E-fólio B | Folha de resolução para E-fólio

UNIDADE CURRICULAR: Computação Numérica

CÓDIGO: 21180

DOCENTE: Paulo Shirley

A preencher pelo estudante

NOME: Pedro Pereira Santos

N.º DE ESTUDANTE: 2000809

CURSO: Licenciatura em Engenharia Informática

DATA DE ENTREGA: 28/12/2023

TRABALHO / RESOLUÇÃO:

Pelo enunciado e pelas regras impostas, podemos afirmar que estas funções serão para ser aplicadas a matrizes quadradas invertíveis (matrizes quadradas não singulares), de ordem n. Posto isto, podemos simplificas os algoritmos, de modo a tomar em conta apenas a ordem da matriz, não necessitando de acompanhar a linha nem a coluna. Também, para os pivots, na eliminação de Gauss, podemos utilizar o algoritmo apresentado no manual indicado no PUC da UC, uma vez que este é simplificado para tais casos.

1.1

Nesta alínea, é solicitada uma função condm(A,p), que calcula o número de condição de uma dada matriz A, usando a norma $||.||_p$. Temos a condição de apenas ser possível utilizá-la caso p seja 1 ou infinito, retornando uma matriz vazia, em caso contrário. Temos que, a condição de uma matriz A é a norma dessa matriz multiplicada pela norma da sua inversa. Posto isto, teremos de calcular a norma, tendo a condição de que p seja 1 ou infinito.

A diferença entre a norma 1 e a norma infinito de uma matriz é que a norma 1 é o máximo das somas por colunas e a norma infinito é o máximo das somas por linhas. Posto isto, podemos aplicar o algoritmo de cálculo para a norma 1 de A, transpondo a matriz A, quando p é infinito, simplificando, assim o código.

```
%A diferença entre a norma 1 e inf é uma atuar sobre linhas e a outra sobre colunas
if(p == inf)            %por isso, verificamos se p=inf,
  A = A';               %utilizando a transposta em caso positivo
end                     %podendo assim reutilizar o algoritmo, sem mais condicionais
```

Após aplicar esta verificação e o algoritmo, obtemos a condição pela multiplicação das normas de A e inversa de A.

```
%onde norma = max do somatório das colunas de A
norma = max(aux);
normaInv = max(auxInv);

c = norma * normaInv;
```

1.2

Nesta alínea é solicitado a implementação de uma função elim_gausspt(A,b,tol), que utilizando o método de eliminação de Gauss com escolha parcial de pivot e a tolerância especificada pelo parâmetro de entrada “tol”, calcula o vetor solução x de equação lineares Ax=b, recendo, como parâmetro de entrada, a matriz A em “A” e o vetor coluna b em “b”.

Temos a condição, de que, se A é singular, temos de retornar um vetor vazio em x, imprimindo a informação de que A é singular. Para tal, é utilizado o parâmetro de tolerância “tol” para a verificação, no valor do pivot em questão em cada ciclo do algoritmo:

```
if(abs(pivot(1))<tol) %se o valor absoluto do pivot for inferior ao valor de tolerância
%quer dizer que essa matriz é singular, retornando com mensagem a informar o mesmo!
  printf("Matriz A singular!")

  x=[];%retorna também a matriz vazia
  return;
end
```

, e no valor do último pivot, em $a_{n,n}$, antes de iniciar a substituição inversa:

```
%Caso o pivot em A(n,n) seja inferior, em valor absoluto, à tolerância, assume-se que a matriz é singular
if(abs(A(n,n))<tol)
  printf("Matriz A singular!");
  x=[];%e retorna o vetor vazio
  return;
end
```

Caso a matriz não seja singular, prosseguimos para o algoritmo de substituição inversa, obtendo os valores do vetor coluna x, que correspondem ao vetor solução x, atrás mencionado.

```
%Valor de x(n) = b(n) - A(n,n) (Início da substituição inversa)
x(n)=b(n)/A(n,n);

%método de substituição inversa
for(i=n-1:-1:1)

  somatorio=0;%inicializa o valor do somatorio em 0

  for(j=i+1:n)%Somatório de a(i,j) * x(j)
    somatorio = somatorio + A(i,j)*x(j);
  end

  %função xi=(bi-somatorio(n,j=i+1) de aij*xj)/ai,i -> método de substituição inversa
  x(i)=(b(i)-somatorio)/A(i,i);

end
```

1.3

Para este script, temos alguns parâmetros especificados no enunciando, como o valor de tolerância ser igual a 10^{-13}, como criar uma matriz de ordem 15 e um vetor coluna de 15 linhas, ambos com valores aleatórios. Para tal, começamos por atribuir estes valores, já descritos, a variáveis, criando também mais algumas variáveis auxiliares para guardar valores obtidos, tais como, os valores da condição da matriz criada.

Para simular um input aleatório do valor de p a ser passado para a função de cálculo de condição, criei uma condição com a função randi(), que cria número aleatórios inteiros, dentro de um intervalo específico, neste caso dando apenas dois resultados diferentes, 1 ou 2, com a mesma probabilidade. Isto é utilizado para atribuir o valor 1 ou infinito a p.

```
%obter o valor de p=1 ou p=inf, com a mesma probabilidade
if(randi([1,2]) == 1)
  p = 1
else
  p = inf
end
```

Após isto, iniciamos o ciclo com 50 iterações, que em cada ciclo, é executado o pedido, variando o parâmetro k de 0,0 a 0.9:

- Calcular o número de condição da matriz A, com a função condm(A,p), criada em 1.1;
- Resolver Ax=b, com tol = 10^{-13}, utilizando a função elim_gausspt(A,b,tol), criada em 1.2
- Calcular a norma do erro $||Ax-b||_2$, onde, sendo Ax-b um vetor, é igual ao somatório dos valores absolutos deste vetor, sendo cada um deles elevado a 2, onde este somatório, após calculado, é elevado a ½. Temos, neste ponto, a condição de que se x estiver vazio, esta norma é infinito.
- Após estes três pontos serem executados, aplicamos a expressão apresentada A(1,:)= (1-k(i))*A(1,:) + k(i)*A(2,:); que aplica o parâmetro k, acima descrito, que aproxima a matriz A, progressivamente, a uma matriz singular.

Em cada iteração do ciclo descrito, são guardados os valores de condição da matriz e de norma do erro, para serem utilizados nos gráficos pedidos.

Estes gráficos, comparam a norma do erro com o número de condição. Na figura 1, vimos cada variável isolada, em cada iteração, onde podemos verificar que seguem o mesmo padrão. Na figura 2, temos o gráfico pedido, com as especificações pedidas, onde mostra o gráfico de “norma do erro vs num. condição", que podemos observar que segue uma relação linear.

Estes resultados, mostram a relação entre a precisão do cálculo, indicando a distância entre a solução obtida de a solução exata, analisando o valor do erro, e a sensibilidade do problema, ao analisar o número de condição, em relação às mudanças nos dados de entrada.

Como dito anteriormente, o gráfico mostras uma relação linear, com inclinação parecida à reta y=x, o que demonstra que existe uma relação entre o erro e o número de condição, que mostra uma relação e sensibilidade exponencial. Isto significa que o erro aumenta exponencialmente com o número de condição, mostrando que pequenas mudanças no número de condição, resultam em aumentos significativos no erro dos cálculos, podendo indicar um problema mal condicionado, ou seja, que pequenas perturbações nos dados de entrada levam a grandes perturbações nos dados de saída.

Logo, com este problema, obtemos uma má condição de problema.

Par de gráficos obtidos de uma execução do script: