MÉTODOS NUMÉRICOS - LISTA DE EXERCÍCIOS III

SÉRGIO CORDEIRO

SUMÁRIO

1. Sobre SVD: apresente modelagem matemática para calculo; apresente aplicações práticas; escolha uma aplicação associada à compressão de matrizes e faça a decomposição, a compressão e a analise o número de condicionamento antes e depois da compressão. Comente e analise todos os resultados.

A aplicação imediata da decomposição SVD é a solução mais fácil de um sistema linear, pois $\mathbf{U}$ e $\mathbf{V}$ são matrizes unitárias, por isso suas transpostas são as inversas; $\mathbf{S}$, por sua vez, é diagonal, e encontrar sua inversa é trivial. Portanto:

$$\begin{aligned} \mathbf{Ax} = \mathbf{b} \implies \qquad \mathbf{x} &= \mathbf{A}^{-1}\mathbf{b} \\ &= \left(\mathbf{USV}^{(T)} \right)^{-1} \mathbf{b} \\ &= \mathbf{VS}^{-1}\mathbf{U}^{(T)}\mathbf{b} \end{aligned}$$

Uma segunda aplicação é a compressão da matriz, que pode ser expressa pelo produto $\hat{\mathbf{U}}\hat{\mathbf{S}}\hat{\mathbf{V}}^{(T)}$, com as matrizes $\hat{\mathbf{U}}$, $\hat{\mathbf{S}}$ e $\hat{\mathbf{V}}^{(T)}$ derivadas das originais por eliminação de colunas menos significativas, correspondentes a valores singulares de menor valor absoluto.
Uma terceira aplicação é a diminuição do número de condicionamento da matriz, através da eliminação dos valores singulares com menor valor absoluto.

A compressão de matrizes por meio da decomposição SVD usa a técnica conhecida como *low-rank approximation*: após encontrarem-se os n autovalores, selecionam-se os m maiores; todas linhas de $\mathbf{U}$ e $\mathbf{V}$ são mantidas, mas apenas as m colunas que correspondem aos maiores autovalores dessas matrizes; quanto a $\mathbf{S}$, são selecionadas tanto as linhas quanto as colunas que correspondem aos autovalores mais importantes. O resultado é uma matriz C que tem o mesmo tamanho da matriz original, mas apenas m autovalores.
A manutenção dos maiores autovalores faz com que a maior parte da variância original seja preservada, e com ela, a informação relevante. O número de condição, por sua vez, é bastante melhorado.

A aplicação escolhida foi a compressão de imagens em preto e branco, com e sem introdução de ruído. As imagens originais foram obtidas no banco de dados do Departamento de Engenharia Elétrica e de Computação do Instituto Politécnico da Universidade de Nova York.
O programa **`exercmat.c`**, em anexo, escrito em C, lê uma matriz em

disco e comprime-a pelo método de decomposição SVD, calculando o número de condicionamento antes e depois da compressão. Basta digitar:

```
exercmat 20 n
```

onde n é o tamanho da matriz (n x n). Ele também calcula o número de operações em ponto flutuante necessário para cada etapa. O próprio programa tenta descobrir qual é a maior taxa de compressão possível sem que resulte perda expressiva de dados.
O algoritmo é totalmente genérico, e pode ser aplicado diretamente a uma matriz que representa uma imagem. Para este exercício, experimentou-se com alguns valores diferentes para a taxa de compressão.
O algoritmo usa o método de Jacobi para encontrar os autovalores e autovetores da matriz $\mathbf{A}^{(T)}\mathbf{A}$, o que não é uma solução ótima.
Os resultados obtidos são mostrados e tabelados a seguir. Foi usada apenas precisão simples, de forma a favorecer a ocorrência de erros de arredondamento.

n		**nome**	**Número de condição**		**custo** [1]		**iteraçõ**
antes	**depois**		**antes**	**depois**	**decomposição**	**¨compressão**	
512	347	Lena	38.146271	1.860093	68728250624	521	406
	240			1.495879			
	149			1.273939			
	71			1.142361			
512	336	Bárbara	23.812378	1.879261	68728250624	521	348
	232			1.471599			
	145			1.275514			
	68			1.138802			

Imagem original: Lena com 512 valores singulares

Imagem original: Bárbara com 512 valores singulares

Imagem comprimida: Lena com 347 valores singulares

Imagem comprimida: Bárbara com 336 valores singulares

Imagem comprimida: Lena com 240 valores singulares

Imagem comprimida: Bárbara com 232 valores singulares

Imagem comprimida: Lena com 149 valores singulares

Imagem comprimida: Bárbara com 145 valores singulares

Imagem comprimida: Lena com 71 valores singulares

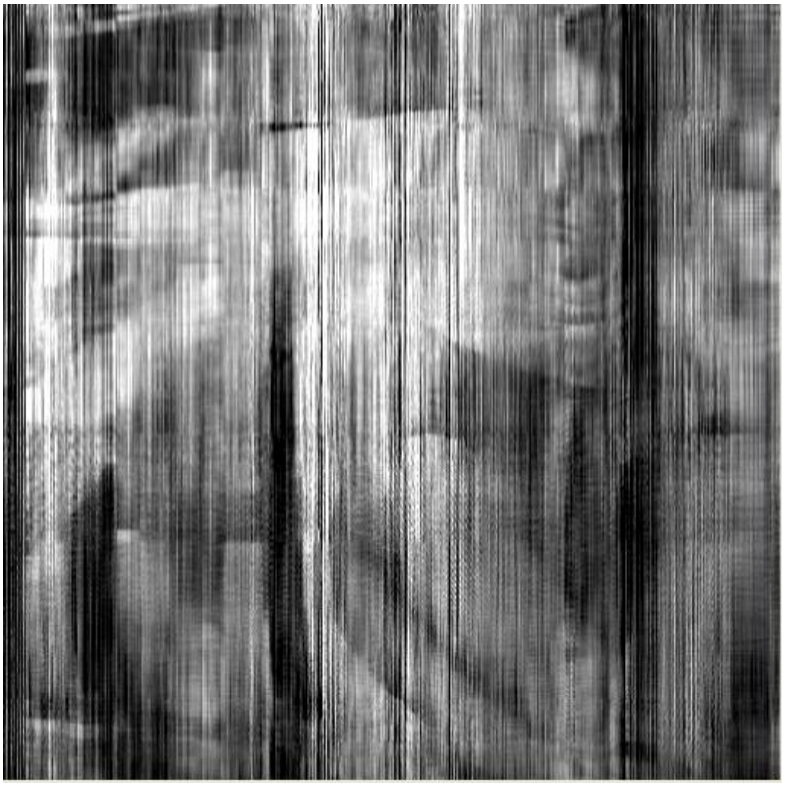
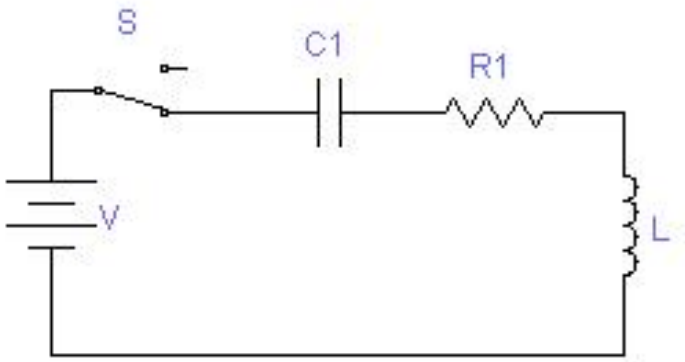


