delta \leq \|\tilde{b} - A\tilde{x}^{(k-1)}\|_2 = \|\tilde{r}^{(k-1)}\|_2. \quad (4.2)$$

O teorema a seguir apresenta a estimativa propriamente dita, lembrando que a desigualdade acima é uma hipótese essencial. Para critérios heurísticos, tal estimativa não é possível, devido à escolha do parâmetro

de regularização ocorrer de maneira mais volátil, i.e., baseada no comportamento típico dos iterados e não em aspectos teóricos mais rígidos (como para DP).

Teorema 4.1. *Suponha que $b \in \mathcal{R}(A)$ e denote o iterado gerado pelo princípio da discrepância por $\tilde{x}^{(k(\delta))}$. Então, é válida a estimativa*

$$\frac{\|\tilde{x}^{(k(\delta))} - x_{LS}\|_2}{\|x_{LS}\|_2} = \mathcal{O}\left(\frac{\delta^{1/2}}{\|b\|_2^{1/2}}\right).$$

Demonstração. Para $k \geq 1$ qualquer, note que, como $\tilde{b} = b + e$,

$$\tilde{x}^{(k)} - x_{LS} = X_k \tilde{b} - A^\dagger b = X_k b - A^\dagger b + X_k e.$$

Portanto,

$$\|\tilde{x}^{(k)} - x_{LS}\|_2 \leq \|X_k b - A^\dagger b\|_2 + \|X_k e\|_2. \quad (4.3)$$

Cada uma das normas acima serão estimadas separadamente. Para o erro de regularização $\|X_k b - A^\dagger b\|_2$, lembre que o Lema 3.6, item (iii), e a monotonia dos resíduos presente no Teorema 3.9, item (ii), implicam que

$$\begin{aligned} \|X_k b - A^\dagger b\|_2 &= \|A^\dagger A X_k b - A^\dagger b\|_2 \\ &\leq \|A^\dagger\|_2 \|A X_k b - b\|_2 \\ &= \|A^\dagger\|_2 \|r^{(k)}\|_2 \\ &\leq \|A^\dagger\|_2 \|r^{(k-1)}\|_2. \end{aligned} \quad (4.4)$$

Agora, como $(I - A X_k) = (I - A X_{k-1})^2$, temos:

$$\begin{aligned} \|r^{(k-1)}\|_2^2 &= \|b - A x^{(k-1)}\|_2^2 \\ &= b^T (I - A X_{k-1})^T (I - A X_{k-1}) b \\ &= b^T (I - A X_k) b \\ &= \langle b, r^{(k)} \rangle \\ &\leq \|b\|_2 \|r^{(k)}\|_2, \end{aligned}$$

o que implica que

$$\begin{aligned} \|r^{(k-1)}\|_2 &\leq \|b\|_2^{1/2} \|r^{(k)}\|_2^{1/2} \\ &= \|b\|_2^{1/2} \|(I - A X_k)(\tilde{b} - e)\|_2^{1/2} \\ &\leq \|b\|_2^{1/2} \left(\|\tilde{r}^{(k)}\|_2 + \|I - A X_k\|_2 \|e\|_2 \right)^{1/2}. \end{aligned} \quad (4.5)$$

Por outro lado, considerando a SVD de A , segue que

$$\|I - AX_0\|_2 = \|I - \beta U \Sigma \Sigma^T U^T\|_2 = \|I - \beta \Sigma \Sigma^T\|_2 = \max_{1 \leq i \leq r} |1 - \beta \sigma_i^2|.$$

Como $0 < \beta < 2/\sigma_1^2$, temos que, para $i \in \{1, \dots, r\}$,

$$0 < \beta \sigma_i^2 < \frac{2}{\sigma_1^2} \sigma_i^2 \leq 2 \Rightarrow 0 > -\beta \sigma_i^2 > -2 \Rightarrow -1 < 1 - \beta \sigma_i^2 < 1,$$

o que implica que $\|I - AX_0\|_2 < 1$. Disto, é válido que

$$\|I - AX_k\|_2 \leq \|I - AX_{k-1}\|_2^2 \leq \dots \leq \|I - AX_0\|_2^{2^k} < 1.$$

Logo, tomando $k = k(\delta)$ daqui em diante, e em vista da primeira desigualdade em (4.2) (i.e., $\|\tilde{r}^{(k)}\|_2 \leq \tau \delta$), a desigualdade (4.5) se torna

$$\|r^{(k-1)}\|_2 \leq \|b\|_2^{1/2} (\tau \delta + \delta)^{1/2} = \|b\|_2^{1/2} (\tau + 1)^{1/2} \delta^{1/2}. \quad (4.6)$$

Portanto, o erro de regularização (4.4) fica estimado por

$$\|X_k b - A^\dagger b\|_2 \leq \|A^\dagger\|_2 \|b\|_2^{1/2} (\tau + 1)^{1/2} \delta^{1/2}. \quad (4.7)$$

Podemos partir então para a segunda parte da estimativa, envolvendo $\|X_k e\|_2$, normalmente chamado de erro de ruído ou de amplificação. Para tanto, vamos extrair uma relação para $\|e\|_2$ a partir da segunda desigualdade em (4.2), a saber:

$$\begin{aligned} \tau \delta &\leq \|\tilde{r}^{(k-1)}\|_2 \\ &= \|(I - AX_{k-1})(b + e)\|_2 \\ &\leq \|r^{(k-1)}\|_2 + \|I - AX_{k-1}\|_2 \|e\|_2 \\ &\leq \|r^{(k-1)}\|_2 + \delta, \end{aligned}$$

o que implica que $\|e\|_2 = \delta \leq \|r^{(k-1)}\|_2 / (\tau - 1)$. Portanto, pelo Lema 3.6, item (iv), e por (4.6), o erro de amplificação é estimado por

$$\|X_k e\|_2 \leq \|X_k\|_2 \|e\|_2 \leq \|A^\dagger\|_2 \frac{\|b\|_2^{1/2} (\tau + 1)^{1/2} \delta^{1/2}}{\tau - 1}. \quad (4.8)$$

Finalmente, juntando (4.7) e (4.8) em (4.3), segue que

$$\|\tilde{x}^{(k(\delta))} - x_{LS}\|_2 \leq C_\tau \|A^\dagger\|_2 \|b\|_2^{1/2} \delta^{1/2},$$

em que $C_\tau = \tau(1 + \tau)^{1/2}/(\tau - 1)$. Lembrando que $b \in \mathcal{R}(A)$, temos que $b = Ax_{LS}$ e, portanto, $\|b\|_2 \leq \|A\|_2 \|x_{LS}\|_2$. Assim,

$$\|\tilde{x}^{(k(\delta))} - x_{LS}\|_2 \leq C_\tau \|A^\dagger\|_2 \|A\|_2 \|x_{LS}\|_2 \frac{\delta^{1/2}}{\|b\|_2^{1/2}},$$

que pode ser reescrito como

$$\frac{\|\tilde{x}^{(k(\delta))} - x_{LS}\|_2}{\|x_{LS}\|_2} \leq C_\tau \|A\|_2 \|A^\dagger\|_2 \frac{\delta^{1/2}}{\|b\|_2^{1/2}} = \mathcal{O}\left(\frac{\delta^{1/2}}{\|b\|_2^{1/2}}\right),$$

completando a demonstração. \square

Em palavras, a estimativa infere que, quanto menor a perturbação nos dados de entrada, melhor será a aproximação de $x_{LS} = A^\dagger b$ gerada pela sequência $\tilde{x}^{(k)} = X_k \tilde{b}$ quando o princípio da discrepância é utilizado como critério de parada.

Vale observar, no entanto, que a constante $C_\tau \|A\|_2 \|A^\dagger\|_2$ acompanha a cota superior e pode atingir valores de alta ordem, já que depende de

$$\kappa(A) = \|A\|_2 \|A^\dagger\|_2,$$

além, é claro, de depender de τ , apesar de sua influência poder ser melhor controlada, já que τ muito próximo de 1 em geral não é utilizado. Mesmo assim, a Tabela 4.1 apresenta algumas das possibilidades para C_τ . Esta constante é, na realidade, um problema secundário e negligenciável se comparado com o número de condição $\kappa(A)$ de problemas práticos (veja, por exemplo, a Tabela 4.2 abaixo).

Tabela 4.1 – Alguns valores da constante C_τ .

τ	1.1	1.05	1.01
C_τ	15.9405	30.0674	143.1922

Fonte: o autor, 2017.

4.2 RESULTADOS NUMÉRICOS

A seguir, conduzimos alguns testes numéricos para o método de Newton (3.23). Estamos considerando resolver sistemas da forma $Ax = b$ em que o lado direito é, na realidade, dado por $\tilde{b} = b + e$, para e vetor com imprecisões. Desta maneira, queremos aproximar a solução

$x_{LS} = A^\dagger b$ através da sequência (3.23) gerada com dados perturbados, isto é, construída a partir do sistema $Ax = \tilde{b}$. Como explanado anteriormente, a sequência gerada desta maneira apresenta propriedade de semi-convergência; para detectá-la, fazemos uso de três critérios de parada: DP, L-curve e MPR, discutidos na Seção 2.4. Os problemas-teste utilizados estão listados na Tabela 4.2, bem como seu posto numérico e número de condição. Ressaltamos que os primeiros sete problemas da tabela são provenientes da toolbox *Regularization Tools* [21] enquanto que os restantes vêm da Galeria do MATLAB (*MATLAB's Gallery*).

Tabela 4.2 – Posto numérico e número de condição κ dos problemas-teste (via rotinas `rank` e `cond` do MATLAB), dimensão $n = 1000$.

Problema	Posto	κ
foxgood	30	2.8795×10^{20}
phillips	1000	2.6415×10^{10}
heat	588	2.5979×10^{232}
shaw	20	1.1476×10^{21}
gravity	45	4.7467×10^{20}
baart	13	3.6112×10^{18}
deriv2	1000	1.2159×10^6
moler	999	5.0406×10^{15}
lotkin	22	9.644×10^{21}
prolate	521	4.727×10^{17}
lehmer	1000	1.0748×10^6
cauchy	23	1.4338×10^{21}
fiedler	1000	6.9481×10^5
frank	999	1.6599×10^{19}
hilb	24	2.5685×10^{21}

Fonte: o autor, 2017.

Os problemas provenientes de *Regularization Tools* geram automaticamente A e b , além da solução exata x_{LS} . Já a Galeria do MATLAB apenas constrói a matriz A ; para estes, tomamos x_{LS} como a solução gerada para o problema `shaw` e, portanto, $b = Ax_{LS}$. Como são exemplos acadêmicos, construímos o vetor de perturbações e através da função `randn` do MATLAB de modo que

$$\|e\|_2 = \|\tilde{b} - b\|_2 = NL\|b\|_2,$$

para NL simbolizando o chamado *nível de ruído*, que representa o erro

Tabela 4.3 – Resultados numéricos para $NL = 0.025$, dimensão $n = 1000$.

		foxgood	phillips	heat	shaw	gravity
$k_m(k_M)$	LSQR	2(2)	4(5)	8(9)	4(4)	4(4)
	DP	8(8)	8(8)	11(12)	8(9)	7(8)
	L-curve	9(9)	10(14)	13(13)	9(13)	11(13)
	MPR	9(9)	10(10)	13(13)	10(12)	10(11)
E_m	LSQR	0.0314	0.0264	0.1785	0.1681	0.0683
	DP	0.0583	0.0473	0.1858	0.1664	0.0708
	L-curve	0.0320	0.0298	0.1127	0.1586	0.0371
	MPR	0.0320	0.0247	0.1127	0.1511	0.0368
		baart	deriv2	moler	lotkin	prolate
$k_m(k_M)$	LSQR	2(3)	4(4)	2(3)	2(2)	1(1)
	DP	8(11)	10(10)	8(11)	9(9)	12(12)
	L-curve	12(12)	11(12)	11(11)	10(10)	13(21)
	MPR	12(12)	11(11)	11(11)	10(11)	13(13)
E_m	LSQR	0.1794	0.3167	0.3333	0.4745	0.0176
	DP	0.3002	0.3163	0.4001	0.4843	0.0239
	L-curve	0.2026	0.2896	0.3418	0.4734	0.0270
	MPR	0.2026	0.2915	0.3418	0.4731	0.0177
		lehmer	cauchy	fiedler	frank	hilb
$k_m(k_M)$	LSQR	4(4)	4(4)	4(5)	5(5)	5(5)
	DP	10(11)	11(11)	11(11)	9(9)	11(12)
	L-curve	11(11)	11(14)	11(12)	12(13)	13(13)
	MPR	11(11)	12(13)	11(12)	12(12)	13(13)
E_m	LSQR	0.2423	0.4646	0.1260	0.1135	0.4469
	DP	0.2333	0.4622	0.1942	0.1058	0.4627
	L-curve	0.2199	0.4598	0.1886	0.1659	0.4472
	MPR	0.2199	0.4506	0.1913	0.1271	0.4472

Fonte: o autor, 2017.

relativo entre \tilde{b} e b com relação a b . Nos testes numéricos, tomamos $NL = 0.025, 0.01$ e 0.001 , inferindo que \tilde{b} possui, respectivamente, 2.5%, 1% e 0.1% de perturbação em relação ao vetor original b . As Tabelas 4.3, 4.4 e 4.5 contêm os resultados para estes três níveis de ruído, considerando o método de Newton (3.23) através da implementação do Algoritmo 3.2 com parada dada por DP (com $\tau = 1.05$), L-curve (com número máximo de 35 iterações) e MPR. O parâmetro β necessário às iterações de Newton é tomado como $\beta = 1/\|A\|_F^2$. Notamos que esta

escolha de β infere a convergência dos recíprocos associados aos maiores valores singulares de A antes, isto é, $\omega_i^{(k)}$ converge para $1/\sigma_i$ preservando a ordem natural dos valores singulares, resultado garantido pelo Teorema 3.7. Tomamos, também, as soluções geradas pelo algoritmo LSQR (com parada dada por DP, $\tau = 1.05$) como critério comparativo para as soluções geradas pelo método de Newton. Ressaltamos que fazemos uso da implementação de LSQR contida na toolbox *Regularization Tools*, dada pela função `lsqr_b`.

