

# Autovalores e Autovetores de Matrizes Tridiagonais Simétricas - O Algoritmo QR

EP1 - MAP3121 - Data de entrega: 27/06/2021

May 31, 2021

## Regras do Jogo

- Você deve implementar o exercício programa em C/C++ ou Python3.7 (alunos da elétrica preferencialmente em C, os demais preferencialmente em Python)
- Python:
  - Pode usar: Matplotlib, NumPy (apenas para trabalhar com aritmética de vetores, matrizes, leitura/escrita de dados), bibliotecas básicas auxiliares: sys, time, datetime, os, math.
  - **Não** pode usar: SciPy ou outras bibliotecas de algebra linear computacional
- C, C++:
  - **Não** pode usar recursos de versões além de C/C++14.
  - Pode usar qualquer biblioteca nativa do gcc/g++ (que não exija instalação adicional).
- Incluir, obrigatoriamente, um arquivo LEIAME.txt com instruções de compilação e execução, indicando versão de interpretador/compilador necessário.
- O exercício pode ser feito em duplas. As duplas podem ser formadas livremente, com alunos de turmas e / ou engenharias distintas.
- Apenas um aluno deve entregar o exercício, destacando no relatório e no código o nome de ambos os alunos.
- A entrega deve conter o relatório (em .pdf), contendo a análise do problema estudado, e o código usado para as simulações computacionais (arquivos fonte). A entrega deve ser feita em um arquivo compactado único.
- O relatório deve apresentar resultados e análises de todas as tarefas descritas neste enunciado.
- O seu código deve estar bem documentado, de forma a facilitar a correção. Rodar os testes também deve ser fácil para o usuário do seu programa, sem que este tenha que editar seu código. Ou seja, você deve pedir como entrada qual teste o usuário quer rodar, qual método e os parâmetros para o teste.

## 1 Introdução

Matrizes reais simétricas, e em particular as tridiagonais, aparecem comumente em aplicações e seus auto-valores e auto-vetores carregam informações relevantes. Neste exercício-programa estudaremos como calcular autovalores e autovetores de matrizes tridiagonais simétricas e seu uso em uma aplicação prática. Este método é também relevante para matrizes simétricas em geral e será utilizado também no segundo exercício programa.

Recordemos que matrizes reais simétricas  $n \times n$  são tais que todos os seus autovalores são reais e os respectivos autovetores podem ser escolhidos de maneira a formar uma base *ortonormal* do  $\mathbb{R}^n$ .

Sejam então  $\{\lambda_1, \lambda_2, \dots, \lambda_n\}$  os autovalores da matriz simétrica  $A \in \mathbb{R}^{n \times n}$ , contando multiplicidade, e  $\{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_n\}$  respectivos autovetores formando uma base ortonormal do  $\mathbb{R}^n$ . Então temos

$$A\mathbf{q}_i = \lambda_i \mathbf{q}_i, \quad 1 \leq i \leq n, \quad \text{e} \quad \mathbf{q}_i^T \mathbf{q}_j = \begin{cases} 0 & \text{se } i \neq j \\ 1 & \text{se } i = j \end{cases},$$

e todo  $\mathbf{x} \in \mathbb{R}^n$  pode ser representado como

$$\mathbf{x} = \sum_{j=1}^n (\mathbf{q}_j^T \mathbf{x}) \mathbf{q}_j.$$

A matriz  $A$  pode ser escrita na sua decomposição espectral

$$A = \sum_{j=1}^n \lambda_j \mathbf{q}_j \mathbf{q}_j^T \quad \text{ou} \quad A = Q \Lambda Q^T,$$

onde  $\Lambda$  é a matriz *diagonal* cujos elementos na sua diagonal são os autovalores  $\{\lambda_i\}_{i=1}^n$  e  $Q$  é a matriz de  $\mathbb{R}^{n \times n}$  cuja  $j$ -ésima coluna é o autovetor  $\mathbf{q}_j$ ,  $1 \leq j \leq n$ . Note que  $Q$  é uma matriz *ortogonal*:  $Q^T = Q^{-1}$  (ou  $Q^T Q = Q Q^T = I$ , onde  $I$  é a matriz identidade).

Nosso objetivo é calcular os autovalores e autovetores de  $A$ . Apresentaremos um dos métodos mais eficientes, que pode ser usado em situações mais gerais do que aqui, em que usaremos algumas particularidades de matrizes tridiagonais simétricas. Trata-se de um método iterativo, como em princípio são todos os esquemas utilizados para este fim.

## 2 O Método da Potência

Antes de apresentarmos o algoritmo QR, veremos primeiro o Método da Potência. Este é um método simples para a aproximação do auto-valor dominante de uma matriz e um respectivo auto-vetor. O algoritmo consiste em aplicar repetidas vezes a matriz  $A$  em um vetor, com alguma normalização para se evitar o possível crescimento indeterminado de vetores. Suponha que os autovalores da matriz simétrica  $A$  estejam ordenados decrescentemente em módulo e que o autor-valor de maior módulo seja simples,

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n|,$$

e que  $\mathbf{v}$  seja um vetor do  $\mathbb{R}^n$  cuja projeção na direção do autovetor  $\mathbf{q}_1$  é diferente de zero.

Então  $\mathbf{v}$  pode ser representado como

$$\mathbf{v} = c_1 \mathbf{q}_1 + c_2 \mathbf{q}_2 + \dots + c_n \mathbf{q}_n, \quad \text{onde } c_1 \neq 0.$$

Da representação espectral de  $A$  segue então que, para todo inteiro  $k \geq 0$ , temos

$$A^k \mathbf{v} = c_1 \lambda_1^k \mathbf{q}_1 + c_2 \lambda_2^k \mathbf{q}_2 + \dots + c_n \lambda_n^k \mathbf{q}_n = \lambda_1^k \left[ c_1 \mathbf{q}_1 + c_2 \left( \frac{\lambda_2}{\lambda_1} \right)^k \mathbf{q}_2 + \dots + c_n \left( \frac{\lambda_n}{\lambda_1} \right)^k \mathbf{q}_n \right].$$

Como  $|\lambda_j/\lambda_1| < 1$ ,  $2 \leq j \leq n$ , vemos que à medida que  $k$  cresce, as componentes do vetor  $A^k \mathbf{v}$  nas direções de  $\{\mathbf{q}_j\}_{j=2}^n$  tendem a zero e ele tende a ficar paralelo a  $\mathbf{q}_1$ . Logo, se normalizarmos os vetores após cada iteração, teremos um algoritmo para aproximar  $\lambda_1$  e  $\mathbf{q}_1$  (ou  $-\mathbf{q}_1$ ):

Dado  $\widehat{\mathbf{v}}^0$  itere:

$$k = 0$$

repita

$$\begin{aligned} \mathbf{v}^{k+1} &= A \widehat{\mathbf{v}}^k \\ \widehat{\mathbf{v}}^{k+1} &= \frac{\mathbf{v}^{k+1}}{\|\mathbf{v}^{k+1}\|} && (\text{aproximação do autovetor}) \\ \widehat{\lambda}_{k+1} &= (\widehat{\mathbf{v}}^{k+1})^T A \widehat{\mathbf{v}}^{k+1} && (\text{aproximação do autovalor}) \end{aligned}$$

até a convergência

No pseudocódigo acima  $\|\mathbf{x}\| = (\sum_{i=1}^n x_i^2)^{1/2}$  é a norma Euclidiana de um vetor do  $\mathbb{R}^n$ . Note que  $\lambda_1 = \mathbf{q}_1^T A \mathbf{q}_1$ , pois  $\mathbf{q}_1$  é unitário, e como  $\widehat{\mathbf{v}}_{k+1}$  é um vetor unitário cuja direção se aproxima da direção de  $\mathbf{q}_1$ , a última igualdade nos dá uma aproximação para  $\lambda_1$ . A taxa de convergência do algoritmo é proporcional a  $|\lambda_2/\lambda_1|$ .