Imagem comprimida: Bárbara com 68 valores singulares

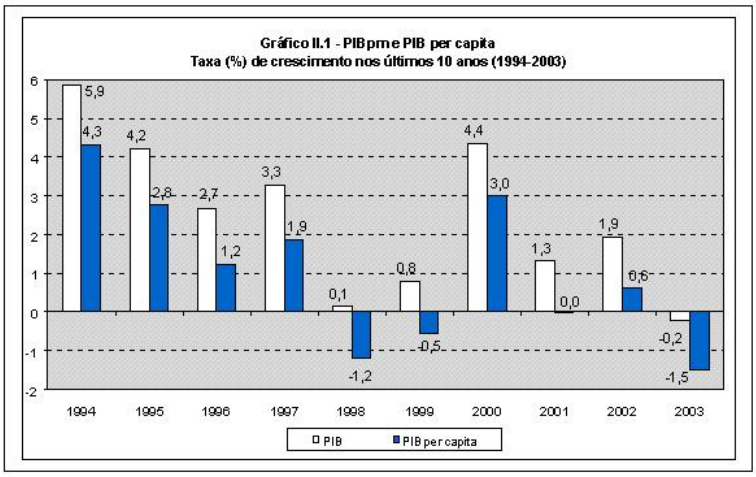
Para a pronunciada compressão obtida, as imagens mostram que a informação importante foi preservada.

2. Simule (utilizando qualquer software) o circuito da figura 1 e obtenha a tabela de valores (t (seg) e I (A)). Resolva analiticamente o circuito. Faça o ajuste da função utilizando regressão (avalie a que melhor se adeque a esse problema). Plote em um gráfico os valores simulados, calculados (solução analítica) e os valores obtidos a partir da regressão. Escolha os valores para os elementos de forma que a resposta seja oscilatória. Comente os resultados.



3. Para os Tópicos: Interpolação de Hermite, Interpolação com Spline Cúbico e Extrapolação faça:
   - Descrição de um problema real;
   - Modelagem matemática e numérica (código em anexo);
   - Resultados e conclusões

4. O gráfico a seguir apresenta as variações do PIB e PIB/percapita.



Para esses dados (PIB e PIB per capita) faça:

1. Encontre o polinômio interpolador que melhor represente essa função (PI de maior grau possível) (Use interpolação);
2. Plote o diagrama de dispersão e o PI encontrado;
3. Comente todos os resultados

O programa **`exercmat.c`**, em anexo, escrito em C, lê uma tabela em disco e encontra o polinômio interpolador com o maior grau possível. Basta digitar:

```
exercmat 18 n
```

onde n é o tamanho da tabela (n x 2). Ele também calcula o número de operações em ponto flutuante necessário para cada etapa.
Os coeficientes encontrados para o PIB foram:
$a_9 = 5.900000$, $a_8 = 24.808952$, $a_7 = -60.370991$, $a_6 = 49.870708$, $a_5 = -19.498470$, $a_4 = 3.804901$, $a_3 = -0.311178$, $a_2 = -0.005871$, $a_1 = 0.002509$, $a_0 = -0.000111$
A expressão do polinômio interpolador é:
$y = \sum_{i=0}^{9} a_i x^i$
com $x = ano - 1994$. Os coeficientes encontrados para o PIB per capita foram:
$a_9 = 4.300000$, $a_8 = 26.969584$, $a_7 = -64.837807$, $a_6 = 53.848827$, $a_5 = -21.452248$, $a_4 = 4.385204$, $a_3 = -0.417683$, $a_2 = 0.005891$, $a_1 = 0.001795$, $a_0 = -0.000093$.

1. ANEXOS

Os seguintes arquivos constam do anexo (arquivo **`exercmat1.zip`**):

- arquivo fonte em C **`exercmat.c`**
- arquivos fontes em MATLAB:
  - **`gerachol.m`**: problema 2
  - **`geradsis.m`**: problema 5
  - **`geraL.m`**: problema 11
- arquivos de dados:
  - S*n*: problemas 1, 3 e 4
  - C*n*: problemas 2, 7 e 8
  - D*n*: problemas 5 e 6

REFERÊNCIAS

[FAH 2013] Folding@home, Pande Lab, Stanford University, **FAQ: FLOPS**. Disponível em https://folding.stanford.edu/home/faq/faq-flops/, acesso em 17/03/2016.

[FASSHAUER 2006] Greg FASSHAUER, **Numerical Linear Algebra/Computational Mathematics I**: Illinois Institute of Technology, 2006. Disponível em http://www.math.iit.edu/~fass/477577_Chapter_16.pdf, acesso em 25/03/2016.

[GATR 2015] GATR Technologies, **2.4m Antenna System Datasheet**. Disponível em http://www.gatr.com/products/2-4-antenna-system, acesso em 21/03/2016.

[INTEL 2016] INTEL Corporation, **Intel 64 and IA-32 Architectures Optimization Reference Manual**, 2016, Tab. 15-3, pag. 15-9 a 15-10.

[KAWAKAMI 1989] Y. KAWAKAMI, H. HOJO e M. UEBA, **Control Design of an Antenna Pointing Control System with Large On-board Reflectors**, *in* T. NISHIMURA - Automatic Control in Aerospace 1989: Selected Papers from The IFAC Symposium.

[MOON 2008] Francis C. MOON, **Applied Dynamics: With Applications to Multibody and Mechatronic Systems**, 2nd Ed., 2008, Weinheim, Wiley-VCH, ISBN 978-3-527-40751-4, Chap. 5, pp. 228 a 231.

[NASA 1980] National Aeronautics and Space Administration, **Satellite Power System (SPS) Antenna Pointing Control**. Disponível em http://www.nss.org/settlement/ssp/library/NASACR3350-AntennaPointingControl.pdf, acesso em 21/03/2016.

[PHILLIPS 1995] Charles L. PHILLIPS e H. Troy NAGLE, **Digital Control System Analysis and Design**, Prentice Hall, Englewood Cliffs, 1995, 3 rd Ed..

[TAMAGAWA 2016] TAMAGAWA Seiki Corporation, **TBL-iIVseries AC Servomotor**. Disponível em http://www.tamagawa-seiki.com/pdf/download/1699N1EJ_shusei.pdf, acesso em 21/03/2016.

[TERAN 2013] Joseph M. TERAN, **Applied Numerical Linear Algebra Course Notes: Week 5**: Mathematics Department, UCLA, 2013. Disponível em http://www.math.ucla.edu/~jteran/270c.1.13s/notes_wk6.pdf, acesso em 25/03/2016.

[USMANI 1994] Riaz A. USMANI, **Inversion of a Tridiagonal Jacobi Matrix**: Department of Applied Mathematics, University of Manitoba, 1994. Disponível em http://ac.els-cdn.com/0024379594904146/1-s2.0-0024379594904146-main.pdf?_tid=1dc5a5dc-f2f2-11e5-9bd3-00000aacb35f&acdnat=1458955920_5a0cc77d01e9dbdd1dfc2d1e0b8627b1, acesso em 25/03/2016.

[VANDENBERGHE 2012] L. VANDENBERGHE, **Complexity of matrix algorithms**: EE103 - Applied Numerical Computing (Fall 2011-12), UCLA. Disponível em http://www.seas.ucla.edu/~vandenbe/103/lectures/flops.pdf, acesso em 26/03/2016.

[VANDEBRIL 2004] Raf VANDEBRIL, **Semiseparable matrices and the symmetric eigenvalue problem**: Katholieke Universiteit Leuven, Faculteit Toegepaste Wetenschappen - Departement Computerwetenschappen, 2004. Disponível em http://www.cs.kuleuven.be/publicaties/doctoraten/tw/TW2004_03.pdf, acesso em 25/03/2016.

[WEISSTEIN 2016] Eric W. WEISSTEIN, **Positive Definite Matrix *MathWorld***. Disponível em http://mathworld.wolfram.com/PositiveDefiniteMatrix.html, acesso em 17/03/2016.

Programas testados com **Octave** 4.0.0 e **MinGW** C 4.8.2:
https://www.gnu.org/software/octave/
https://www.mingw.org
Texto formatado com **pdflatex** em ambiente **MiKTeX** 2.9:
http://miktex.org/download/