Tabela 4.4 – Resultados numéricos para $NL = 0.01$, dimensão $n = 1000$.

		foxgood	phillips	heat	shaw	gravity
$k_m(k_M)$	LSQR	2(2)	4(5)	11(11)	4(5)	5(5)
	DP	9(9)	9(9)	13(13)	9(10)	9(9)
	L-curve	10(14)	16(16)	14(17)	14(15)	13(16)
	MPR	10(10)	10(10)	14(15)	14(14)	11(13)
E_m	LSQR	0.0310	0.0247	0.0987	0.1611	0.0467
	DP	0.0318	0.0291	0.1046	0.1621	0.0448
	L-curve	0.0242	0.0453	0.0700	0.0785	0.0321
	MPR	0.0299	0.0239	0.0651	0.0977	0.0297
		baart	deriv2	moler	lotkin	prolate
$k_m(k_M)$	LSQR	3(3)	5(6)	4(4)	2(3)	1(1)
	DP	13(13)	12(12)	13(13)	11(11)	13(13)
	L-curve	13(13)	13(15)	13(14)	13(13)	22(24)
	MPR	13(13)	13(14)	13(14)	13(13)	13(13)
E_m	LSQR	0.1663	0.2626	0.1894	0.4658	0.0072
	DP	0.1723	0.2657	0.2002	0.4659	0.0072
	L-curve	0.1723	0.2299	0.1975	0.4536	0.0260
	MPR	0.1723	0.2289	0.1975	0.4536	0.0072
		lehmer	cauchy	fiedler	frank	hilb
$k_m(k_M)$	LSQR	6(6)	5(5)	5(6)	7(7)	5(5)
	DP	12(13)	12(13)	12(13)	10(10)	13(13)
	L-curve	13(13)	14(15)	13(13)	15(15)	16(16)
	MPR	13(13)	14(14)	13(14)	15(15)	14(15)
E_m	LSQR	0.1310	0.4420	0.1131	0.0728	0.4466
	DP	0.1315	0.4477	0.0838	0.0770	0.4476
	L-curve	0.0806	0.4420	0.0647	0.2383	0.4405
	MPR	0.0806	0.4426	0.0634	0.2383	0.4438

Fonte: o autor, 2017.

Tabela 4.5 – Resultados numéricos para $NL = 0.001$, dimensão $n = 1000$.

		foxgood	phillips	heat	shaw	gravity
$k_m(k_M)$	LSQR	2(3)	7(8)	16(17)	7(7)	7(7)
	DP	10(11)	13(13)	16(17)	16(16)	13(13)
	L-curve	17(22)	22(23)	23(23)	18(22)	20(22)
	MPR	10(16)	11(14)	17(18)	17(17)	13(14)
E_m	LSQR	0.0304	0.0101	0.0360	0.0476	0.0223
	DP	0.0301	0.0158	0.0350	0.0532	0.0198
	L-curve	0.0140	0.0632	0.0739	0.0474	0.0274
	MPR	0.0295	0.0116	0.0295	0.0478	0.0186
		baart	deriv2	moler	lotkin	prolate
$k_m(k_M)$	LSQR	3(3)	11(11)	6(6)	3(4)	1(1)
	DP	14(15)	17(18)	16(16)	14(14)	13(13)
	L-curve	21(21)	20(22)	20(21)	19(21)	29(31)
	MPR	15(15)	18(19)	18(18)	17(19)	14(14)
E_m	LSQR	0.1659	0.1686	0.0411	0.4516	0.0016
	DP	0.1645	0.1714	0.0412	0.4514	0.0010
	L-curve	0.1154	0.1501	0.0654	0.4462	0.0158
	MPR	0.1633	0.1483	0.0231	0.4490	0.0009
		lehmer	cauchy	fiedler	frank	hilb
$k_m(k_M)$	LSQR	9(9)	6(6)	7(7)	11(12)	6(6)
	DP	15(15)	16(16)	15(15)	13(13)	17(17)
	L-curve	21(21)	19(23)	22(22)	-(-)	21(23)
	MPR	17(21)	17(17)	16(16)	-(-)	17(17)
E_m	LSQR	0.0278	0.4394	0.0161	0.0170	0.4396
	DP	0.0223	0.4402	0.0215	0.0188	0.4396
	L-curve	0.1024	0.4395	0.0979	-	0.4393
	MPR	0.0698	0.4396	0.0107	-	0.4396

Fonte: o autor, 2017.

Repetimos a resolução de cada problema com o método de Newton e com LSQR um total de 30 vezes, gerando erros relativos e iterações de parada para cada uma destas resoluções. Portanto, nas Tabelas 4.3, 4.4 e 4.5, as quantidades k_m e k_M representam, respectivamente, a iteração mínima e máxima de parada, de todas as repetições. Analogamente, E_m denota o erro relativo médio, que consiste da média aritmética dos erros relativos gerados em cada resolução. Para melhor interpretação dos resultados obtidos, os erros relativos de melhor qualidade aparecem

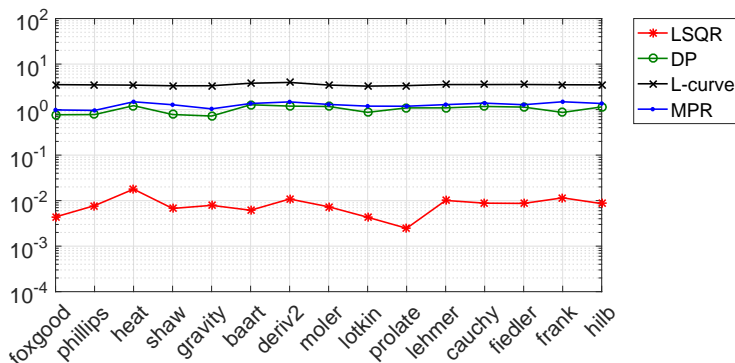
em negrito. Observamos que, caso sejam geradas soluções com erros relativos superiores a 50%, consideramos que o método não obteve sucesso e, deste modo, a solução é descartada.

Através dos resultados, exceto pelo fato dos erros associados aos problemas *lotkin*, *cauchy* e *hilb* serem grandes, em geral, os erros relativos associados com o método de Newton com parada via DP, L-curve e MPR são relativamente próximos dos valores gerados por LSQR, em muitos problemas até superiores em qualidade, informação facilmente verificada através de uma inspeção às Tabelas 4.3, 4.4 e 4.5. Notamos que MPR possui a vantagem, seguido de L-curve e, depois, DP (caso pensamos somente nas soluções geradas pelo método de Newton).

Ressaltamos que L-curve e MPR não obtiveram sucesso na resolução do problema *frank* (veja Tabela 4.5), gerando resultados com erro relativo superior a 50%. Esta situação confirma o fato bem conhecido de que as regras heurísticas deveriam falhar, ao menos para alguns problemas, um resultado de Bakushinski [3].

É também interessante a constatação de que o número de passos k decresce quando o nível de ruído cresce. Isto está de acordo com o efeito regularizante de métodos iterativos, o qual implica que quanto maior o nível de ruído mais valores singulares são comprometidos e, portanto, a iteração deve ser parada antes (veja, por exemplo, Hanke e Hansen [18]).

Figura 4.2 – Tempo médio, em segundos, para computar os dados da Tabela 4.4.



Fonte: o autor, 2018.

Os resultados sugerem que o método de Newton possui funcionalidade

dade prática (ao menos para os problemas de baixo porte considerados), apresentando soluções de qualidade, em geral, quando associado com critérios de parada como os utilizados aqui. No entanto, como evidencia a Figura 4.2, a eficiência do método quando comparado com LSQR é um empecilho: mesmo para os problemas pequenos do exemplo, Newton necessita em torno de duas ordens de grandeza mais tempo que LSQR. A situação é agravada para L-curve, que demanda o cálculo prévio de um grande número de iterações. Este cenário é um reflexo da necessidade de lidar com produtos envolvendo a matriz $U_0 \in \mathbb{R}^{n \times n}$, cujo custo cresce com n . O custo operacional do algoritmo proposto através do método de Newton é o foco do próximo capítulo, para o qual buscamos lidar com algumas possíveis soluções.

5 ESTRATÉGIAS DE ACELERAÇÃO

Nos capítulos anteriores apresentamos e descrevemos o método de Newton, tanto sua versão matricial quanto a aplicada ao problema de mínimos quadrados associado a $Ax = b$, que denominamos método de Newton vetorial. Sua taxa de convergência e resultados teóricos reforçam o interesse no estudo e aplicação do mesmo. No entanto, o custo numérico para uma aplicação direta é certamente um dos maiores empecilhos à sua utilização, evidenciada pela necessidade de operar com a matriz $U_0 \in \mathbb{R}^{n \times n}$ na formulação (3.23), dada por

$$x^{(k+1)} = x^{(k)} + U_0^{2^k} x^{(k)}, \quad k = 0, 1, \dots$$

Tendo isto em mente, neste capítulo buscamos abordar algumas possíveis técnicas de aceleração do algoritmo, visando contornar esta dificuldade.

Essencialmente, o capítulo está dividido em duas partes: a primeira seção apresenta algumas modificações à iteração matricial de Newton (3.5), como tentativa de acelerar o processo; a segunda seção lida com uma estratégia de projeção do problema original em algum subespaço, visando reduzir a dimensão da matriz U_0 na aplicação subsequente do método de Newton ao problema projetado.

5.1 VARIAÇÕES AO MÉTODO DE NEWTON

Esta seção está baseada no trabalho de Pan e Schreiber [35], que fizeram avanços interessantes no que diz respeito à aceleração das iterações do método de Newton. Seu foco é voltado para a construção das matrizes X_k de modo que convirjam para A^\dagger mais rapidamente que Newton usual, focando também no ganho em operações.

Perceba que as iterações de Newton usual (3.5) são, basicamente, a aplicação em um polinômio quadrático em X_k :

$$X_{k+1} = 2X_k - X_k A X_k.$$

De maneira grosseira, as estratégias desta seção consistem em tomar variações no polinômio a ser avaliado, gerando melhor aproveitamento nas iterações respectivas.

Observação 5.1. Os exemplos apresentados nesta seção são voltados para a resolução do problema de mínimos quadrados associado a $Ax = b$

(ou $Ax = \tilde{b} = b + e$). Portanto, por vias de notação, as aproximações para a solução deste problema são dadas por

$$x^{(k)} = X_k b \quad (\text{ou } \tilde{x}^{(k)} = X_k \tilde{b}),$$

para X_k sequência do método respectivo apresentado, devidamente referenciado no momento apropriado. As operações são brutas, ou seja, construímos X_k para então calcular $x^{(k)}$ (ou $\tilde{x}^{(k)}$).

5.1.1 Aceleração via *scaling*

A ideia aqui é considerar o método de Newton matricial (3.5), mas com um parâmetro escalar adicional, atualizado em cada iteração. Os iterados são definidos considerando $X_0 = \alpha_0 A^T$ e

$$X_{k+1} = \alpha_{k+1}(2I - X_k A)X_k, \quad (5.1)$$

em que α_{k+1} é escolhido de modo a minimizar a distância do p -ésimo valor singular (diferente de zero) de $X_{k+1}A$ de 1. A maneira de efetuar tal escolha será explicitada a seguir. Suponha que conhecemos limitantes σ_{min} e σ_{max} para os valores singulares positivos de A , ou seja, $\sigma_{min} \leq \sigma_r^2 \leq \sigma_1^2 \leq \sigma_{max}$, em que r é o posto de A e σ_r e σ_1 representam, respectivamente, o menor e o maior valores singulares positivos de A . Assim, tome

$$\alpha_0 = \frac{2}{\sigma_{min} + \sigma_{max}}, \quad (5.2)$$

$$\underline{\rho}^{(0)} = \alpha_0 \sigma_{min} = \frac{2\sigma_{min}}{\sigma_{min} + \sigma_{max}} \quad \text{e} \quad (5.3)$$

$$\bar{\rho}^{(0)} = \alpha_0 \sigma_{max} = \frac{2\sigma_{max}}{\sigma_{min} + \sigma_{max}}. \quad (5.4)$$

Note que a escolha de α_0 como acima em geral garante convergência do método de Newton usual, tomando a aproximação inicial como $X_0 = \alpha_0 A^T$, pois $0 < \alpha_0 < 2/\sigma_1^2$. Agora, para $k \geq 0$, efetuamos a atualização

$$\alpha_{k+1} = \bar{\rho}^{(k+1)} = \frac{2}{1 + (2 - \underline{\rho}^{(k)})\underline{\rho}^{(k)}} \quad \text{e} \quad (5.5)$$

$$\underline{\rho}^{(k+1)} = \alpha_{k+1}(2 - \underline{\rho}^{(k)})\underline{\rho}^{(k)}. \quad (5.6)$$

A construção acima garante as relações, para todo $1 \leq j \leq r$ e $k \geq 1$,

$$\underline{\rho}^{(k)} \leq \rho_j^{(k)} \leq \bar{\rho}^{(k)} \quad \text{e} \quad \underline{\rho}^{(k)} = 2 - \bar{\rho}^{(k)},$$

em que $\rho_j^{(k)}$ representa o j -ésimo valor singular de $X_k A$. Segundo Pan e Schreiber [35], exceto para algumas iterações antes da convergência, em geral temos $\rho^{(k)} \ll 1$, implicando que $\alpha_{k+1} \approx 2$. Portanto, segue que $\rho_r^{(k+1)} \approx 4\rho_r^{(k)}$. Deste modo,

$$-\log_4 \rho_r^{(0)} \approx \log_2 \left[(\kappa(A)^2 + 1)^{1/2} \right] = \log_2 (\kappa(A)) + O(1/\kappa(A)^2)$$

passos são suficientes para levar todos os valores singulares de $X_k A$ acima de $1/2$, o que é metade do necessário para a versão sem aceleração.