### 3 Rotações de Givens e a fatoração QR de uma matriz tridiagonal

Apresentamos a fatoração QR de matrizes  $n \times n$  tridiagonais simétricas por rotações de Givens, parte importante do método QR para a determinação dos seus auto-valores. Rotações de Givens são transformações lineares ortogonais de  $\mathbb{R}^n$  em  $\mathbb{R}^n$ . Recordemos inicialmente que uma transformação linear  $Q$  dada por uma matriz  $n \times n$  é ortogonal se  $Q^T Q = Q Q^T = I$ . Note que o produto de duas matrizes ortogonais também é ortogonal. Lembremos que uma transformação de  $\mathbb{R}^2$  em  $\mathbb{R}^2$  dada por

$$Q = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

representa uma rotação em torno da origem por um ângulo  $\theta$  em sentido anti-horário. Uma tal transformação é claramente ortogonal. Rotações de Givens são transformações ortogonais  $Q(i, j, \theta)$  de  $\mathbb{R}^n$  em  $\mathbb{R}^n$  tais que para  $y = Q(i, j, \theta)x$  obtemos  $y_k = x_k$ , para  $k \neq i$  e  $k \neq j$ ,  $y_i = \cos \theta x_i - \sin \theta x_j$  e  $y_j = \sin \theta x_i + \cos \theta x_j$ . Ou seja, tal transformação corresponde a uma rotação no plano das coordenadas  $i$  e  $j$ , deixando as outras invariantes. Ao aplicarmos uma rotação de Givens  $Q(i, j, \theta)$  a uma matriz  $A_{n \times n}$  apenas as linhas  $i$  e  $j$  de  $A$  são modificadas. Denotando  $c = \cos \theta$  e  $s = \sin \theta$  teremos após a aplicação da transformação que as linhas  $i$  e  $j$  da matriz resultante  $B = Q(i, j, \theta)A$  serão dadas por

$$b_{i,k} = ca_{i,k} - sa_{j,k} \quad \text{e} \quad b_{j,k} = sa_{i,k} + ca_{j,k} \quad , \quad k = 1, \dots, n \quad (1)$$

enquanto que as linhas restantes não se alteram.

Vamos detalhar como transformar uma matriz  $A$  tridiagonal simétrica em uma matriz  $R_{n \times n}$  triangular superior através de rotações de Givens. Inicialmente notemos que podemos armazenar apenas os elementos não nulos de  $A$  em vetores  $\alpha$ ,  $\beta$  e  $\gamma$  tal que

$$A = \begin{bmatrix} \alpha_1 & \gamma_1 & 0 & \dots & 0 \\ \beta_1 & \alpha_2 & \gamma_2 & 0 & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & 0 & \beta_{n-2} & \alpha_{n-1} & \gamma_{n-1} \\ 0 & \dots & 0 & \beta_{n-1} & \alpha_n \end{bmatrix} .$$

Note que quando  $A$  é simétrica, temos  $\beta = \gamma$ .

Começamos aplicando transformação de Givens  $Q_1 = Q(1, 2, \theta_1)$  a  $A$  de forma a zerar a posição  $\beta_1$  da matriz. Para tanto, basta definir adequadamente o valor de  $\theta_1$ , ou equivalentemente, de  $c_1 = \cos \theta_1$  e de  $s_1 = \sin \theta_1$ . Escolhendo (para  $k = 1$ )

$$c_k = \frac{a_k}{\sqrt{a_k^2 + b_k^2}} \quad \text{e} \quad s_k = -\frac{b_k}{\sqrt{a_k^2 + b_k^2}}$$

teremos o efeito desejado.

Por exemplo, considere a matriz:

$$A = \begin{bmatrix} 2 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix} .$$

Temos que  $c_1 = 2/\sqrt{5}$  e  $s_1 = -1/\sqrt{5}$  e após a aplicação da transformação obtemos (verifique!) :

$$Q_1 A = \begin{bmatrix} 5/\sqrt{5} & 4/\sqrt{5} & 1/\sqrt{5} & 0 \\ 0 & 3/\sqrt{5} & 2/\sqrt{5} & 0 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix} . \quad (2)$$

Note que desta forma zera-se o elemento abaixo da diagonal na segunda linha e surge um elemento não nulo na posição (1,3) da matriz. Com transformações do tipo  $Q_k = Q(k, k+1, \theta_k)$ ,  $k = 2, \dots, n-1$  zera-se sucessivamente os elementos abaixo da diagonal, ao mesmo tempo em que a matriz ganha uma nova diagonal não nula, nas posições  $(k, k+2)$ ,  $k = 1, \dots, n-2$ . A matriz resultante é triangular superior.

Observe que só precisamos de mais um vetor para armazenar esta nova diagonal, sem necessidade de armazenar toda a matriz.

**Observação:** Uma forma essencialmente equivalente (os sinais de  $c_k$  e  $s_k$  podem ser trocados), mas numericamente mais estável, de determinar valores de  $c_k$  e  $s_k$  (no passo  $k$ ) é a seguinte: Se  $|a_k| > |b_k|$  defina

$$\tau = -b_k/a_k, c_k = 1/\sqrt{1+\tau^2} \text{ e } s_k = c_k\tau. \quad (3)$$

Caso contrário, defina

$$\tau = -a_k/b_k, s_k = 1/\sqrt{1+\tau^2} \text{ e } c_k = s_k\tau. \quad (4)$$

As sucessivas transformações de Givens  $Q_1, Q_2, \dots, Q_{n-1}$  são tais que  $Q_{n-1} \dots Q_2 Q_1 A = R$ , onde  $R$  é triangular superior. Como as transformações de Givens são ortogonais, teremos que  $A = QR$ , com  $Q$  ortogonal dada por  $Q = Q_1^T Q_2^T \dots Q_{n-1}^T$ .

## 4 O algoritmo QR

O algoritmo QR para a determinação dos auto-valores de uma matriz simétrica  $A \in \mathbb{R}^{n \times n}$  é dado por

$$A^{(0)} = A; V^{(0)} = I_{n \times n}$$

$$k = 0$$

*repita*

$$\begin{aligned} A^{(k)} &\rightarrow Q^{(k)} R^{(k)} && (\text{fatoração QR de } A^{(k)}) \\ A^{(k+1)} &= R^{(k)} Q^{(k)} && (\text{atualização da matriz}) \\ V^{(k+1)} &= V^{(k)} Q^{(k)} && (\text{atualização dos autovetores}) \end{aligned}$$

*até a convergência*

Como  $A^{(k+1)} = R^{(k)} Q^{(k)} = (Q^{(k)})^T Q^{(k)} R^{(k)} Q^{(k)} = (Q^{(k)})^T A^{(k)} Q^{(k)}$ , segue que  $A^{(k+1)}$  e  $A^{(k)}$  são (ortogonalmente) semelhantes.

Vamos agora considerar as particularidades do algoritmo quando aplicado a uma matriz tridiagonal simétrica. Vimos na seção anterior que:

$$Q_{n-1} \dots Q_2 Q_1 A = R,$$

quando  $A$  é tridiagonal, as matrizes  $Q_j$  representam rotações de Givens e a matriz resultante  $R$  é triangular superior, com elementos não nulos apenas na diagonal principal e nas duas diagonais imediatamente acima desta. No algoritmo QR devemos notar que  $A^{(k)} = Q^{(k)} R^{(k)}$ , com  $Q^{(k)} = (Q_{n-1}^{(k)})^T \dots (Q_1^{(k)})^T$ . Devemos formar a nova matriz  $A^{(k+1)} = R^{(k)} Q^{(k)}$ , ou seja,

$$A^{(k+1)} = R^{(k)} (Q_1^{(k)})^T \dots (Q_{n-1}^{(k)})^T.$$

Teremos

$$A^{(k+1)} = Q_{n-1}^{(k)} (\dots (Q_1^{(k)} A^{(k)} (Q_1^{(k)})^T) \dots) (Q_{n-1}^{(k)})^T,$$

e portanto  $A^{(k+1)}$  também é simétrica. Este fato pode, e deve, ser usado na implementação do algoritmo. Para facilitar a compreensão, vejamos um exemplo, com a seguinte matriz:

$$A^{(0)} = \begin{bmatrix} 4 & 3 & 0 \\ 3 & 4 & 3 \\ 0 & 3 & 4 \end{bmatrix}.$$

Temos então que  $c_1 = 4/5$  e  $s_1 = -3/5$ . Após aplicar a primeira rotação de Givens à matriz obtemos:

$$Q_1^{(0)} A = \begin{bmatrix} 5 & 24/5 & 9/5 \\ 0 & 7/5 & 12/5 \\ 0 & 3 & 4 \end{bmatrix}.$$

Para a segunda rotação de Givens temos  $c_2 = 7/\sqrt{274}$  e  $s_2 = -15/\sqrt{274}$  (com a forma mais estável teríamos os sinais de  $c_2$  e  $s_2$  trocados) e obtemos

$$Q_2^{(0)} Q_1^{(0)} A = R = \begin{bmatrix} 5 & 24/5 & 9/5 \\ 0 & 54.8/\sqrt{274} & 76.8/\sqrt{274} \\ 0 & 0 & -8/\sqrt{274} \end{bmatrix}.$$

No proximo passo avaliamos

$$R^{(0)}(Q_1^{(0)})^T = \begin{bmatrix} 172/25 & 21/25 & 9/5 \\ 32.88/\sqrt{274} & 43.84/\sqrt{274} & 76.8/\sqrt{274} \\ 0 & 0 & -8/\sqrt{274} \end{bmatrix}.$$

Por fim, completamos o cálculo de  $A^{(1)} = R^{(0)}Q^{(0)}$ :

$$A^{(1)} = R^{(0)}(Q_1^{(0)})^T(Q_2^{(0)})^T = \begin{bmatrix} 172/25 & 32.88/\sqrt{274} & 0 \\ 32.88/\sqrt{274} & 1458.88/274 & -120/274 \\ 0 & -120/274 & -56/274 \end{bmatrix}.$$

Todos os resultados apresentados neste exemplo são exatos, procure verificar como foram obtidos. As contas devem ajudá-lo a entender os comentários que seguem.

Agora vejamos alguns fatos que serão úteis para a implementação. O primeiro deles é que tudo que precisamos armazenar (durante cada iteração) de informação sobre cada rotação de Givens são os respectivos valores de  $c_k$  e  $s_k$ , para o que basta o uso de dois vetores. Numa nova iteração pode-se utilizar os mesmos vetores, as rotações de Givens da iteração anterior não mais serão utilizadas. As sucessivas aplicações das transpostas das rotações de Givens  $Q_i^{(k)}$  à direita alteram apenas as colunas  $i$  e  $i+1$  da matriz  $R^{(k)}$ . Como a matriz  $R^{(k)}$  é triangular superior, a cada multiplicação à direita por  $(Q_i^{(k)})^T$  surge um elemento não nulo na primeira diagonal abaixo da diagonal principal (na posição  $(i+1, i)$  da matriz), com o restante da matriz abaixo dessa subdiagonal permanecendo nulo. Como a matriz resultante  $R^{(k)}Q^{(k)}$  será simétrica e o elemento atualizado não mais será modificado nos passos seguintes, já sabemos que o elemento da posição  $(i, i+1)$  será igual a ele, sem necessidade de recalculá-lo. Também da simetria, deduzimos que os elementos não nulos que haviam surgido na segunda diagonal acima da principal (no exemplo, o valor  $9/5$  na posição  $(1, 3)$ ), voltará a ser zero. Segue portanto que a nova matriz  $A^{(k+1)}$  também é tridiagonal simétrica e que no seu armazenamento em vetores (como descrito anteriormente) teremos novamente  $\gamma = \beta$ . Procure perceber que se usarmos o valor final da subdiagonal  $\beta$  para atualizar o valor da sobrediagonal  $\gamma$ , o elemento  $9/5$  da matriz do exemplo não precisaria sequer ter sido calculado. Isso pode ser usado na implementação do algoritmo.

