Escola Politécnica da Universidade de São Paulo

MAP3121 - Métodos Numéricos e Aplicações

Exercício Programa 1

01/07/2021

Integrantes:

Bruno Carneiro Camara - 11257230

Leonardo Akira Shimabukuro - 9838053

São Paulo - Escola Politécnica

1° Semestre de 2021

Introdução:

Neste segundo exercício programa implementamos o algoritmo de Householder para a tridiagonalização de uma matriz simétrica qualquer no intuito de usar o algoritmo QR, desenvolvido no primeiro exercício programa, para determinar autovalores e autovetores. Com esses algoritmos em mãos fomos capazes de calcular as frequências e os modos de vibração de uma Treliça Plana específica, apresentada abaixo:

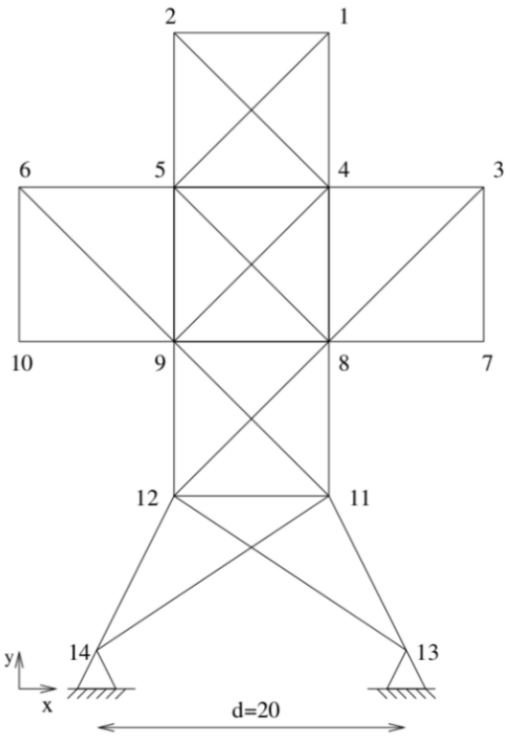

Figura 1 - Treliça Plana utilizada utilizada no EP

O algoritmo de Householder:

No enunciado do programa foram apresentadas duas maneiras de se fazer a transformação de Householder, uma calculada utilizando H_w como matrizes e outra fazendo o produto interno diretamente com w. No código desenvolvido foi utilizado a versão sem matrizes a mais, devido à eficiência computacional economizada. Enquanto que utilizando as matrizes temos um $O(n^2)$, sem matrizes temos um $O(n)$.

Para a execução do algoritmo foram criadas duas funções auxiliares principais, “multiplicaHwnEsquerda” e “multiplicaHwnDireita”. Com essas funções estaríamos simulando a multiplicação de Hwn por uma outra matriz, tanto pela esquerda, quanto pela direita.

Para não utilizar as matrizes Hwn foi necessário calcular os vetores w_n em todas as iterações a partir da fórmula abaixo:

$$w_n = a_n + \delta||a_n||e_n$$

Na qual delta representa a função signo do primeiro elemento de a_n e e_n é um vetor coluna com apenas o primeiro elemento valendo 1 e o restante valendo zero. O vetor an tem número de linhas igual à da coluna da iteração abaixo da diagonal principal. Por exemplo, o a_1, ou seja, o an na primeira iteração, pegaríamos todas as colunas à direita da coluna de mesmo número da iteração, com um número de linhas igual ao número de linhas abaixo da diagonal principal, como abaixo:

$$A = \begin{bmatrix} 2 & -1 & 1 & 3 \\ -1 & 1 & 4 & 2 \\ 1 & 4 & 2 & -1 \\ 3 & 2 & -1 & 1 \end{bmatrix}$$

Se tivéssemos que pegar an na segunda iteração, pegaríamos tudo que está a direita a partir do elemento abaixo da diagonal principal da segunda coluna, como abaixo:

$$A = \begin{bmatrix} 2 & -1 & 1 & 3 \\ -1 & 1 & 4 & 2 \\ 1 & 4 & 2 & -1 \\ 3 & 2 & -1 & 1 \end{bmatrix}$$

Com ele podemos utilizar a fórmula abaixo para calcular a multiplicação de Hw pela direita e pela esquerda da matriz de entrada:

$$H_w x = x - 2\frac{w \cdot x}{w \cdot w}w$$

Na qual o ponto representa a operação produto escalar entre dois vetores.

Na multiplicação pela esquerda o vetor x é parte da coluna da matriz original enquanto que na multiplicação pela direita x é parte das linhas da matriz original. O tamanho de x e de w depende da iteração e do tamanho da matriz original.

Para não haver repetição de cálculos devemos rebater a coluna da matriz resultante da multiplicação pela direita (HwnAHwn) na coluna correspondente à iteração.

Com todos os passos acima, o algoritmo de Householder fica:

1. Iniciar a matriz H^T como identidade
2. Iterar até as duas últimas colunas da matriz de entrada
    a. Obter w_n
    b. Obter HwnA, utilizando multiplicaHwnEsquerda
    c. Obter HwnAHwn, utilizando multiplicaHwnDireita
    d. Rebate a Coluna de HwnAHwn na Linha de HwnAHwn
    e. Calcula H^T com a função multiplicaHwnDireita

Como saída da função Householder obtemos a matriz tridiagonalizada HwnAHwn e a matriz H^T que é o produto das transformações de Householder.

Tarefas

Tarefa A

A primeira tarefa envolve calcular autovalores e autovetores de matrizes simétricas. Para isso, é necessário reduzir a matriz de entrada para sua forma tridiagonal simétrica utilizando o algoritmo de Householder. Desta forma, a matriz pode ser utilizada como entrada para o algoritmo QR com deslocamento espectral implementado no exercício anterior. Adicionalmente, é realizada uma verificação de $Av = \lambda v$ para validar os autovetores e autovalores obtidos. Por fim, verifica-se que a matriz de autovetores é ortogonal, multiplicando-a por sua transposta e obtendo uma matriz identidade.

A entrada da rotina desta tarefa recebe dados através da entrada padrão. A primeira linha contém a dimensão n da matriz de entrada. Para as n linhas seguintes, espera-se os valores dos n elementos da linha separados por espaço.

O enunciado desta tarefa forneceu a seguinte matriz de entrada:

$$A = \begin{pmatrix} 2 & 4 & 1 & 1 \\ 4 & 2 & 1 & 1 \\ 1 & 1 & 1 & 2 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

Aplicando o algoritmo de Householder sobre esta matriz de entrada, foi obtida a seguinte matriz tridiagonal simétrica:

$$A = \begin{pmatrix} 2 & -4.24264069 & 0 & 0 \\ -4.24264069 & 3 & 1.41421356 & 0 \\ 0 & 1.41421356 & 2 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Esta matriz foi utilizada como entrada para o algoritmo QR com deslocamento espectral, e foram obtidos os autovalores 7, -2, 2 e -1, associados respectivamente aos autovetores da seguinte matriz:

$$V = \begin{bmatrix} 0.63245554 & 0.70710675 & 0.31622783 & 0 \\ 0.63245554 & -0.70710681 & 0.31622768 & 0 \\ 0.31622775 & 0.00000007 & -0.63245554 & -0.70710678 \\ 0.31622775 & 0.00000007 & -0.63245554 & 0.70710678 \end{bmatrix}$$

Os autovalores corretos foram fornecidos pelo enunciado e confirmam os autovalores obtidos. Para cada autovalor foram realizadas as multiplicações Av e λv, e a maior diferença entre elementos resultantes dessas multiplicações foi de 2.6226e-07, confirmando que os autovetores obtidos estão corretos.

Tarefa B

Esta tarefa tem os mesmos objetivos da tarefa anterior, porém a matriz de entrada segue o formato

$$A = \begin{pmatrix} n & n-1 & n-2 & \cdots & 2 & 1 \\ n-1 & n-1 & n-2 & \cdots & 2 & 1 \\ n-2 & n-2 & n-2 & \cdots & 2 & 1 \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ 2 & 2 & 2 & \cdots & 2 & 1 \\ 1 & 1 & 1 & \cdots & 1 & 1 \end{pmatrix}$$

A entrada desta tarefa segue o mesmo modelo da tarefa anterior, com a primeira linha contendo as dimensões da matriz e as linhas seguintes contendo seus elementos.

Uma propriedade deste formato de matriz é que seus autovalores podem ser calculados analiticamente através da fórmula

$$\lambda_i = \frac{1}{2}\left[1 - \cos\frac{(2i-1)\pi}{2n+1}\right]^{-1}, \quad i = 1, 2, \ldots, n.$$

A implementação desta tarefa calcula os autovalores utilizando esta fórmula para compará-los aos obtidos através do algoritmo de Householder e QR com deslocamento espectral.

A entrada fornecida no enunciado para esta tarefa é uma matriz deste formato com n = 20.

Os autovalores obtidos pelo algoritmo encontram-se na tabela abaixo, e equivalem aos valores calculados utilizando a fórmula acima.

140.40426751	19.00809949	6.89678489	3.56048281	2.1880802
1.49398983	1.09545235	0.84612196	0.68025499	0.56476973
0.48155512	0.42003002	0.37368736	0.33835914	0.31128881
0.29060955	0.27503819	0.26369005	0.25596443	0.25147358

Novamente, para cada autovalor foram realizadas as multiplicações Av e λv, e a maior diferença entre elementos resultantes dessas multiplicações foi de 6.66558e-08, confirmando que os autovetores obtidos estão corretos. A matriz de autovetores multiplicada por sua transposta resultou na matriz identidade, confirmando a ortogonalidade.

Tarefa C

A terceira tarefa é uma aplicação dos algoritmos implementados nas tarefas anteriores para calcular as frequências e os modos de vibração de uma treliça plana.

Na primeira linha da entrada desta tarefa, são esperados o número total de nós da treliça, o número de nós que não estão fixos e o número *n* de barras da treliça. Na linha seguinte, devem ser fornecidos a densidade, a área da seção transversal das barras e o módulo de elasticidade. As *n* linhas seguintes

correspondem às barras da treliça e contêm os nós dos extremos, o ângulo da barra com a horizontal e o comprimento da barra.

Para cada barra fornecida, é calculada sua matriz de rigidez Kij através da fórmula

$$K^{\{i,j\}} = \frac{AE}{L_{\{i,j\}}} \cdot \begin{pmatrix} C^2 & CS & -C^2 & -CS \\ CS & S^2 & -CS & -S^2 \\ -C^2 & -CS & C^2 & CS \\ -CS & -S^2 & CS & S^2 \end{pmatrix},$$

onde $L_{\{i,j\}}$ é o comprimento da barra, θ é o ângulo que a barra forma com o eixo horizontal, C = cos θ, S = sen θ, A é a seção transversal e E é o módulo de elasticidade da barra. Os elementos desta matriz são somados a elementos específicos da matriz de rigidez total.

Para cada barra, também é calculada sua contribuição à matriz de massa M. Essa matriz é uma matriz diagonal que tem como dimensão o dobro do número de nós não fixos. Por ter elementos apenas em sua diagonal principal, a implementação dessa matriz é feita como um vetor unidimensional.

A contribuição de cada barra para cada nó da treliça é dada por

$$m_i = \frac{\rho A L}{2}$$

e este valor é adicionado aos elementos

$$M_{2i-1,2i-1} = m_i,$$
$$M_{2i,2i} = m_i,$$

da matriz de massa M.

Uma vez calculada a matriz K, é possível calcular K̃ através da fórmula

$$\tilde{K} = M^{-\frac{1}{2}} K M^{-\frac{1}{2}} .$$

Aplicando o algoritmo de Householder e o algoritmo QR com deslocamento espectral usando K̃ como entrada, obtemos as frequências de vibração ao quadrado como autovalores. Multiplicando os autovetores obtidos por $M^{-\frac{1}{2}}$, é possível obter os modos de vibração associados aos respectivos autovalores.

Para a treliça fornecida no enunciado, as cinco menores frequências de vibração e seus respectivos módulos encontrados são os seguintes:

```
As 5 menores frequências são:

[1] Freq:  24.592547769722934
[1] Modo:  [-0.00289419  0.00234198  0.00419591  0.00248209  0.00028551 -0.0013057
  0.00098526  0.00151052  0.00022662  0.00038914 -0.00073802 -0.0001747
  0.00028799  0.00088098 -0.00132866  0.00059156 -0.00065368  0.00158733
 -0.00029456 -0.00139091  0.00222922 -0.00284524  0.00225084 -0.00236277]

[2] Freq:  92.01244464604143
[2] Modo:  [ 0.00068929  0.00005221 -0.0000603   0.00056394  0.0020657   0.00036113
  0.00127479  0.00083005  0.00229367  0.00102458 -0.0014041   0.0045805
  0.00227686 -0.00172978  0.00364141  0.00037394  0.00227883  0.00107473
 -0.00267236 -0.00040333  0.00006096  0.00092456 -0.00007675 -0.00149463]

[3] Freq:  94.70336537381655
[3] Modo:  [ 0.00046639 -0.00006748  0.0001149   0.00071595  0.00143544  0.00042405
  0.00083898 -0.00004741  0.00292066 -0.00003828  0.00484535  0.000807
  0.00122527 -0.00233281 -0.00117764 -0.00080496 -0.00513683  0.00088744
  0.00003202 -0.00005696  0.00031484 -0.00006811 -0.00157551 -0.00005737]

[4] Freq:  142.80969710649
[4] Modo:  [-0.00046639  0.00006748  0.0001149  -0.00071595 -0.00143544  0.00042405
 -0.00083898 -0.00004741 -0.00292066 -0.00003828 -0.00484535  0.000807
  0.00122527 -0.00233281  0.00117764  0.00080496 -0.00513683  0.00088744
 -0.00003202 -0.00005696  0.00031484  0.00006811 -0.00157551  0.00005737]

[5] Freq:  150.82212651081593
[5] Modo:  [ 0.00068929  0.00005221  0.0000603   0.00056394  0.0020657  -0.00036113
  0.00127479 -0.00083005  0.00229367 -0.00102458 -0.0014041  -0.0045805
 -0.00227686  0.00172978  0.00364141  0.00037394 -0.00227883 -0.00107473
 -0.00267236  0.00040333 -0.00006096  0.00092456  0.00007675 -0.00149463]
```