A seguir, apresentaremos um resultado acerca da otimalidade dos parâmetros de aceleração dispostos em (5.5). Para tanto, considere a matriz de resíduo inicial

$$E = I - X_0 A = I - \alpha_0 A^T A. \quad (5.7)$$

Ao iniciarmos com $X_0 = \alpha_0 A^T$, o método de Newton usual produz iterados X_k satisfazendo

$$X_k A = I - E^{2^k}, \quad (5.8)$$

fato facilmente verificável através de indução matemática. Para o caso de A ser não-singular, a escolha de α_0 por (5.2) garante que os autovalores de E pertencem ao intervalo $(-1, 1)$ e, portanto, $E^{2^k} \rightarrow \mathbf{0}$, o que implica que $X_k A \rightarrow I$, ambos com $k \rightarrow \infty$. Unindo as equações (5.7) e (5.8) com as iterações do método de Newton usual (3.5) (e um passo de indução), temos

$$X_k = (I + E + \dots + E^{2^k - 1}) \alpha_0 A^T.$$

Por outro lado, sabemos da expansão em série de Neumann que

$$(I - E)^{-1} = I + E + E^2 + \dots,$$

inferindo que o método de Newton está aproximando esta inversa em cada iteração através do truncamento da série. Veremos que a estratégia de aceleração proposta pelas iterações (5.1) está relacionada a uma melhor aproximação polinomial para $(I - E)^{-1}$. De fato, é equivalente a uma aproximação desta inversa usando polinômios de Tchebychev, cujas propriedades de aproximação são bem conhecidas na literatura.

Consideremos os polinômios de Tchebychev no intervalo $(-1, 1)$ dados por

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \quad \text{e} \\ T_{2^k}(x) &= \cos(2^k \cos^{-1}(x)), \end{aligned}$$

em que o último é polinômio de grau 2^k , e lembre da recorrência

$$T_{2^{k+1}}(x) = 2T_{2^k}^2(x) - 1. \quad (5.9)$$

Agora, defina o polinômio de Tchebychev escalado em $(\sigma_{min}, \sigma_{max})$ por

$$t_k(x) = \frac{T_{2^k}(\gamma x + \delta)}{T_{2^k}(\delta)}, \text{ em que } \gamma = \frac{-(\sigma_{max} + \sigma_{min})}{\sigma_{max} - \sigma_{min}} \text{ e } \delta = \frac{2}{\sigma_{max} - \sigma_{min}}.$$

Claramente, $t_k(x)$ tem grau 2^k e $t_k(0) = 1$, implicando que $t_k(x) = 1 - x\bar{t}_k(x)$, para algum polinômio $\bar{t}_k(x)$ de grau $2^k - 1$. Pela relação de recorrência (5.9) e tomando $\beta_k = T_{2^k}(\delta)$, segue que

$$\beta_k t_k(x) = 2(\beta_{k-1} t_{k-1}(x))^2 - 1.$$

Agora, vamos ao teorema propriamente dito.

Teorema 5.2. [35] *Seja a seqüência de matrizes X_k , para $k = 0, 1, 2, \dots$, gerada de acordo com a aproximação inicial dada por $X_0 = \alpha_0 A^T$ e as relações (5.1)–(5.6). Assim,*

$$X_k = \bar{p}_k(A^T A)A^T, \quad (5.10)$$

em que $\bar{p}_k(x)$ é um polinômio de grau $2^k - 1$. Mais ainda, $1 - x\bar{p}_k(x)$ é o polinômio de Tchebychev escalado $t_k(x)$ de grau 2^k no intervalo $(\sigma_{min}, \sigma_{max})$.

Observação 5.3. Note que, de (5.10), temos

$$I - X_k A = I - \bar{p}_k(A^T A)A^T A,$$

que implica

$$\|I - X_k A\|_2 = \max_{1 \leq j \leq n} |1 - \sigma_j^2 \bar{p}_k(\sigma_j^2)|,$$

que exhibe a relevância da conclusão do teorema.

Tendo em vista alguma implementação computacional do método dado em (5.1), é necessário conhecer σ_{min} e σ_{max} , ambos extremamente ligados à matriz A . O segundo pode ser tomado como $\sigma_{max} = \|A\|_F^2$ ou por alguns passos da bidiagonalização de Lanczos, satisfazendo a propriedade buscada. Por outro lado, σ_{min} não é tão facilmente estimado, principalmente para problemas mal postos, devido ao rápido decaimento dos valores singulares. Neste cenário, $\sigma_r \approx 0$ e pode ser frequentemente considerado nulo na aritmética finita. Necessariamente, precisamos que

$\sigma_{min} > 0$; em caso de igualdade, teremos $\alpha_{k+1} = 2$, para todo $k \geq 0$, o que impede convergência.

Comentários à parte, o algoritmo 5.1 apresenta uma possível implementação para o que denotaremos, de aqui por diante, método *NS* (*Newton Scaled*). Observe que, uma vez determinados σ_{min} e σ_{max} , as atualizações de α_k são efetuadas sem dificuldade.

Algoritmo 5.1 Método NS

Entrada: A

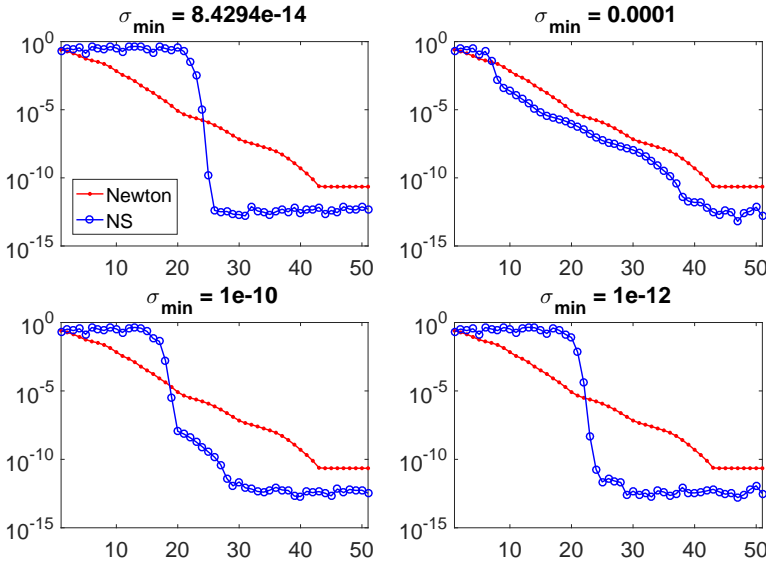
Saída: Aproximação para A^\dagger

- 1: Calcule σ_{min} e σ_{max} .
 - 2: $\alpha_0 \leftarrow \frac{2}{\sigma_{min} + \sigma_{max}}$
 - 3: $\rho^{(0)} \leftarrow \alpha_0 \sigma_{min}$
 - 4: $\bar{X}_0 \leftarrow \alpha_0 A^T$
 - 5: **Para** $k = 0, 1, 2, \dots$ **faça**
 - 6: $\alpha_{k+1} \leftarrow \frac{2}{1 + (2 - \rho^{(k)})\rho^{(k)}}$
 - 7: $\rho^{(k+1)} \leftarrow \alpha_{k+1} (2 - \rho^{(k)})\rho^{(k)}$
 - 8: $\bar{X}_{k+1} = \alpha_{k+1} (2I - \bar{X}_k A) \bar{X}_k$
 - 9: **Fim Para**
-

Todavia, algumas possibilidades interessantes são apresentadas nos exemplos das Figuras 5.1 e 5.2, embora alguns valores de σ_{min} não respeitem a propriedade procurada. De fato, apenas o primeiro gráfico (canto superior esquerdo) respeita explicitamente que $\sigma_{min} \leq \sigma_r^2$. Note que *shaw* (Figura 5.2) nem apresenta convergência de qualidade para este σ_{min} , possivelmente por ser muito pequeno e gerar dificuldades à aritmética computacional. Além disso, ao tomarmos valores maiores para σ_{min} , temos por resultado algo semelhante com o comportamento do método de Newton usual (por exemplo, com $\sigma_{min} = 0.0001$). Portanto, é preciso encontrar algum equilíbrio, para então conseguir bom aproveitamento do algoritmo, que pode atingir resultados com precisão superior ao Newton usual com menos iterações. Note, também, que não temos monotonia nos resíduos apresentados, o mesmo acontecendo com $\|x^{(k)}\|_2$, apesar de não explicitado aqui. Isto dificulta (de fato impossibilita) o uso de alguns critérios heurísticos de parada, tais como L-curve e MPR.

A despeito dos aspectos acima mencionados, deve ser observado que NS necessita de duas multiplicações de matrizes por iteração na forma apresentada em (5.1), o que é um produto a mais do que utilizado pela forma final de Newton (3.23) (caso explícito). Para alguns σ_{min} ,

Figura 5.1 – Resíduos $\|b - Ax^{(k)}\|_2$ Newton e NS para diferentes σ_{\min} , aplicados ao problema *heat*, dimensão 100, sem ruído.



Fonte: o autor, 2017.

entretanto, NS apresenta convergência mais rápida e de maior qualidade, reduzindo em 20 ou até 30 iterações em relação ao Newton.

5.1.2 Aceleração por polinômio cúbico

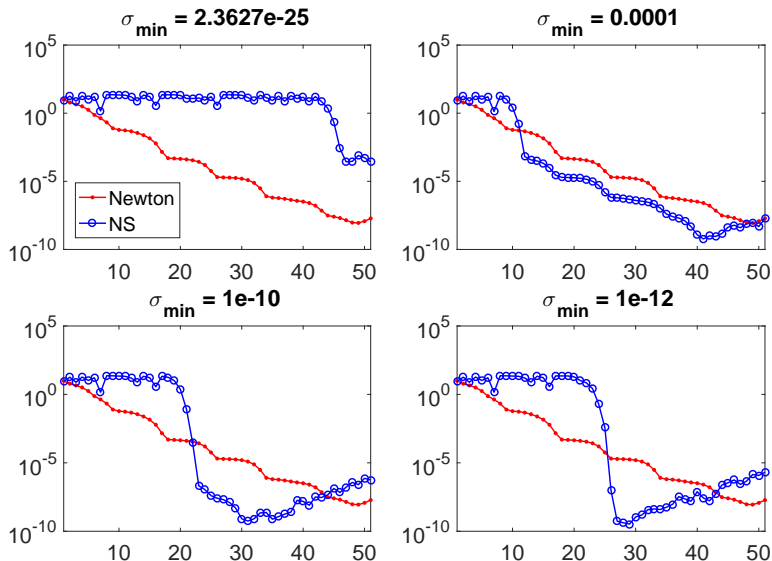
Esta estratégia é uma alternativa ao Newton escalado apresentado acima, quando algum *gap* é detectado no espectro singular de $X_k A$. Naturalmente, esperamos que alguns $\sigma_i(X_k A) \rightarrow 1$ enquanto que outros $\sigma_i(X_k A) \rightarrow 0$, com $k \rightarrow \infty$. Em alguns casos, é possível detectar *clusters* de valores singulares como, por exemplo, vários $\sigma_i(X_k A)$ próximos de 0 ao passo que os outros estão próximos de 1, gerando este *gap*. Aqui, veremos quando é possível usar esta propriedade em favor das iterações.

Tome X tal que $XA = VRV^T$, $R = \text{diag}(\rho_1, \dots, \rho_n)$, com $0 \leq \rho_j \leq 2$, para todo j , e V com colunas ortonormais. Buscamos uma aproximação X_1 de A^\dagger tal que

$$X_1 = (\gamma_3(XA)^2 + \gamma_2(XA) + \gamma_1 I)X. \quad (5.11)$$

Na prática, X pode ser pensada como uma aproximação X_k gerada por Newton e X_1 tomará o lugar do próximo iterado X_{k+1} . A essência

Figura 5.2 – Resíduos $\|b - Ax^{(k)}\|_2$ para Newton e NS para diferentes σ_{\min} , aplicados ao problema shaw, dimensão 100, sem ruído.



Fonte: o autor, 2017.

da estratégia (5.11) consiste em efetuar, ao invés do passo quadrático do Newton usual, um passo cúbico, em que as constantes γ_i buscam ressaltar (e aproveitar) o *gap* detectado no espectro de XA .

Escrevendo (5.11) na forma de polinômio, temos $c(x) = \gamma_1 x + \gamma_2 x^2 + \gamma_3 x^3$. Pedimos que $c(x)$ satisfaça as seguintes propriedades:

- (i) $c(1) = 1$;
- (ii) $c'(1) = 0$;
- (iii) $c(0) = 0$;
- (iv) $c'(0) \gg 2$.

A ideia central gira em torno de que, com este passo cúbico, os valores singulares pequenos de XA são amplificados por um fator $c'(0)$, enquanto os perto de 1 continuam a convergir. Deste modo, forçamos

aceleração na convergência dos valores singulares pequenos sem perder o comportamento conquistado para os outros valores singulares.

Iniciamos encontrando $\underline{\rho}$ tal que não há ρ_j em $(\underline{\rho}, 1 - \underline{\rho})$ ou em $(1 + \underline{\rho}, 2]$. Para tanto, tome $T = XA$, compute T^2 e calcule $\delta = \|T - T^2\|_F$. Segue de (veja [35, equação (3.3)])

$$\delta^2 = \sum_{j=1}^n (\rho_j(1 - \rho_j))^2 \quad (5.12)$$

que $\rho_j|1 - \rho_j| \leq \delta$, para todo $j \in \{1, \dots, n\}$. Se $\delta \geq 1/4$, tal informação não nos é útil. No entanto, se $0 < \delta < 1/4$, então podemos concluir que os autovalores ρ_j estão em dois intervalos, a saber: $[0, \underline{\rho}]$ e $[1 - \underline{\rho}, 1 + \bar{\rho}]$, em que

$$\underline{\rho} = \frac{1}{2} - \sqrt{\frac{1}{4} - \delta} \quad \text{e} \quad \bar{\rho} = -\frac{1}{2} + \sqrt{\frac{1}{4} + \delta}.$$

Agora, vamos calcular $c(x)$ tal que os critérios (i)-(iv) são satisfeitos, juntamente com

$$(v) \quad c : [0, \underline{\rho}] \rightarrow [0, 1];$$

$$(vi) \quad c : [1 - \underline{\rho}, 1 + \bar{\rho}] \rightarrow [1, 1 + \underline{\rho}].$$

Para determinar c , então, pedimos os itens (i)-(iii) junto com

$$(vii) \quad c(\underline{\rho}) = 1.$$

Assim, a única solução é dada por

$$c(x) = \frac{1}{\underline{\rho}}(x^2 - (2 + \underline{\rho})x + (1 + 2\underline{\rho}))x.$$

Teorema 5.4. [35] *Seja $0 \leq \underline{\rho} \leq 1/2$. Se $c(x)$ é o único polinômio satisfazendo (i), (ii), (iii) e (vii), então também são válidos os itens (v) e (vi).*

Assim, por vias de conclusão, note que se $\delta \geq 1/4$, então não podemos acelerar. Neste caso, tomamos

$$X_1 = (2I - T)X$$

e passamos à próxima iteração (isto é, estamos efetuando um passo de Newton padrão). Por outro lado, se $\delta < 1/4$, calculamos $\underline{\rho} = 1/2 - \sqrt{1/4 - \delta}$ e tomamos

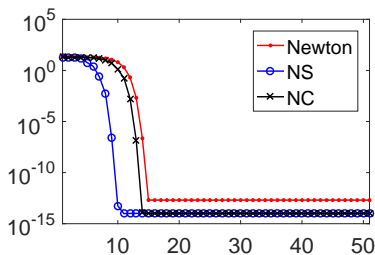
$$X_1 = \frac{1}{\underline{\rho}}(T^2 - (2 + \underline{\rho})T + (1 + 2\underline{\rho})I)X,$$

de modo que X_1 será resultado de um passo cúbico. Em última instância, pode ser pensado que o número δ determina a presença ou não de *gap* no espectro de XA , decidindo, assim, se o passo cúbico será ou não aplicado.

Para as implementações, Pan e Schreiber [35] afirmam que não podemos aplicar as iterações cúbicas indefinidamente: é necessário, para cada passo cúbico, a aplicação de ao menos um passo de Newton, o que garante que conseguimos encontrar intervalos $[0, \rho]$ e $[1 - \rho, 1 + \rho]$ que contém todos os autovalores. Além disso, quando estamos perto da convergência, iterações cúbicas não devem mais ser aplicadas, pois podem comprometer a convergência dos valores singulares. Computacionalmente, tal informação é verificada pelo cálculo de $\text{trace}(XA)$, que determina a aplicação de somente o método de Newton nas iterações seguintes.

O método descrito acima, que denotaremos por *Newton cúbico* (NC), possui uma implementação bastante detalhada e completa que pode ser encontrada em [35]. O custo por iteração aumenta em relação a NS: necessitamos de três multiplicações de matrizes. Apesar da necessidade do cálculo de T^2 para a verificação da aplicação ou não do passo cúbico, tal operação não é desperdiçada (caso a iteração cúbica seja rejeitada), pois o passo de Newton seguinte a aplica.

Figura 5.3 – Resíduos $\|b - Ax^{(k)}\|_2$ para Newton, NS (com $\sigma_{\min} = \sigma_r/2 = 2.0019$) e NC. Aqui, $A = \text{tridiag}(-1, 4, -1)$ e $b = Ax_{LS}$, com x_{LS} solução de shaw, dimensão 100. (Note que $\kappa(A) = 2.9981$.)

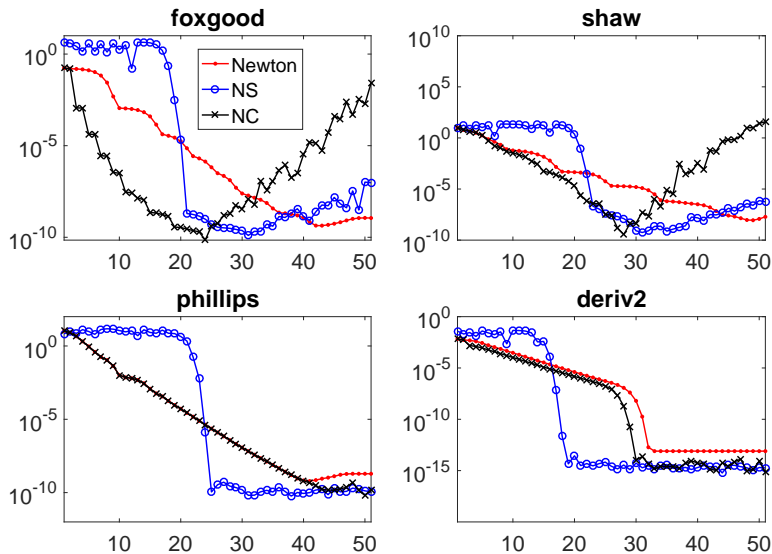


Fonte: o autor, 2017.

Nas Figuras 5.3 e 5.4 apresentamos alguns resultados de NC, comparando os mesmos com Newton e NS. Note que, em geral, NC aparenta possuir a propriedade de monotonia nas normas dos resíduos (diferentemente de NS), ao menos até alguma iteração, para em seguida

se comprometer. O mesmo aparenta acontecer com os plots de $\|x^{(k)}\|_2$, apesar de não representados aqui. Como NC atua acelerando os iterados de Newton, este comportamento se torna intuitivo.

Figura 5.4 – Resíduos $\|b - Ax^{(k)}\|_2$ para Newton, NS (com $\sigma_{\min} = 10^{-10}$) e NC, para alguns problemas, dimensão 100, sem ruído.



Fonte: o autor, 2017.

No entanto, perceba que seu funcionamento é inferior ao de NS, ao menos na maioria dos casos, em especial para matrizes melhor condicionadas, como na Figura 5.3. Um caso diferenciado ocorre para o problema *foxgood*, em que NC funciona bem. Por outro lado, para *phillips*, por exemplo, as iterações cúbicas são rejeitadas na maioria dos casos, sendo apenas aplicadas nas iterações finais. Ambos estão representados na Figura 5.4.

5.1.3 Suprimindo pequenos valores singulares

Considere a matriz A_ε , gerada a partir de A e um escalar $\varepsilon > 0$, de modo que tomamos como nulos todos os valores singulares de A que não excedem ε . A matriz A_ε é conhecida por ser a melhor aproximação de A por matrizes cujo posto é igual ao de A_ε . Esta modificação de A tem

efeito regularizante sobre a mesma, em especial se A for severamente mal condicionada, pelo fato de ignorar os valores singulares pequenos. A ideia desta subseção é gerar um método que calcule A_ε^\dagger .

Análise à parte, o método sugerido em [35] para o cálculo de A_ε^\dagger consiste em uma mistura entre as iterações apresentadas em (5.1) (basicamente tomando $\sigma_{min} = \varepsilon^2$) e o seguinte método:

$$X_{k+1/2} = (2I - X_k A) X_k, \quad X_{k+1} = X_{k+1/2} A X_{k+1/2}, \quad (5.13)$$

proposto por Schreiber [42]. O Algoritmo 5.2 abaixo descreve o que denotaremos por *método NS- ε* , com alguns comentários.

Algoritmo 5.2 Método NS- ε

- 1: Calcule σ_{max} limitante superior de σ_1^2 (e.g., tome $\sigma_{max} := \|A\|_F^2$ ou via Lanczos).
- 2: Tome

$$\alpha_0 \leftarrow \min \left\{ \frac{2}{\sigma_{max} + \varepsilon^2}, \frac{\hat{\rho}_1}{\varepsilon^2} \right\},$$

em que $\hat{\rho}_1 = (3 - \sqrt{5})/2$ (Observação 5.5). Em seguida, construa $\bar{\rho}^{(0)} \leftarrow \alpha_0 \sigma_{max}$ e $\underline{\rho}^{(0)} \leftarrow \alpha_0 \varepsilon^2$.

- 3: Aplique a iteração (5.1) atualizando os parâmetros requeridos de acordo com (5.5) e (5.6) até que $\underline{\rho}^{(k)} \geq \hat{\rho}_1$.
- 4: Na iteração k em que o passo 3 acima parar, tome

$$X_k \leftarrow \begin{pmatrix} \hat{\rho}_1 \\ \underline{\rho}^{(k)} \end{pmatrix} X_k.$$

Este passo garante que, de fato, os valores singulares menores que ε são realmente suprimidos.

- 5: Aplique a iteração (5.13) até que a matriz A_ε^\dagger esteja com a precisão desejada.
-

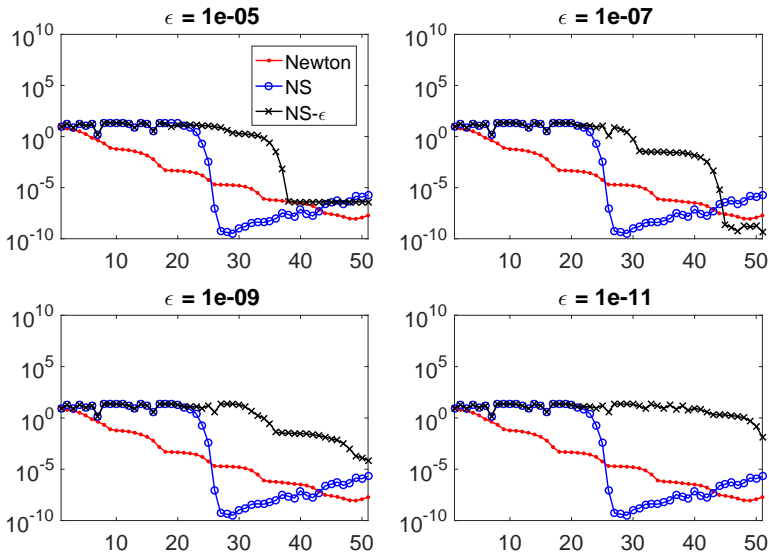
Observação 5.5. O número $\hat{\rho}_1 = (3 - \sqrt{5})/2$ é uma constante técnica necessária ao funcionamento do algoritmo. Está intimamente ligada com os pontos fixos do polinômio que representa a atualização dos autovalores de $X_k A$. Esta escolha de α_0 garante que os autovalores $\rho_j^{(0)}$ de $X_0 A$ correspondentes aos valores singulares $\sigma_j(A) \geq \varepsilon$ estejam mais perto de 1 do que os pequenos valores singulares. Ao tomarmos $\underline{\rho}^{(0)} = \alpha_0 \varepsilon^2$, veja que ε^2 está, basicamente, tomando o lugar de σ_{min} , se comparamos com o que foi apresentado na Subseção 5.1.1.

A iteração (5.13) não é a maneira mais eficiente de computar A_ϵ^\dagger . Podemos trocar tal por

$$X_{k+1} = (-2X_k A + 3I)X_k A X_k, \quad (5.14)$$

que está associada com um polinômio cúbico, reduzindo, assim, uma multiplicação de matrizes se comparado com (5.13): apenas três são necessárias. Como o polinômio envolvido é diferente, é preciso também alterar o valor de $\hat{\rho}_1$ no algoritmo apresentado acima. Assim, se escolhemos por utilizar (5.14), tomamos $\hat{\rho}_1 = 1/2$.

Figura 5.5 – Resíduos $\|b - Ax^{(k)}\|_2$ para Newton, NS (com $\sigma_{\min} = 10^{-12}$) e NS- ϵ , aplicados a shaw, dimensão 100, sem ruído.



Fonte: o autor, 2017.

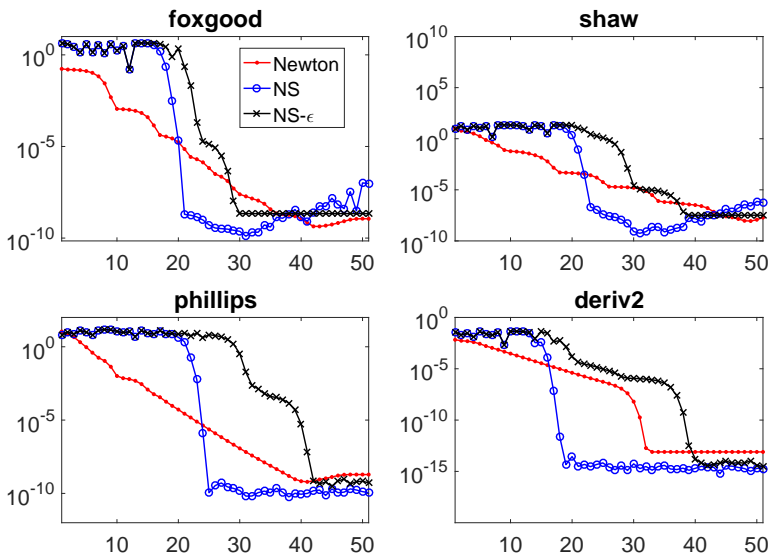
A principal vantagem do ponto de vista teórico se encontra na estabilidade de (5.13) e (5.14) para qualquer matriz A , diferentemente do que acontece com o método de Newton e sua versão acelerada (5.1). De fato, Newton é estável para matrizes não-singulares, mas esta propriedade é alterada no caso singular [35]. Portanto, a utilização de (5.13) no Algoritmo 5.2 tem efeito sobre a instabilidade da versão acelerada do método de Newton, gerando uma solução melhor condicionada. Uma estratégia similar poderia ser utilizada diretamente no método de Newton,

a partir da aplicação de algumas iterações de (5.13) ou (5.14) quando sabemos que os valores singulares significativos já convergiram, o que tem efeito sobre a instabilidade do método.

Assim como acontece com σ_{min} para a aplicação de (5.1) na Subseção 5.1.1, a escolha de ε influi diretamente na eficiência do algoritmo, como ilustra a Figura 5.5. Pelo algoritmo apresentado acima, quanto menor o valor de ε , maior é o número de iterações de Newton modificado, para somente então aplicar (5.13). Mas mesmo para ε maiores, os resultados finais são inferiores aos apresentados por somente NS.

Agora, a Figura 5.6 apresenta resultados de NS- ε para um valor de ε fixo. Apesar da inferioridade de velocidade de convergência e precisão (quando comparado com NS), note a estabilidade numérica de NS- ε , visível pelos erros se manterem “constantes” a partir de alguma iteração em específico, perto da convergência máxima atingida.

Figura 5.6 – Resíduos $\|b - Ax_k\|_2$ para Newton, NS (com $\sigma_{min} = 10^{-10}$) e NS- ε (com $\varepsilon = 10^{-6}$), dimensão 100, sem ruído.



Fonte: o autor, 2017.

5.1.4 NCS: método misto

Na última seção de [35] é sugerido o que é entendível como uma “iteração mista”, que seria uma junção de Newton, NS e NC; sua vantagem se encontra na escolha automática do parâmetro que atualiza α_k , baseado na negação da iteração cúbica. Lembramos que uma das dificuldades inerentes ao método NS é a escolha de $\sigma_{min} \leq \sigma_r^2$, aqui efetuada internamente, como veremos. Este método seria a melhor das possibilidades (no sentido de aceleração do método de Newton), embora torna o algoritmo sensivelmente mais complexo: possibilita a inclusão das estratégias de aceleração cúbica e escalada no mesmo algoritmo, somente aplicando Newton diretamente caso as duas anteriores tivessem sido rejeitadas.

A ideia aqui consiste de uma variação do método NC, quando o passo cúbico é descartado, momento em que buscamos introduzir uma iteração da aceleração com polinômios de Tchebychev, como discutido na Subseção 5.1.1. Para uma aplicação prática, precisamos encontrar um limitante $\rho_* > 0$ tal que $\rho_r^{(k)} < \rho_*$. Então, tomamos como parâmetro de aceleração

$$\alpha_k = \frac{2}{1 + (2 - \rho_*)\rho_*}. \quad (5.15)$$

Esta escolha garante que o processo de aceleração não leva um valor singular grande a ser mapeado à esquerda de um pequeno, o que tornaria o problema mais difícil. Para o método NC, tomamos $T = X_k A$, calculamos $\delta = \|T - T^2\|_F$ e, caso $\delta < 1/4$, aceleramos com o polinômio cúbico. Agora, caso $\delta \geq 1/4$, consideramos

$$\underline{\delta} = \delta / \sqrt{n}.$$

Então, se $\underline{\delta} < 1/4$, tomamos

$$\rho_* = \frac{1}{2} - \sqrt{\frac{1}{4} - \underline{\delta}}$$

e, desde que (5.12) é válida,

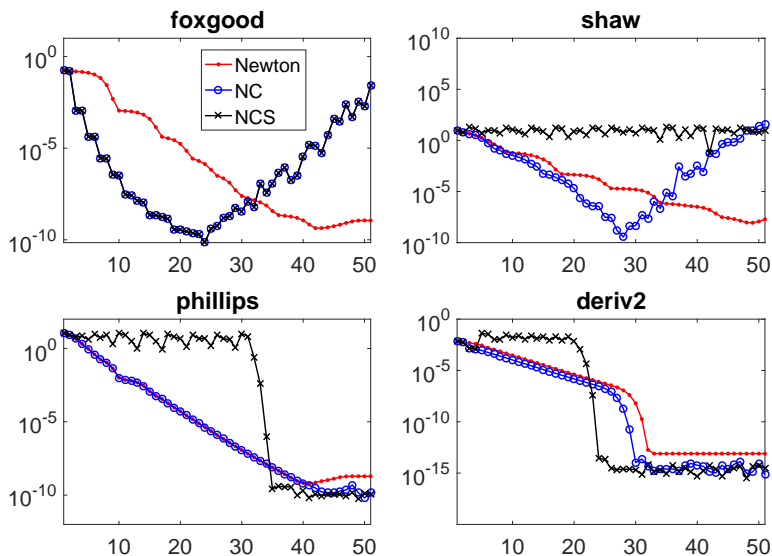
$$\min_{1 \leq j \leq n} (\rho_j - \rho_j^2) \leq \underline{\delta}.$$

Portanto, devemos estar no caso em que ρ_r (o menor valor singular positivo de $X_k A$) é menor que ρ_* ou então todos os valores singulares estão no intervalo $(1 - \rho_*, 1]$. Para descartar esta última possibilidade, verificamos se

$$\text{trace}(X_k A) \geq n(1 - \rho_*)$$

e, em caso de negação, aplicamos a aceleração com NS, isto é, um passo de (5.1), com o parâmetro α_k dado como em (5.15). Caso estes testes sejam negados, então aplicamos, simplesmente, um passo de Newton usual e reiniciamos o processo. Este método pode ser facilmente implementado através de uma modificação do algoritmo para NC presente em [35] e, portanto, não vamos explicitar aqui. Por ser uma mistura das iterações de NC e NS, denotamos este método por *NCS*. A Figura 5.7 apresenta alguns resultados de NCS quando comparado com Newton e NC, para problemas sem ruído. Por ser uma estratégia modificada de NC, é esperado que NCS funcione com eficiência e qualidade igual ou melhor que NC, que é o que verificamos na figura, exceto para o problema *shaw*.

Figura 5.7 – Resíduos $\|b - Ax_k\|_2$ para Newton, NC e NCS, dimensão 100, sem ruído.



Fonte: o autor, 2017.

5.1.5 Aplicação à problemas com perturbações

Buscamos verificar o funcionamento dos métodos apresentados nesta seção quando aplicados na resolução de problemas com perturbações da forma $Ax = \tilde{b} = b + e$, em que buscamos a solução do problema original $Ax = b$, sem conhecer b propriamente.

Por vias de notação, tomamos $\tilde{x}^{(k)} = X_k \tilde{b}$, com X_k a aproximação para a pseudo-inversa A^\dagger gerada pelo método respectivo (Newton, NS (com $\sigma_{min} = 10^{-10}$), NC, NS- ε (com $\varepsilon = 10^{-6}$) e NCS). Além disso, x_{LS} representa a solução de mínimos quadrados de norma mínima para o problema $Ax = b$, ou seja, $x_{LS} = A^\dagger b$. Os testes são nos moldes dos apresentados na Seção 4.2: tomamos 35 como número máximo de iterações, efetuamos 30 resoluções de cada problema e, ao fim, computamos erros relativos para cada resolução, bem como iteração mínima k_m e máxima k_M de parada, para o parâmetro ótimo (OP) e princípio da discrepância (DP). Por parâmetro ótimo, estamos considerando a iteração que minimiza o erro relativo entre x_{LS} e as soluções $\tilde{x}^{(k)}$ computadas pelo método respectivo. Em palavras, OP encontra a melhor solução possível entre todas as calculadas. Fizemos uso apenas de DP como critério de parada simplesmente pelo fato de que não possuímos a propriedade de monotonia nas normas dos resíduos e dos iterados para os métodos desta seção, fato este verificável através de uma inspeção aos exemplos gráficos apresentados. A falta desta propriedade impossibilita o uso de critérios heurísticos, tais como L-curve e MPR, o que é uma dificuldade prática à aplicação dos métodos. A Tabela 5.1 exemplifica resultados encontrados para alguns problemas, tomando ruído de 1%. Lembramos que resultados associados a erro relativo superior a 0.5 são descartados e, por esta razão, algumas células da tabela se encontram vazias.

Veja que os resultados para NS e NS- ε não são de qualidade, mesmo para os parâmetros ótimos; note também que, para NS e NS- ε , o princípio da discrepância não encontra iteração de parada ou, se encontra, não é com erro inferior a 0.5. Em contraste, NC reduz (em alguns dos problemas, como foxgood e baart) o número de iterações drasticamente, se comparado com Newton, garantindo precisão próxima deste também. Já NCS apresenta comportamento diferenciado, em alguns problemas adquirindo precisão de alta qualidade, não atingida por nenhum dos métodos, ao passo que o número de iterações utilizadas cresce.

Todavia, o comportamento em geral dos métodos, quando aplicados à problemas com perturbações, não é satisfatório: temos dificuldade de encontrar soluções de qualidade e, em caso afirmativo, geralmente implicam em alto custo operacional. Como não conseguimos introduzir alguma manipulação nos moldes de (3.22), evitando os altos custos por iteração dos métodos desta seção, seu uso apresenta dificuldades, visto que perdemos propriedades boas do método de Newton (como a monotonia das normas) e, mais ainda, temos complicações com escolhas de parâmetros iniciais e possíveis ineficiências (como, por exemplo, o

Tabela 5.1 – Resultados numéricos para os métodos de Newton (Ne.), NS (com $\sigma_{min} = 10^{-10}$), NC, NS- ε (com $\varepsilon = 10^{-6}$) e NCS, dimensão 500 e $NL = 0.01$.

Método			foxgood	shaw	gravity	baart	deriv2
OP	$k_m(k_M)$	Ne.	10(17)	15(23)	10(15)	14(26)	14(16)
		NS	4(8)	7(7)	1(1)	3(3)	4(9)
		NC	3(6)	12(17)	10(15)	5(8)	12(14)
		NS- ε	4(8)	7(7)	1(1)	3(3)	4(9)
		NCS	16(16)	30(30)	11(11)	19(19)	32(32)
	E_m	Ne.	0.0208	0.0634	0.0282	0.1347	0.2195
		NS	0.2938	0.1869	0.3676	0.3417	0.4692
		NC	0.0275	0.0644	0.0282	0.1493	0.2194
		NS- ε	0.2938	0.1869	0.3676	0.3417	0.4695
		NCS	0.0001	0.1077	0.1634	0.0196	0.0000
DP	$k_m(k_M)$	Ne.	9(9)	9(11)	8(9)	12(13)	12(12)
		NS	-(-)	-(-)	-(-)	-(-)	-(-)
		NC	3(3)	9(9)	8(9)	5(5)	9(10)
		NS- ε	-(-)	-(-)	-(-)	-(-)	-(-)
		NCS	3(3)	-(-)	-(-)	5(5)	28(28)
	E_m	Ne.	0.0324	0.1619	0.0462	0.1757	0.2662
		NS	-	-	-	-	-
		NC	0.0311	0.1368	0.0462	0.1667	0.2656
		NS- ε	-	-	-	-	-
		NCS	0.0310	-	-	0.1659	0.0003

Fonte: o autor, 2017.

passo cúbico ser sempre rejeitado e termos apenas custo do cálculo sem aceleração).

Sem dúvidas, as estratégias propostas por Pan e Schreiber [35] apresentam resultados significativos em ganho de iterações e precisão quando aplicados à sua funcionalidade original, que é computar a pseudo-inversa A^\dagger , se comparados com o algoritmo de Newton usual. No entanto, no que diz respeito a problemas lineares com perturbações, caso de interesse neste trabalho, estes métodos para aproximar A^\dagger não apresentam resultados satisfatórios. Em algumas situações, é possível conseguir soluções melhores que as computadas por Newton usual, conseqüentemente necessitando de maior esforço computacional. Portanto, como um todo, estas variações ao método de Newton não apresentam ganho em aceleração operacional suficiente quando aplicadas a problemas lineares e, assim,

alguma outra estratégia precisa ser considerada.

5.2 ACELERAÇÃO POR PROJEÇÃO

Resolver o problema

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2, \quad (5.16)$$

com $A \in \mathbb{R}^{m \times n}$, pode levar a alto computacional se atacado diretamente via o método vetorial baseado na iteração de Newton (3.23), dado por

$$x^{(k+1)} = x^{(k)} + U_0^{2^k} x^{(k)}, \quad k = 0, 1, \dots,$$

em que $x^{(0)} = \beta A^T b$ e $U_0 = I - \beta A^T A$, para β apropriado. Veja que $U_0 \in \mathbb{R}^{n \times n}$ e, portanto, para um implementação de (3.23), podemos ter um número alto de operações por iteração, ocasionado pela necessidade de calcular um produto de matrizes $n \times n$ ou então 2^k produtos matriz-vetor envolvendo U_0 . Em ambos os casos, a aplicação do método pode se tornar impraticável para problemas de grande porte.

Tendo em vista contornar esta dificuldade, buscamos utilizar uma técnica de projeção, ou seja, vamos considerar resolver o seguinte problema:

$$\min_x \|Ax - b\|_2 \quad \text{sujeito a} \quad x \in \mathcal{V}_\ell = \text{span}\{\mathbf{v}_1, \dots, \mathbf{v}_\ell\}, \quad (5.17)$$

que chamaremos de *problema projetado*, para vetores \mathbf{v}_i apropriadamente escolhidos. Esta estratégia tem como premissa a esperança de que o espaço \mathcal{V}_ℓ contenha a maior parte dos componentes da solução x_{LS} procurada; esta ideia demanda que a escolha dos vetores \mathbf{v}_i seja efetuada com cautela, como abordaremos a frente. Se denotarmos

$$\mathbf{V}_\ell = [\mathbf{v}_1, \dots, \mathbf{v}_\ell] \in \mathbb{R}^{n \times \ell},$$

então estamos buscando $y \in \mathbb{R}^\ell$ de modo que $x = \mathbf{V}_\ell y$. Portanto, note que (5.17) pode ser visto como

$$\min_{y \in \mathbb{R}^\ell} \|By - b\|_2, \quad \text{em que} \quad B = A\mathbf{V}_\ell, \quad (5.18)$$

que permite aplicação direta do método iterativo proposto neste trabalho. Perceba que $B \in \mathbb{R}^{m \times \ell}$ e, deste modo, ao aplicarmos a sequência vetorial gerada pelo método de Newton no sistema $By = b$, a matriz $U_0 = I - \beta B^T B$ é tal que $U_0 \in \mathbb{R}^{\ell \times \ell}$. Logo, necessitamos efetuar

operações matriciais, agora, com uma matriz $\ell \times \ell$, ao invés de $n \times n$, como na abordagem direta pelo método de Newton vetorial para (5.16), isto é, $Ax = b$. Portanto, se $\ell \ll n$, a vantagem numérica de tal possibilidade é evidente.

Cada iteração do método (3.23) quando aplicado ao problema (5.18) gera um vetor $y^{(k)} \in \mathbb{R}^\ell$ que representa uma aproximação para $y_{LS} = B^\dagger b$, já que $y^{(k)} \rightarrow y_{LS}$. Assim, se tomamos $x^{(k)} = V_\ell y^{(k)}$, obtemos aproximações para a solução de norma mínima de (5.16), isto é, $x^{(k)} \approx x_{LS}$, o que estamos buscando. Na prática, aplicamos a iteração de Newton em (5.18) até algum critério de parada ser atingido (gerando, digamos, $y^{(*)}$) e, então, tomamos $x^{(*)} = V_\ell y^{(*)}$ como aproximação para x_{LS} . O Algoritmo 5.3 resume esta ideia.

Algoritmo 5.3 Método de Newton com projeção

Entrada: A, b, V_ℓ

Saída: Aproximação $x^{(*)}$ de x_{LS}

- 1: $B \leftarrow AV_\ell$
 - 2: Aplique o método de Newton (Algoritmo 3.2 ou 3.3) no sistema $By = b$ até algum critério de parada apropriado ser atingido, gerando uma solução regularizada, digamos, $y^{(*)}$.
 - 3: $x^{(*)} \leftarrow V_\ell y^{(*)}$
-

Dos aspectos computacionais, a aplicação do método de Newton no sistema $By = b$ gera a sequência

$$y^{(k+1)} = y^{(k)} + U_0^{2^k} y^{(k)}, \quad k = 0, 1, \dots, \quad (5.19)$$

em que $y^{(0)} = \beta B^T b$ e $U_0 = I - \beta B^T B \in \mathbb{R}^{\ell \times \ell}$, com

$$0 < \beta < \frac{2}{\rho(BB^T)}. \quad (5.20)$$

O cálculo da constante β pode ser feito, em última instância, através da bidiagonalização de Lanczos. No entanto, para melhor aproveitamento computacional da estratégia de projeção como um todo, sugerimos o cálculo de $B^T B$ explicitamente num estágio inicial, matriz esta intimamente ligada à construção de U_0 também. Mesmo com A na forma implícita, como os vetores v_i do espaço \mathcal{V}_ℓ são dados, podemos computar as quantidades Av_i e armazenar os ℓ vetores resultantes, levando em conta que ℓ é pequeno (se comparado com n). Deste modo, as entradas de $B^T B$ são acessadas através de produtos internos, pois

$$(B^T B)_{i,j} = (Av_i)^T (Av_j).$$

Assim, como $\rho(BB^T) = \rho(B^TB)$, podemos encontrar β de (5.20) fazendo uso de alguma estratégia que utiliza a matriz explícita de pequena ordem B^TB já computada. Por exemplo, usando que $\|B^TB\|_2 \leq \|B^TB\|_F$, basta tomar

$$\beta = \frac{1}{\|B^TB\|_F^2}.$$

Com esta constante em mãos, construímos rapidamente $U_0 = I - \beta(B^TB)$ e o processo iterativo (5.19) pode ser iniciado.

A grande questão é a escolha do subespaço \mathcal{V}_ℓ , de modo que informações importantes do problema sejam representadas nesse espaço e, ao mesmo tempo, com poucos vetores na base (i.e., com ℓ pequeno), assegurando a aplicação eficiente da sequência vetorial. Nas subseções seguintes abordamos algumas possibilidades e, ao final, apresentamos testes numéricos.

5.2.1 Determinação da dimensão do subespaço

Considere o problema linear com lado direito perturbado $Ax = \tilde{b}$. Para cada $j \geq 1$, tome a notação

$$x^{[j]} = \operatorname{argmin}_{x \in \mathcal{V}_j} \|Ax - \tilde{b}\|_2$$

e assuma que $\mathbf{V}_j = [\mathbf{v}_1, \dots, \mathbf{v}_j] \in \mathbb{R}^{n \times j}$ é tal que $\mathcal{R}(\mathbf{V}_j) = \mathcal{V}_j$, para $j = 1, 2, \dots$, como anteriormente. A ideia aqui consiste em escolher ℓ de modo que $x^{[\ell]}$ capture informações relevantes da solução sem ruído x_{LS} . Para tanto, faremos uso do princípio da discrepância. Suponha que possuímos uma matriz $\mathbf{U}_j = [\mathbf{u}_1, \dots, \mathbf{u}_j] \in \mathbb{R}^{m \times j}$ tal que suas colunas formam uma base ortonormal para $\mathcal{R}(A\mathbf{V}_j)$. É fácil ver que o j -ésimo resíduo $r^{[j]} = \tilde{b} - Ax^{[j]}$ tem norma satisfazendo

$$\|r^{[j]}\|_2 = \|\tilde{b} - \mathbf{U}_j \mathbf{U}_j^T \tilde{b}\|_2. \quad (5.21)$$

Assim, podemos monitorar as quantidades $\|r^{[j]}\|_2$ até que DP seja satisfeito e escolher a dimensão do subespaço como j . Essencialmente, este procedimento identifica qual deve ser o tamanho do subespaço para que uma solução de qualidade possa ser calculada dentro dele. Após determinar ℓ , aplicamos o método de Newton no problema projetado (5.17) até convergência satisfatória. Como já utilizamos DP para encontrar ℓ (que aqui atua como parâmetro de regularização), Newton é aplicado até a convergência.

Observe que o procedimento para determinação do parâmetro de regularização através de DP não demanda o cálculo dos vetores $x^{[j]}$ explicitamente: necessitamos, no passo $j + 1$, computar a projeção

$$U_{j+1}U_{j+1}^T\tilde{b} = u_1(u_1^T\tilde{b}) + u_2(u_2^T\tilde{b}) + \cdots + u_{j+1}(u_{j+1}^T\tilde{b}),$$

que é rapidamente gerada através do passo anterior, uma vez que

$$U_{j+1}U_{j+1}^T\tilde{b} = U_jU_j^T\tilde{b} + u_{j+1}(u_{j+1}^T\tilde{b}).$$

Para as aplicações numéricas, consideramos os seguintes subespaços de Krylov:

$$\mathcal{V}_j = \mathcal{K}_j(A^T A, A^T \check{r}) = \text{span}\{A^T \check{r}, (A^T A)A^T \check{r}, \dots, (A^T A)^{j-1}A^T \check{r}\}, \quad (5.22)$$

para \check{r} vetor randômico, e

$$\mathcal{V}_j = \mathcal{K}_j(A, A\tilde{b}) = \text{span}\{A\tilde{b}, A^2\tilde{b}, \dots, A^j\tilde{b}\}. \quad (5.23)$$

Além da capacidade de gerar boas aproximações para o subespaço dos vetores singulares associados com os maiores valores singulares, estes subespaços são interessantes pois permitem construir iterativamente uma base para o subespaço de Krylov ao mesmo tempo que provém uma base aproximada para $\mathcal{R}(AV_j)$ [20]. No caso de (5.22), podemos utilizar a já apresentada bidiagonalização de Lanczos, que gera uma decomposição parcial da forma

$$A\check{V}_j = \check{U}_{j+1}B_j, \quad (5.24)$$

em que $\check{V}_j \in \mathbb{R}^{n \times j}$ contém uma base ortonormal para o subespaço de Krylov, $\check{U}_{j+1} \in \mathbb{R}^{m \times (j+1)}$ tem colunas ortonormais com $\check{u}_1 = \check{r}/\|\check{r}\|_2$ e B_j é bidiagonal inferior. De maneira semelhante, podemos construir uma base para (5.23) através do algoritmo de Arnoldi/Lanczos [16] aplicado a A , gerando a decomposição parcial

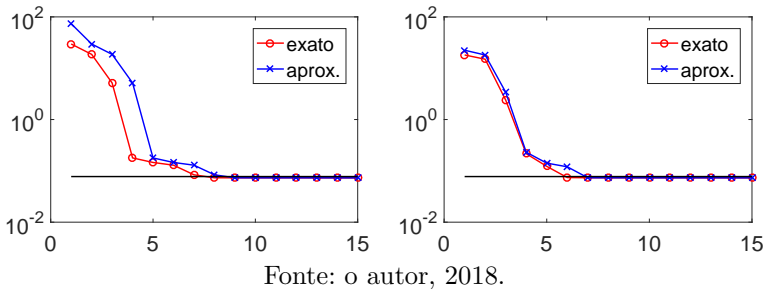
$$A\check{Q}_j = \check{Q}_{j+1}G_j, \quad (5.25)$$

em que, paralelamente ao caso anterior, \check{Q}_j provém uma base ortonormal para o subespaço de Krylov (5.23) e $G_j \in \mathbb{R}^{n \times (j+1)}$ é uma matriz Hessenberg superior. Vale observar que, no caso em que A é simétrica, G_j se reduz a uma matriz tridiagonal. A primeira coluna de \check{Q}_j é dada por $\check{q}_1 = A\tilde{b}/\|A\tilde{b}\|_2$.

Note que (5.24) não produz uma base ortonormal para $\mathcal{R}(A\check{V}_j)$. Entretanto, na resolução de problemas discretos mal postos, com o

crescimento de j as colunas de \check{U}_j tendem a gerar uma boa aproximação para $\mathcal{R}(A\check{V}_j)$. Portanto, uma maneira de aproximar a norma residual (5.21) é através da utilização de \check{U}_j no lugar de U_j (que é desconhecida a priori). Uma observação semelhante se aplica para a decomposição (5.25), referente ao subespaço de Krylov (5.23). Resultados neste sentido são apresentados na Figura 5.8 e evidenciam que, com o crescimento de j , as aproximações das normas residuais se tornam cada vez melhores.

Figura 5.8 – Normas residuais (5.21) exatas e aproximadas para alguns j , problema *shaw*, dimensão 1000, com perturbação de 0.1% nos dados de entrada e $\tau = 1.05$. À esquerda, resultados para o subespaço (5.22); à direita, resultados para o subespaço (5.23); a linha horizontal corresponde a τNL , determinando parada por DP.



Resumindo, a escolha de ℓ neste caso consiste em dois passos: calculamos as normas residuais de acordo com (5.21) para $j \geq 1$, substituindo U_j por \check{U}_j ou \check{Q}_j (conforme o espaço utilizado) e terminamos as iterações quando o princípio da discrepância é atingido.

Observação 5.6. (i) Após a determinação de ℓ , a decomposição parcial (5.24) simplifica ainda mais a resolução do problema projetado (5.18), pois:

$$\|A\check{V}_\ell y - \tilde{b}\|_2 = \|\check{U}_{\ell+1} B_\ell y - \tilde{b}\|_2 = \|B_\ell y - \check{U}_{\ell+1}^T \tilde{b}\|_2.$$

Isto é, substituímos sem perda o sistema projetado $By = \tilde{b}$ por $B_\ell y = \check{U}_{\ell+1}^T \tilde{b}$, em que a estrutura bidiagonal de B_ℓ reduz o número de operações realizadas pelo método de Newton aplicado nesse sistema, tornando o custo computacional desta parte do processo negligenciável, devido a ℓ ser pequeno se comparado com m ou n .

Um comentário análogo se aplica à decomposição (5.25), que leva à resolução de um problema Hessenberg ou tridiagonal (dependendo da simetria de A).

- (ii) Lembramos que o algoritmo LSQR inicia o processo de bidiagonalização com $\mathbf{u}_1 = \tilde{\mathbf{b}}/\|\tilde{\mathbf{b}}\|_2$, o que não pode ser efetuado aqui, pois os resíduos (5.21) se tornariam constantes e o uso de DP para encontrar o parâmetro ℓ que estamos procurando não iria funcionar. Isto explica o motivo de iniciarmos o processo de bidiagonalização com um vetor randômico $\check{\mathbf{r}}$.

5.2.2 Subespaço dado a priori

Diferentemente do caso anterior, aqui supomos que uma base para o subespaço \mathcal{V}_ℓ é fornecida previamente. Nesta situação, a escolha de ℓ pode ser feita através de tentativa e erro ou por treinamento, como efetuado em [13]. É necessário, no entanto, que ℓ seja suficientemente grande de modo que as soluções $x^{(k)} = \mathbf{V}_\ell y^{(k)}$ sejam dominadas pelo ruído. Em posse dos iterados $x^{(k)}$, as iterações são terminadas pelo uso de algum critério de parada apropriado (DP, L-curve e MPR são alguns exemplos).

