CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
MÉTODOS NUMÉRICOS COMPUTACIONAIS
Lista de exercícios 01

Thiago Henrique Gonçalves Mello - 201612060188 - thiagohgmello@gmail.com

Questão 1)

Para resolução da primeira questão da lista de exercícios, foi desenvolvida uma rotina em linguagem C. Primeiramente, criou-se um arquivo *header* nomeado "matrix.h" com o intuito de separar as rotinas desenvolvidas para cada tipo de problema. Nele, foram inseridas as funções responsáveis pela verificação da possibilidade de multiplicação entre matrizes, inicializador de matrizes, multiplicador e uma rotina específica para desalocar matrizes uma vez que estas são criadas dinamicamente.

A verificação é feita através da comparação entre o número de colunas da primeira matriz na multiplicação e o número de linhas da segunda. O tamanho dos elementos multiplicados é definido pelo usuário exigindo então uma requisição de dados inicialmente.

Com os tamanhos especificados, passa-se então para a multiplicação das matrizes, caso a operação possa ser realizada. Uma rotina em MATLAB foi feita com o intuito de comparar os resultados oriundos da rotina em C com precisões de ponto flutuante (*float*) e *double*. Para explorar de forma mais completa as vantagens e desvantagens de determinado tipo de variável, as entradas de cada uma das matrizes foram determinadas a partir de uma função matemática, a saber:

$$a_{ij} = \log(\pi^x)\,, \tag{1}$$

em que x é uma variável semi-aleatória criada através da função "rand" da biblioteca *stdlib*.

A norma implementada para comparação de resultados foi a norma de Frobenius, uma vez que esta expressa de forma similar a média geométrica dos elementos da matriz.

A título de comparação entre as duas precisões, vários tamanhos de matrizes distintos foram gerados aleatoriamente e então comparadas com o resultado oriundo da rotina em MATLAB.

Com a precisão *float*, 5 tamanhos distintos de matrizes foram apresentados com os erros da norma de Frobenius expressos na Tabela I.

Tabela I. Erros percentuais sobre as normas de Frobenius para multiplicação de matrizes considerando precisão float. Comparação entre programa C e resultado do MATLAB.

Tamanho das Matrizes ($\mathrm{A} \cdot \mathrm{B} = \mathrm{C}$)			Erro (%)
A	B	C	
[500][400]	[400][700]	[500][700]	$1{,}9450 \cdot 10^{-5}$
[70][96]	[96][78]	[70][78]	$2{,}2318 \cdot 10^{-6}$
[1000][800]	[800][792]	[1000][792]	$4{,}5576 \cdot 10^{-5}$
[100][100]	[100][100]	[100][100]	$2{,}2246 \cdot 10^{-6}$
[9846][654]	[654][6546]	[9846][6546]	$3{,}9493 \cdot 10^{-5}$

De forma similar, o mesmo procedimento com matrizes do mesmo tamanho, porém com possibilidade de serem diferentes, foi realizado considerando precisão estendida (*double*). Os resultados estão na Tabela II.

Tabela II. Erros percentuais sobre as normas de Frobenius para multiplicação de matrizes considerando precisão double. Comparação entre programa C e resultado do MATLAB.

Tamanho das Matrizes $(\mathrm{A} \cdot \mathrm{B} = \mathrm{C})$			Erro (%)
A	B	C	
$[500][400]$	$[400][700]$	$[500][700]$	$8{,}4449 \cdot 10^{-13}$
$[70][96]$	$[96][78]$	$[70][78]$	0
$[1000][800]$	$[800][792]$	$[1000][792]$	$1{,}0854 \cdot 10^{-12}$
$[100][100]$	$[100][100]$	$[100][100]$	$4{,}3359 \cdot 10^{-14}$
$[9846][654]$	$[654][6546]$	$[9846][6546]$	$2{,}3115 \cdot 10^{-10}$

Comparando os dados, vê-se que, mesmo apresentando erros baixos, variáveis *float* cometem erros cerca de um milhão de vezes superiores aos casos em que se utiliza *double*. Para matrizes maiores como por exemplo a última experimentada, os erros absolutos com números de ponto flutuante chegaram na casa de milhares de unidades, o que pode representar grandezas demasiadamente grandes para problemas específicos.

Outro ponto que merece ser ressaltado é o efeito da propagação do erro. Para o caso de multiplicação, houve apenas dois pontos de combinação de incertezas: 1) no cálculo do elemento a_{ij} em (1) e 2) no produto entre os elementos, entretanto, caso houvesse mais operações subsequentes, a propagação ocasionaria problemas maiores, como será visto em outras questões da lista.