Tarefa D

A tarefa D, ou tarefa bônus, consiste em gerar uma imagem para visualizar as vibrações da treliça. Ao executar o programa no modo D, a saída do programa segue o formato

```
# i Xi(0) Yi(0) Xi(200) Yi(200)
1 5.01 7.19 4.97 6.92
2 8.15 3.07 7.93 2.90
3 3.04 8.29 3.18 8.35
4 6.07 2.25 5.90 2.17
```

onde a primeira linha é um comentário, e cada linha subsequente corresponde a um nó não fixo. Os valores das colunas alternam entre o deslocamento horizontal e vertical do nó.

Esta saída pode ser redirecionada como entrada para o programa plot.py, que gera uma sequência de imagens seguindo os parâmetros especificados no arquivo README.md.

Conclusão:

Concluímos por meio dos dois exercícios programa que podemos calcular os autovalores e autovetores de qualquer matriz simétrica, utilizando métodos computacionais. Os autovalores e autovetores se mostraram muito úteis na resolução de problemas concretos cotidianos e não apenas matemáticos abstratos.

Apêndice

Saída do programa para a Tarefa A

```
PS C:\Users\leona\Downloads\Householder-EP2-numerico> cat input-a | python .\main2.py -a
Matriz de entrada:
[[2. 4. 1. 1.]
 [4. 2. 1. 1.]
 [1. 1. 1. 2.]
 [1. 1. 2. 1.]]
Matriz tridiagonalizada:
[[ 2.         -4.24264069 -0.         -0.        ]
 [-4.24264069  3.          1.41421356  0.        ]
 [-0.          1.41421356  2.         -0.        ]
 [-0.          0.         -0.         -1.        ]]
Verificando A*v = w*v para 1° autovetor
V =  [0.63245554 0.63245554 0.31622775 0.31622775]
A*v =  [4.42718874 4.42718874 2.21359434 2.21359434]
w*v =  [4.42718876 4.42718876 2.21359428 2.21359428]
Verificando A*v = w*v para 2° autovetor
V =  [ 0.70710675 -0.70710681  0.00000007  0.00000007]
A*v =  [-1.41421363  1.4142135   0.00000013  0.00000013]
w*v =  [-1.4142135   1.41421363 -0.00000013 -0.00000013]
Verificando A*v = w*v para 3° autovetor
V =  [ 0.31622783  0.31622768 -0.63245554 -0.63245554]
A*v =  [ 0.63245531  0.6324556  -1.2649111  -1.2649111 ]
w*v =  [ 0.63245566  0.63245536 -1.26491108 -1.26491108]
Verificando A*v = w*v para 4° autovetor
V =  [ 0.          0.         -0.70710678  0.70710678]
A*v =  [ 0.          0.          0.70710678 -0.70710678]
w*v =  [-0.         -0.          0.70710678 -0.70710678]
O maior erro entre A*v e w*v foi 2.622610094065338e-07

Autovalores:
[6.999999999999998, -1.9999999999999578, 1.9999999999999578, -1.0]

Matriz de autovetores:
[[ 0.63245554  0.70710675  0.31622783  0.        ]
 [ 0.63245554 -0.70710681  0.31622768  0.        ]
 [ 0.31622775  0.00000007 -0.63245554 -0.70710678]
 [ 0.31622775  0.00000007 -0.63245554  0.70710678]]

Verificando a ortogonalidade:
[[ 1.  0. -0. -0.]
 [ 0.  1.  0.  0.]
 [-0.  0.  1.  0.]
 [-0.  0.  0.  1.]]
Como a multiplicação da matriz de autovetores pela sua transposta é a própria identidade a
matriz de autovetores é ortogonal
```