Nos nossos exemplos, faremos uso do espaço gerado pela base de vetores ortonormais da transformada discreta de cosseno [22] (conhecida pela sigla em inglês DCT, *discrete cosine transform*). Os vetores $\mathbf{v}_i \in \mathbb{R}^n$ desta base são dados por

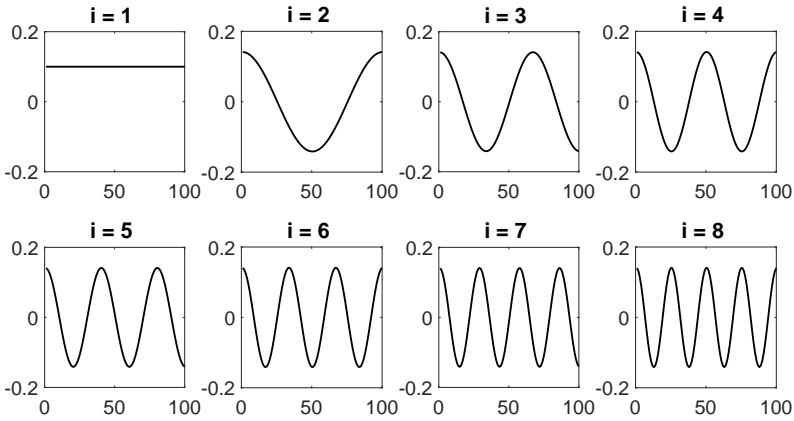
$$\mathbf{v}_1 = \sqrt{1/n} (1, 1, \dots, 1)^T \quad (5.26)$$

e, para $i = 2, 3, \dots$,

$$\mathbf{v}_i = \sqrt{2/n} \left(\cos\left(\frac{i\pi}{2n}\right), \cos\left(\frac{3i\pi}{2n}\right), \dots, \cos\left(\frac{(2n-1)i\pi}{2n}\right) \right)^T. \quad (5.27)$$

A Figura 5.9 apresenta os gráficos de alguns dos vetores iniciais da base. Note a oscilação característica da função cosseno presente nos vetores. Estas mesmas oscilações são o que torna a base DCT de importância aqui: de certa maneira, ela emula o comportamento dos vetores singulares dominantes de A . Em outros termos, DCT gera um subespaço que aproxima bem o subespaço gerado pelos vetores singulares dominantes de A . Como as componentes mais importantes da solução estão contidas, em geral, nestes mesmos vetores singulares, é razoável esperar que \mathcal{V}_ℓ assim escolhido contenha aproximações de qualidade para x_{LS} .

Figura 5.9 – Alguns vetores v_i da base DCT, para dimensão $n = 100$.

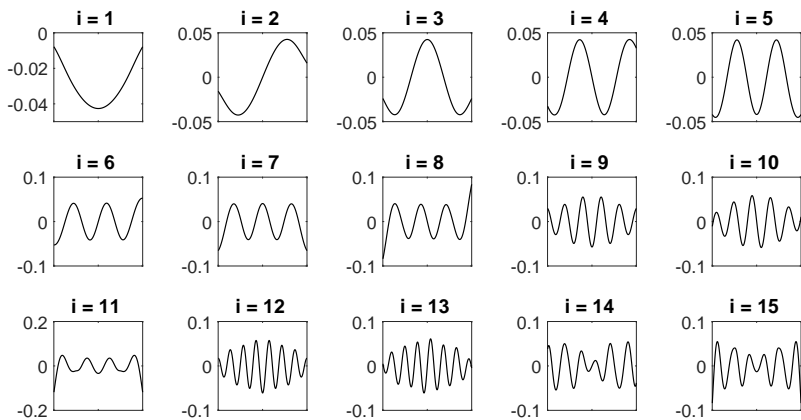


Fonte: o autor, 2017.

Observe, por exemplo, a Figura 5.10, que contém os primeiros vetores singulares à esquerda para o problema phillips. Como os valores singulares são ordenados de forma não crescente, estes vetores são os associados aos maiores valores singulares de A , em geral responsáveis por gerar boas aproximações para x_{LS} . Portanto, ao conseguirmos uma base que imite o comportamento destes vetores, não é surpresa gerarmos aproximações de qualidade para x_{LS} .

Observação 5.7. Ressaltamos que o comportamento oscilatório dos vetores singulares tende a aumentar de acordo com o decrescimento dos valores singulares [22], como podemos observar na Figura 5.10. Isto é, os primeiros vetores singulares tendem a ser suaves, enquanto que vetores associados a σ_i 's menores são de alta frequência.

Figura 5.10 – Comportamento de alguns vetores singulares à esquerda u_i do problema phillips, dimensão 1000.



Fonte: o autor, 2017.

5.2.3 Resultados numéricos

Para testar as ideias de aceleração propostas nesta seção, executaremos dois exemplos numéricos: o primeiro nos moldes do que foi efetuado na Seção 4.2, mas para $n = 10000$, e o segundo é um problema de remoção de *blur* de uma imagem.

Para a primeira parte, como mencionado, testaremos as estratégias de projeção em problemas já estabelecidos neste trabalho, apenas com acréscimo na dimensão, agora dada por $n = 10000$. Para a descrição dos resultados, os métodos baseados nos subespaços (5.22), (5.23) e no subespaço gerado pelos vetores em (5.26)-(5.27) serão denotados por N-Bid, N-Trid e N-DCT, respectivamente. No caso de N-DCT, fornecemos a base DCT com $\ell = 30$ vetores. As comparações são novamente efetuadas com o algoritmo LSQR. Por questões de simplicidade, tomamos apenas DP como critério de parada para LSQR e N-DCT, com $\tau = 1.05$; N-Bid e N-Trid já incorporam DP internamente (através da escolha do tamanho ℓ do subespaço de Krylov respectivo), sendo que as iterações de Newton subsequentes são paradas quando detectamos que as normas dos resíduos $r^{(k)}$ estão praticamente estacionadas, isto é, quando

$$\frac{\left| \|r^{(k)}\|_2 - \|r^{(k-1)}\|_2 \right|}{\|r^{(k)}\|_2} < \varepsilon,$$

em que, aqui, $\varepsilon = 10^{-4}$. Erros relativos médios de 30 realizações para os níveis de ruído $NL = 0.01$ e $NL = 0.001$ são apresentados nas Tabelas 5.2 e 5.3. Lembramos que os menores erros relativos são exibidos em negrito e que erros maiores que 0.5 (50%) são descartados, representados pelo “traço” nas tabelas.

Tabela 5.2 – Erros relativos para $NL = 0.01$, $n = 10000$.

Problema	LSQR	N-DCT	N-Bid	N-Trid
foxgood	0.0310	0.0317	0.0235	0.0310
phillips	0.0244	0.0245	0.0239	0.0239
heat	0.0976	0.0850	0.1197	-
shaw	0.1679	0.1631	0.1326	0.1665
gravity	0.0467	0.0478	0.0422	0.0613
baart	0.1659	0.1704	0.1267	0.0360
deriv2	0.2508	0.2621	0.2715	0.3038
moler	0.1877	0.1959	0.1476	0.2721
lotkin	0.4827	0.4866	0.4672	-
prolate	0.0071	0.0012	-	0.0071
lehmer	0.1308	0.1280	0.1185	0.2082
cauchy	0.4441	0.4435	0.4436	0.4612
fiedler	0.1177	0.0861	0.0788	0.1023
frank	0.0707	0.0647	0.0999	0.3142
hilb	0.4492	0.4439	0.4473	0.4478

Fonte: o autor, 2018.

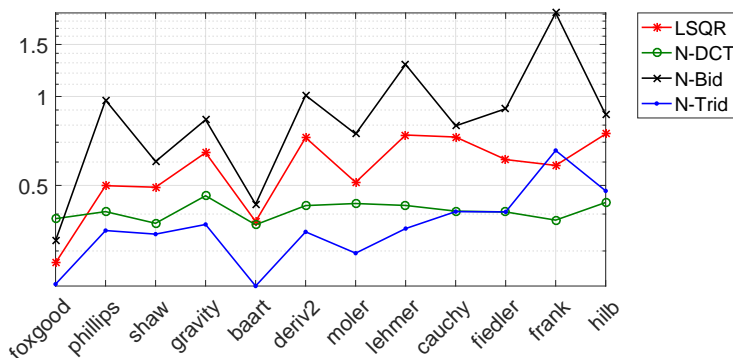
Em essência, os resultados apresentados são comparáveis com os obtidos pelo algoritmo LSQR e, em alguns casos, mais precisos que este (como indicado pelos números em negrito). Observamos também que mesmo com o aumento da dimensão em relação aos resultados com $n = 1000$ da Seção 4.2, os erros relativos do método de Newton acoplado com as estratégias de projeção se mantém de qualidade, em algumas situações até melhores (veja Tabelas 4.4 e 4.5).

Agora, com relação ao custo operacional dos métodos, apresentamos os tempos médios de execução (calculados via MATLAB) para os dois níveis de ruído nas Figuras 5.11 e 5.12. Ressaltamos que nas ilustrações apenas consideramos métodos que produziram erros relativos inferiores a 0.5. Diferentemente dos tempos computados para $n = 1000$ (veja Figura 4.2), no cenário atual Newton com projeção se apresenta comparável ao algoritmo LSQR, com alguma pequena vantagem para N-DCT e N-Trid em alguns problemas.

Tabela 5.3 – Erros relativos para $NL = 0.001$, $n = 10000$.

Problema	LSQR	N-DCT	N-Bid	N-Trid
foxgood	0.0310	0.0301	0.0193	0.0310
phillips	0.0089	0.0129	0.0114	0.0201
heat	0.0355	0.0349	0.0430	-
shaw	0.0474	0.0487	0.0485	0.0475
gravity	0.0223	0.0212	0.0175	0.0277
baart	0.1659	0.1648	0.1130	0.0358
deriv2	0.1688	0.1507	0.1811	0.1877
moler	0.0401	0.0397	0.0599	0.0592
lotkin	0.4506	0.4503	0.4521	-
prolate	0.0008	0.0018	-	0.0008
lehmer	0.0272	0.0190	0.0405	0.0450
cauchy	0.4397	0.4401	0.4398	0.4397
fiedler	0.0161	0.0153	0.0203	0.0434
frank	0.0152	0.0127	0.0288	0.0857
hilb	0.4402	0.4401	0.4397	0.4404

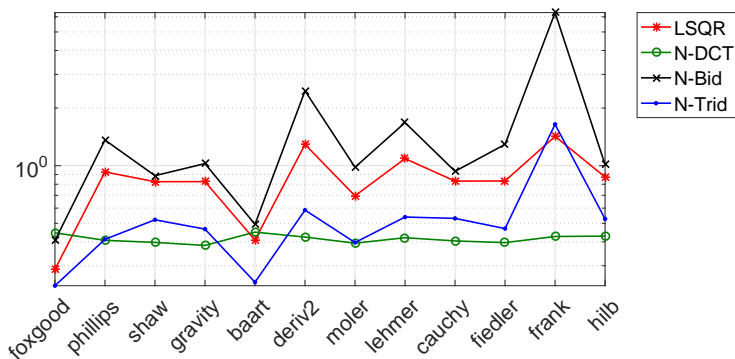
Fonte: o autor, 2018.

Figura 5.11 – Tempo operacional, em segundos, para $NL = 0.01$, $n = 10000$.

Fonte: o autor, 2018.

Para a segunda parte dos testes, faremos uso de uma imagem proveniente do pacote *Restore Tools* [30]. O objetivo é recuperar uma imagem armazenada em um vetor $x^{\text{exato}} \in \mathbb{R}^{MN}$ de uma imagem com

Figura 5.12 – Tempo operacional, em segundos, para $NL = 0.001$, $n = 10000$.



Fonte: o autor, 2018.

blur e ruído $\tilde{b} = b^{\text{exato}} + e$ tal que $Ax^{\text{exato}} = b^{\text{exato}}$, sendo que $A \in \mathbb{R}^{MN \times MN}$ corresponde ao chamado *blurring operator*, isto é, operador que introduz o *blur*. Frequentemente, este operador é conhecido também por matriz PSF (*point spread function*), por determinar quais pontos da imagem serão distorcidos. Note que, neste contexto, b^{exato} representa a imagem com *blur*, na qual adicionamos algum ruído, resultando em \tilde{b} , que contém ambas imprecisões.

No caso do exemplo proposto aqui, a imagem considerada é *satellite*, que contém 256×256 pixels, isto é, $M = N = 256$, de modo que o sistema linear envolvido é de ordem $n = 256^2 = 65536$. Nos experimentos, o nível de ruído é $NL = 0.001$ e efetuamos um total de 30 realizações para os métodos LSQR, N-Bid e N-Trid, cujos resultados médios são exibidos na Tabela 5.4. Por vias de notação, t_m representa o tempo médio de execução e ℓ_m e ℓ_M consistem, respectivamente, dos menores e maiores tamanhos dos subespaços utilizados/escolhidos. Lembre que ℓ é a ordem do problema projetado que resolvemos através do método de Newton. Além disso, k_m e k_M representam, para N-Bid e N-Trid, o número de passos utilizados por Newton aplicado ao problema projetado até atingir convergência.

Na Figura 5.13 apresentamos tanto a imagem original x^{exato} e a imagem com *blur* e ruído \tilde{b} como também os melhores resultados para os métodos utilizados. Na figura, LSQR, N-Bid e N-Trid geraram soluções com erro relativo de 0.2700, 0.5386 e 0.2820, respectivamente.

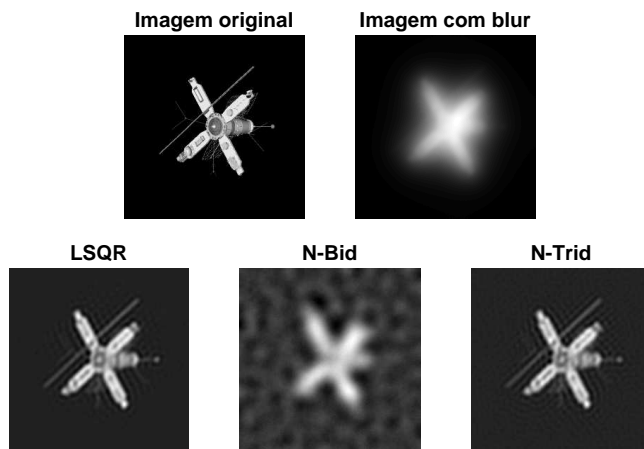
Note que, visualmente, as imagens recuperadas através de LSQR e N-Trid são virtualmente as mesmas, reforçando a qualidade das soluções capturadas.

Tabela 5.4 – Resultados para imagem satellite, $NL = 0.001$, $n = 65536$.

	LSQR	N-Bid	N-Trid
$k_m(k_M)$	119(121)	26(27)	20(22)
E_m	0.2705	0.6092	0.2831
t_m	11.6386	23.2551	2.6234
$\ell_m(\ell_M)$	-	200(200)	45(59)

Fonte: o autor, 2018.