A desvantagem em se utilizar tamanhos maiores de variáveis ficou explícita ao avaliar o arquivo .txt gerado como saída do produto. Para o caso de precisão *double*, o arquivo teve cerca de 1 GB de memória, o que impossibilitava até sua abertura em programas para edição deste tipo de extensão.

Portanto, o que se pode concluir neste problema é que, caso não seja necessária uma precisão exorbitante, variáveis do tipo *float* são interessantes.

Questão 2)

No caso de solução de sistemas lineares, tamanhos menores de matrizes foram implementados, uma vez que são geradas aleatoriamente, fazendo com que a chance de surgimento de matrizes singulares seja consideravelmente elevada. Com base nisto, sistemas $4x4$ tanto superiores quanto inferiores foram utilizados. Os resultados da comparação entre as normas de Frobenius para o sistema diagonal superior utilizando variáveis do tipo *float* e MATLAB estão na Tabela III enquanto que para sistemas diagonais inferiores estão na Tabela IV. De forma similar, as Tabelas V e VI apresentam os comparativos quando as variáveis são do tipo *double*.

Tabela III. Erros percentuais sobre as normas de Frobenius para solução de sistemas lineares triangulares superiores considerando precisão float. Comparação entre programa C e resultado do MATLAB.

Execução $(4x4)$	Erro (%)
1	$2{,}2821 \cdot 10^{-7}$
2	$2{,}6167 \cdot 10^{-6}$
3	$8.1015 \cdot 10^{-6}$
4	$2{,}5107 \cdot 10^{-6}$
5	$2{,}8405 \cdot 10^{-7}$

Tabela IV. Erros percentuais sobre as normas de Frobenius para solução de sistemas lineares triangulares inferiores considerando precisão float. Comparação entre programa C e resultado do MATLAB.

Execução $(4x4)$	Erro (%)
1	$1{,}4589 \cdot 10^{-5}$
2	$1{,}6580 \cdot 10^{-5}$
3	$4{,}0311 \cdot 10^{-6}$
4	$7{,}7075 \cdot 10^{-8}$
5	$2{,}4263 \cdot 10^{-6}$

Tabela V. Erros percentuais sobre as normas de Frobenius para solução de sistemas lineares triangulares superiores considerando precisão double. Comparação entre programa C e resultado do MATLAB.

Execução $(4x4)$	Erro (%)
1	$3{,}6179 \cdot 10^{-14}$
2	$6{,}3039 \cdot 10^{-14}$
3	$6{,}4135 \cdot 10^{-14}$
4	0
5	$3{,}6202 \cdot 10^{-14}$

Tabela VI. Erros percentuais sobre as normas de Frobenius para solução de sistemas lineares triangulares superiores considerando precisão double. Comparação entre programa C e resultado do MATLAB.

Execução $(4x4)$	Erro (%)
1	$3{,}6404 \cdot 10^{-14}$
2	$1{,}9950 \cdot 10^{-13}$
3	0
4	$3{,}7193 \cdot 10^{-14}$
5	$1{,}7715 \cdot 10^{-14}$

Mais uma vez, como esperado, percebe-se uma precisão consideravelmente maior quando se utiliza variáveis do tipo *double*. Como a aplicação foi pequena (solução de sistemas lineares 4x4), a diferença no tempo de execução não foi perceptível. Apesar disso, caso se tenha sistemas de mais alta ordem, operações com maior precisão apresentam maior custo computacional.

Outro ponto que merece destaque é o caso de propagação de erros. Para sistemas lineares, como a determinação de uma variável pode depender diretamente da definição de outras, a propagação de erros pode ser considerável à medida que se eleva o tamanho do sistema. Com isso, deve-se conhecer bem o problema de forma a escolher o menor custo de implementação, seja ele de tempo ou de espaço.

Questão 3)

A implementação da pivotação foi o problema que mais demandou tempo dentre aqueles apresentados neste trabalho. Apesar disso, alguns pontos são de suma importância de serem ressaltados: 1) quando se trata de problemas essencialmente numéricos, igualdade não se mostram bem-vindas em comparações; 2) processos de operação contínua entre elementos da matriz aumentam a propagação de erros e 3) operações intermediárias devem ser feitas com a maior precisão dentro daquelas disponíveis.

O primeiro ponto que mereceu destaque foi oriundo de um problema deparado durante o desenvolvimento do algoritmo. Por serem comparações numéricas, aproximações são inerentes ao processo e, com elas, os erros numéricos também o são.

No caso estudado, comparações entre números e o valor nulo retornavam falso mesmo quando o processo era, na realidade, verdadeiro. Isso se dava porque havia propagação de erro inerente aos cálculos. Mesmo incorporando variáveis de maior precisão, o problema tende a existir, uma vez que dois valores nunca serão comparados em toda sua magnitude. Por ser algo inerente à programação, tal erro não foi atribuído à escolha de variáveis, sendo então presente nos dois casos abordados.

O outro ponto ressaltado fica ainda mais evidente quando se tem sucessivas operações com os mesmos elementos. Isto porque um número que essencialmente é uma dízima, seja ela periódica ou não, será sempre representado por um valor finito. Ao ser alterado de qualquer forma, o erro do truncamento também é alterado na mesma magnitude, tornando-o mais significativo.

Por fim, o terceiro ponto se refere às variáveis como de somatório, que auxiliam no resultado final, mas existem apenas por limitação do ambiente de programação. Com isso, diminui-se a chance de propagação de erros.

Assim como nos casos anteriores, o mesmo programa foi executado cinco vezes com determinada precisão e então esta foi alterada e o programa executado cinco vezes. A geração de valores das matrizes segue a mesma regra expressa em (1) e por ser aleatória, possui extrema limitação na geração de matrizes não singulares e, consequentemente, sistemas lineares possíveis e determinados. Como saídas, tem-se as variáveis de interesse de $\mathrm{Ax} = \mathrm{b}$, a saber, os valores de x, o determinante da matriz A e a norma euclidiana de x. A Tabela VII expressa a comparação entre o mesmo sistema solucionado utilizando rotina em C e em MATLAB para variáveis do tipo *float* e a Tabela VIII para variáveis do tipo *double*.

Tabela VII. Erros percentuais sobre as normas euclidianas e o determinante para solução de sistemas lineares pelo método de Gauss com pivotação considerando precisão float. Comparação entre programa C e resultado do MATLAB.

Execução $(4x4)$	Erro determinante (%)	Erro norma (%)
1	$7{,}7808 \cdot 10^{-6}$	$8{,}1966 \cdot 10^{-6}$
2	$2{,}8073 \cdot 10^{-5}$	$1{,}2081 \cdot 10^{-5}$
3	$2{,}0263 \cdot 10^{-5}$	$8{,}4137 \cdot 10^{-7}$
4	$3{,}2501 \cdot 10^{-6}$	$1{,}6788 \cdot 10^{-6}$
5	$4{,}8760 \cdot 10^{-6}$	$7{,}4837 \cdot 10^{-6}$

Tabela VIII. Erros percentuais sobre as normas euclidianas e o determinante para solução de sistemas lineares pelo método de Gauss com pivotação considerando precisão double. Comparação entre programa C e resultado do MATLAB.

Execução $(4x4)$	Erro determinante (%)	Erro norma (%)
1	$3{,}3444 \cdot 10^{-8}$	$3{,}4571 \cdot 10^{-14}$
2	$3{,}0173 \cdot 10^{-9}$	$3{,}5830 \cdot 10^{-14}$
3	$1{,}9280 \cdot 10^{-8}$	0
4	$3{,}8421 \cdot 10^{-9}$	$1{,}3942 \cdot 10^{-13}$
5	$1{,}1268 \cdot 10^{-9}$	$3{,}2833 \cdot 10^{-14}$

Como esperado mais uma vez, os erros se mostraram consideravelmente maiores quando as simulações foram feitas com variáveis do tipo *float*. Os erros elevados na Tabela VIII quando se compara os valores dos determinantes é explicado pela precisão com a variáveis é impressa na tela. Para o caso dos determinantes, os valores foram dados com precisão de seis casas decimais. Mesmo com isso, percebe-se uma diferença entre os dois tipos de variáveis de cerca de 100 vezes, explicitando ainda mais as consequências da propagação de erros de aproximação nas operações.

Com isso, assim como foi dito nos casos anteriores, deve-se possuir bem definida a relação entre custo computacional e precisão demandada pelo projeto.

Questão 4)

De forma geral, os sistemas que descrevem algum comportamento da natureza são essencialmente não lineares, entretanto, é comum buscar a análise de desempenho apenas em uma determinada faixa de operação, tornando possível assim a linearização do problema. Com isso, todas as interações da modelagem são expressas através da relação $Ax = b$ e suas variantes. Aqui, serão apresentadas apenas aplicações de sistemas lineares em eletromagnetismo, controle, SEP e modelagem de sistemas, não ficando restrito a estes toda a usabilidade das características aqui descritas.

No caso de problemas eletromagnéticos, aplicações comuns são na solução das equações de Maxwell através de métodos numéricos chamados de *mesh*, sendo que cada um apresenta matrizes com características distintas, sejam elas esparsas ou densas. Independente disso, os métodos de forma geral resolvem um problema na forma $Ax = b$, em que b é um vetor de excitação e A apresenta as características do meio quando submetido às ondas eletromagnéticas. Pode-se citar mais especificamente o método dos momentos, método dos elementos finitos e método das diferenças finitas. As metodologias utilizadas para solucionar estes problemas são as mais rebuscadas, uma vez que se pode obter matrizes quase singulares.