Figura 5.13 – Melhores erros relativos para imagem satellite, $NL = 0.001$.



Fonte: o autor, 2018.

Perceba que os erros relativos gerados por LSQR e N-Trid são essencialmente os mesmos; N-Bid, por outro lado, apresenta dificuldades na escolha do subespaço (atingindo o número máximo considerado de 200 vetores), gerando erros de alta ordem. Esta dificuldade pode ser referente ao fato de N-Bid não considerar informações de \tilde{b} na geração do subespaço (como acontece com N-Trid através do subespaço (5.23)), já que o fazemos pela introdução de um vetor randômico, levando o

método a necessitar de mais vetores na base para emular um subespaço de qualidade. Agora, com relação ao tempo de execução, N-Trid é superior a LSQR, apresentando vantagem mais clara, ao menos para este problema.

Ressaltamos que, no caso de N-Trid, o método de Newton aplicado ao problema projetado efetuou operações envolvendo matrizes de ordem entre 45 e 59, como exibidos na Tabela 5.4. Comparado com o problema original envolvendo 65536 variáveis, o custo de lidar com as matrizes menores é negligenciável numericamente, o que leva ao ganho de tempo operacional evidenciado.

6 CONCLUSÃO

Abordamos aspectos relacionados a problemas discretos mal postos e a dificuldade de encontrar aproximações estáveis para a solução livre de ruído x_{LS} . Neste contexto, propomos um novo método iterativo para resolver tais problemas que se baseia em iterações matriciais geradas pelo método de Newton. Mostramos que tal iteração é quadraticamente convergente e que, sob algumas condições, captura as informações associadas aos elementos de baixa frequência primeiro, respeitando a ordenação natural dos valores singulares. Também verificamos monotonia na norma dos resíduos e dos iterados, possibilitando a utilização de regras heurísticas de parada, tais como L-curve e MPR.

Analisamos o comportamento de semi-convergência dos iterados, apresentando também uma estimativa de erro para o caso em que DP é utilizado como critério de parada, inferindo que, se o nível de ruído tende a zero, então as soluções computadas através do uso deste critério de parada tendem a x_{LS} . Os resultados numéricos apresentados em seguida indicam que o método de Newton vetorial captura soluções de qualidade, mas que o custo operacional é um empecilho em aplicações práticas.

As estratégias de aceleração através do uso de variantes das iterações matriciais de Newton, propostas em [35], apresentam melhora no cenário original, isto é, quando utilizadas para acelerar a convergência à pseudo-inversa A^\dagger . No entanto, na tentativa de aplicar estas ideias em sistemas lineares, dificuldades teóricas e práticas surgem, como a perda da monotonia das normas dos iterados e dos resíduos e até mesmo aumento do esforço computacional. Por outro lado, ao aliarmos o método de Newton vetorial com métodos de projeção em subespaços, conseguimos alternativas aplicáveis em problemas de grande porte, equiparando os resultados gerados pelo algoritmo LSQR. Os exemplos numéricos exibidos mostram que o método de Newton funciona bem quando acoplado com critérios de parada apropriados, tais como DP, L-curve e MPR, indicando que o método é atrativo para problemas discretos mal postos envolvendo dados com ruído em níveis realísticos.

A determinação de subespaços de projeção não é uma tarefa trivial, ainda mais em casos como para a base DCT, que não possui alguma aproximação para a base de $\mathcal{R}(AV_\ell)$, como acontece com os subespaços de Krylov através das decomposições parciais. Uma proposta para trabalhos futuros é analisar técnicas para uma determinação automática

da dimensão do subespaço a partir unicamente dos vetores v_i da base e de Av_i . Além disso, para o caso de restauração de imagens, a base DCT-2D (literalmente DCT em duas dimensões) pode trazer alternativas aos subespaços de Krylov e o uso dessas possíveis estratégias automáticas se torna imperativo.

Outras propostas de trabalhos futuros envolvem:

1. Estender os resultados teóricos apresentados no Capítulo 3 para outras versões do método matricial de alta ordem para a pseudo-inversa, uma vez que fizemos uso somente do caso particular que resulta na iteração de Newton.
2. Generalizar a estimativa teórica do Capítulo 4 incluindo condições de fonte do tipo Holder e logarítmicas, afim de inferir uma estimativa mais fina para casos em que temos informações acerca da suavidade da solução x_{LS} .
3. Analisar técnicas para a inclusão de informações adicionais do problema no processo do método de Newton. Em muitas situações na prática, é possível que tenhamos informações adicionais do problema $Ax = b$ que não estão contidas na matriz A ou no vetor b . Um caso típico ocorre em problemas com equações integrais, em que é natural esperarmos que a solução exata seja, em muitos casos, ao menos diferenciável. Estas informações dadas a priori tendem a introduzir dados acerca da suavidade e/ou comportamento da solução exata, podendo gerar melhorias significativas nas soluções aproximadas.

REFERÊNCIAS

- [1] BAI, Z.; DEMMEL, J.; DONGARRA, J.; RUHE, A.; VAN DER VORST, H., editors. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, Philadelphia: SIAM, 2000. 410 p.
- [2] BAKER, C. T. H. *The Numerical Treatment of Integral Equations*, Oxford: Clarendon Press, 1977.
- [3] BAKUSHINSKI A. B.. Remarks on choosing a regularization parameter using quasioptimality and ratio criterion, *USSR Computational Mathematics and Mathematical Physics*, v. 24, p. 181–182, 1984.
- [4] BAZÁN, F. S. V. Fixed-point iterations in determining the Tikhonov regularization parameter, *Inverse Problems*, v. 24, n. 3, DOI:10.1088/0266-5611/24/3/035001, 2008.
- [5] BAZÁN, F. S. V.; BOOS, E. Schultz matrix iteration based method for stable solution of discrete ill-posed problems, *Linear Algebra and its Applications*, submetido, Dezembro/2016 (1ª revisão em Janeiro/2018).
- [6] BAZÁN, F. S. V.; BORGES, L. S. GKB-FP: an algorithm for large-scale discrete ill-posed problems, *BIT*, DOI 10.1007/s10543-010-0275-3, 2010.
- [7] BAZÁN, F. S. V.; FRANCISCO, J. B. Improved fixed-point algorithm for determining the Tikhonov regularization parameter, *Inverse Problems*, v. 25, n. 4, DOI:10.1088/0266-5611/25/4/045007, 2009.
- [8] BEN-ISRAEL, A.; COHEN, D. On iterative computation of generalized inverses and associated projections, *J. SIAM Numer. Anal.*, v. 3, n. 3, p. 410–419, 1966.
- [9] BEN-ISRAEL, A.; GREVILLE, T. N. E. *Generalized Inverses: Theory and Applications*, 2nd. ed., New York: Springer, 2003. 420 p.
- [10] BJÖRCK, A. *Numerical Methods for Least Squares Problems*, Philadelphia: SIAM, 1996.

-
- [11] BORGES, L. S.; BAZÁN, F. S. V.; CUNHA, M. C. Automatic stopping rule for iterative methods in discrete ill-posed problems, *Comp. Appl. Math.*, DOI 10.1007/s40314-014-0174-3, 2014.
- [12] CALVETTI, D.; LEWIS, B.; REICHEL, L. GMRES-type methods for inconsistent systems, *Linear Algebra and its Applications*, v. 316, n. 1, p. 157–169, 2000.
- [13] ELFVING, T.; NIKAZAD, T.; HANSEN, P. C. Semiconvergence and relaxation parameters for a class of SIRT algorithms, *Electronic Transactions on Numerical Analysis*, v. 37, p. 321–336, 2010.
- [14] ENGL, H. W.; HANKE, M; NEUBAUER, A. *Regularization of Inverse Problems*, Dordrecht: Kluwer Academic Publishers, 1996.
- [15] FIERRO, R. D. et al. Regularization by truncated total least squares, *SIAM J. Sci. Comput.*, v. 18, n. 4, p. 1223–1241, 1997.
- [16] GOLUB, G. H.; VAN LOAN, C. F. *Matrix Computations*, 3. ed., Maryland: Johns Hopkins University Press, 1996. 694 p.
- [17] HANKE, M. *Conjugate gradient type methods for ill-posed problems*, Harlow, UK: Longman, 1995.
- [18] HANKE M.; HANSEN, P. C. Regularization methods for large-scale problems, *Surv. Math. Ind.*, v. 3, p. 253–315, 1993.
- [19] HANSEN, P. C. Analysis of discrete ill-posed problems by means of the L-curve. *SIAM Rev.*, v. 34, p. 561–580, 1992.
- [20] HANSEN, P. C. *Rank-Deficient and Discrete Ill-Posed Problems*. Philadelphia: SIAM, 1998. 247 p.
- [21] HANSEN, P. C. Regularization Tools: A MATLAB package for analysis and solution of discrete ill-posed problems. *Numerical Algorithms*, v. 6, p. 1–35, 1994.
- [22] HANSEN, P. C. *Discrete Inverse Problems: Insight and Algorithms*. 2nd. ed. Philadelphia: SIAM, 2010. 213 p.
- [23] HANSEN, P. C. The discrete Picard condition for discrete ill-posed problems, *BIT*, v. 30, p. 658–672, 1990.
- [24] HESTENES, M. R.; STIEFEL, E. Methods of Conjugate Gradients for Solving Linear Systems, *J. Res. Nat. Bureau Standards*, v. 49, n. 6, p. 409–436, 1952.

- [25] KILMER, M. E.; O'LEARY D. P. Choosing regularization parameters in iterative methods for ill-posed problems, *SIAM J. Matrix Anal. Appl.*, v. 22, n. 4, p. 1204–1221, 2001.
- [26] KIRSCH, A. *An Introduction to the Mathematical Theory of Inverse Problems*, Applied Mathematical Sciences, Vol. 120, New York: Springer, 1996.
- [27] MEYER, C. D. *Matrix Analysis and Applied Linear Algebra*. Philadelphia: SIAM, 2000. 718 p.
- [28] MOORE, E. H. On the reciprocal of the general algebraic matrix. *Bull. Amer. Math. Soc.*, v. 26, p. 394–395, 1920.
- [29] MOROZOV, V. A. On the solution of functional equations by the method of regularization, *Soviet Math. Dokl.*, v. 7, p. 414–417, 1966.
- [30] NAGY, J. G.; PALMER, K.; PERRONE, L. Iterative methods for image deblurring: a Matlab object-oriented approach, *Numerical Algorithms*, v. 36, p. 73–93, 2004.
- [31] NATTERER, F. *The Mathematics of Computerized Tomography*. New York: Wiley, 1986. 226 p.
- [32] PAIGE, C. C.; SAUNDERS, M. A. Solution of sparse indefinite systems of linear equations, *SIAM Journal on Numerical Analysis*, v. 12, n. 4, p. 617–629, 1975.
- [33] PAIGE, C. C.; SAUNDERS, M. A. LSQR: An algorithm for sparse linear equations and sparse least squares, *ACM Trans. Math. Softw.*, v. 8, n. 1, p. 43–71, 1982.
- [34] PAN, V.Y. Newton's iteration for matrix inversion, advances and extensions, *Vadim Olshevsky, Eugene Tyrtyshnikov (Eds.), Matrix Methods: Theory, Algorithms and Applications*, World Scientific Publishing, p. 364–381, 2010.
- [35] PAN, V. Y.; SCHREIBER, R. An improved Newton iteration for the generalized inverse of a matrix, with applications. *SIAM J. Sci. Stat. Comput.*, v. 12, n.5, p. 1109–1130, 1991.
- [36] PENROSE, R. A generalized inverse for matrices. *Proc. Cambridge Phil. Soc.*, v. 51, p. 406–413, 1955.

- [37] PETKOVIĆ, M. D.; STANIMIROVIĆ, P. S. Iterative method for computing the Moore-Penrose inverse based on Penrose equations, *J. Comput. Appl. Math.*, v. 235, p. 1604–1613, 2011.
- [38] PETRYSHYN, W.V. On generalized inverses and on the uniform convergence of $(I - \beta K)^n$ with application to iterative methods, *J. Math. An. and App.*, v. 18, 417–439, 1967.
- [39] REGIŃSKA, T. A regularization parameter in discrete ill-posed problems, *SIAM J. Sci. Comput.*, v. 17, n. 3, p. 740–749, 1996.
- [40] SAAD, Y.; SCHULTZ, M. GMRES: A Generalized Minimum Residual Algorithm for Solving Nonsymmetric Linear Systems., *SIAM J. Scientific and Stat. Comp.*, v. 7, 856–869, 1986.
- [41] SILVA NETO, A. J.; MOURA NETO, F. D. *Problemas Inversos: Conceitos fundamentais e aplicações*, Rio de Janeiro: UERJ, 2005.
- [42] SCHREIBER, R. Computing Generalized Inverses and Eigenvalues of Symmetric Matrices Using Systolic Arrays. *Computing Methods in Applied Science and Engineering*, R. Glowinski and J.-L. Lions, editors, North Holland, Amsterdam, 1984.
- [43] SCHULTZ, G. Iterative Berechnung der Reciproken Matrix, *Z. Angew. Meth. Mech.*, v. 13, p. 57–59, 1933.
- [44] SÖDERSTRÖM, T; STEWART, G. W. On the numerical properties of an iterative method for computing the Moore-Penrose generalized inverse, *SIAM J. Numer. Anal.*, v.11, n. 1, p. 61–74, 1974.
- [45] TIKHONOV, A. N. Solution of incorrectly formulated problems and the regularization method, *Soviet Math. Dokl.*, v. 4, p. 1035–1038, 1963.
- [46] VOGEL, C. R. *Computational Methods for Inverse Problems*, Frontiers in Applied Mathematics, Philadelphia: SIAM, 2002.
- [47] ZOBLEC, S. On computing the best least squares solutions in Hilbert space, *Rend. Circ. Mat. Palermo*, v. 25, n. 3, p. 256–270, 1977.