Outro ramo do eletromagnetismo no qual há presença constante de sistemas lineares é a análise de circuitos elétricos, no qual geralmente tem-se como objetivo determinar as tensões e correntes em determinado circuito. Pode-se citar o método das correntes de malha e tensões de nó. De maneira mais genérica, as leis de Kirchhoff das tensões e correntes que norteiam os estudos de circuitos em baixa frequência são aplicações diretas da metodologia de sistemas lineares. Nestes casos, dependendo da quantidade de paralelismo, efeitos mútuos e compartilhamento de correntes, as matrizes A podem ser mais ou menos densas. O vetor b depende diretamente da quantidade de partes ativas no circuito analisado. É comum implementar métodos simples de solução, uma vez que as matrizes costumam ser bem definidas.

A área de controle possui aplicações intimamente relacionadas à sistemas lineares, com um ramo de estudos específico para este tipo de modelagem. Controle de processos industriais pode ser um grande exemplo, no qual deseja-se garantir que o sistema funcione em certo ponto característico. Para isto, a relação entre as variáveis de entrada e a as variáveis de saída pode ser aproximada por um problema da forma $Ax = b$. Nestes casos, a forma das matrizes pode ser a mais diversa, com dependência intensa entre todos os parâmetros ou um nível de desacoplamento elevado, tornando a matriz mais esparsa. A partir disto, as técnicas de solução podem ser as mais simples até as mais robustas. Exemplos são controle de braços robóticos, auto fornos, esteiras, máquinas, entre outros processos.

Ainda em controle, outra aplicação comum de SL (sistemas lineares) é em espaço de estados. Neste ramo, as equações são tipicamente diferenciais, entretanto, com auxílio de transformadas como de Laplace tem-se SL da forma típica. Exemplos são controle de motores, cujas malhas são mais densas, entretanto não costumam apresentar muitos termos; projeto de filtros, com as ordens mais diversas; e controle de braços robóticos, com matrizes de transformadas e com seu grau de densidade dependendo do grau de liberdade do projeto. Com isso, as técnicas de solução são as mais diversas.

Em SEP, SL são comuns em várias partes. Entre elas, modelagem de fluxo de potência, no qual deseja-se compreender curtos-circuitos, trocas de potência, entre outros fatores, através dos dados elétricos das subestações (SEs) e das linhas de transmissão (LTs). Nestes casos, a modelagem pode ser dada pela matriz de impedância ou de admitância entre cada uma das SEs, sendo que a matriz de impedância costuma ser mais esparsa enquanto que matrizes de admitância são cheias, porém com muitos termos próximos de zero em sistemas grandes. Com isso, é comum implementar o método de Newton-Raphson.

Outra utilização é na escolha de cabos para-raios das LTs, nas quais precisa-se conhecer a relação entre a tensão induzida pelas fases e a corrente circulante nos para-raios. Normalmente a matriz é pequena (5x5) e mais densa, utilizando sistemas simples de solução como eliminação de Gauss.

Análise de desempenho de LTs também utilizam SL, uma vez que resolvem um sistema elétrico de natureza distribuída (circuitos π). Nestes casos, as matrizes são mais esparsas, uma vez que a interferência entre parcelas muito distantes umas das outras é pequena.

Como último ponto ressaltado de SEP, pode-se citar a aplicação de SL em problemas de dimensionamento de aterramentos de torres de LTs. Nestes casos, a modelagem, muitas vezes feita através do método da distribuição de cargas, utiliza SL para relacionar a influência entre os circuitos série ou paralelos do modelo de aterramento. Da mesma forma como dito anteriormente, devido à distância elevada entre alguns circuitos π, a interferência entre eles é pequena e a matriz pode se tornar esparsa. As técnicas podem ser as mais simples até aquelas mais implementadas em sistemas quase singulares.

A área de modelagem é aquela na qual pode-se inserir todas as aplicações anteriores, uma vez que são modelos de um fenômeno. Mesmo assim, de forma distinta, a área de otimização linear é uma ainda não descrita neste trabalho. Processos de otimização de problemas como melhor rota de transporte visando menor gasto, maior fluxo de veículos por um semáforo, maior quantidade de material guardado em um container ou até mesmo o menor consumo de energia de uma planta industrial são exemplos de aplicação de modelos de SL. Nestes casos, os problemas são possíveis, entretanto indeterminados, possibilitando o grau de liberdade essencial para a otimização. Com isso, para sistemas menores, as matrizes costumam ser mais densas, mesmo assim, à medida que se eleva a quantidade de parâmetros de otimização, pode-se obter matrizes mais esparsas. Um método comum utilizado para solução de problemas com poucas variáveis de otimização é o Simplex.