Para o cálculo da matriz  $V^{(k)}$  contendo os auto-vetores em suas colunas, inicia-se com a matriz  $V^{(0)} = I$  e a cada iteração se multiplica sucessivamente por  $(Q_1^{(k)})^T, (Q_2^{(k)})^T, \dots, (Q_{n-1}^{(k)})^T$ . Após este cálculo, na nova iteração pode-se utilizar os mesmos vetores usados para armazenar  $c_k$  e  $s_k$  sobrescrevendo-os.

No exemplo acima teremos:

$$V^{(0)}(Q_1^{(0)})^T = \begin{bmatrix} 4/5 & -3/5 & 0 \\ 3/5 & 4/5 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

e

$$V^{(1)} = V^{(0)}(Q_1^{(0)})^T(Q_2^{(0)})^T = \begin{bmatrix} 4/5 & -4.2/\sqrt{274} & 9/\sqrt{274} \\ 3/5 & 5.6/\sqrt{274} & -12/\sqrt{274} \\ 0 & 15/\sqrt{274} & 7/\sqrt{274} \end{bmatrix}.$$

Observamos novamente que a multiplicação à esquerda por  $(Q_i^{(k)})^T$  altera apenas as colunas  $i$  e  $i+1$  da matriz.

## 4.1 Convergência do Algoritmo QR

Apresentamos aqui uma demonstração simples da convergência do algoritmo, sob certas hipóteses. O método é convergente mesmo removendo algumas destas hipóteses, porém a demonstração é mais envolvida. Vocês podem ver por exemplo o livro clássico *Matrix Computations*, de Gene Golub e Charles Van Loan.

A demonstração seguinte é para o caso em que  $A$  é simétrica positiva definida com auto-valores distintos  $\lambda_1 > \lambda_2 > \dots > \lambda_n > 0$ . Vamos ainda supor que a transposta da matriz ortogonal  $V$ , cujas colunas são os auto-vetores correspondentes (ou seja  $AV = V\Lambda$ , sendo  $\Lambda$  a matriz diagonal com os auto-valores de  $A$ ), possua uma decomposição  $V^T = LU$ , com  $L$  triangular unitária inferior e  $U$  triangular superior.

Já observamos que  $A^{(k+1)} = (Q^{(k)})^T A^{(k)} Q^{(k)}$ . Usando esta igualdade recursivamente temos que:

$$A^{(k+1)} = (Q^{(k)})^T (Q^{(k-1)})^T \dots (Q^{(0)})^T A Q^{(0)} \dots Q^{(k-1)} Q^{(k)} . \quad (5)$$

Vamos agora mostrar por indução que potências da matriz  $A$  satisfazem a relação:

$$A^{k+1} = Q^{(0)} \dots Q^{(k-1)} Q^{(k)} R^{(k)} R^{(k-1)} \dots R^{(0)} .$$

A base da indução, para  $k = 0$  decorre do fato de que  $A = A^{(0)} = Q^{(0)} R^{(0)}$ . Temos então que

$$A^{k+1} = AA^k = A(Q^{(0)} \dots Q^{(k-1)} R^{(k-1)} \dots R^{(0)}) ,$$

da hipótese de indução e de (5) que

$$Q^{(0)} \dots Q^{(k-1)} A^{(k)} = A Q^{(0)} \dots Q^{(k-1)} .$$

Substituindo esta igualdade na equação anterior obtemos

$$A^{k+1} = Q^{(0)} \dots Q^{(k-1)} A^{(k)} R^{(k-1)} \dots R^{(0)} = Q^{(0)} \dots Q^{(k-1)} Q^{(k)} R^{(k)} R^{(k-1)} \dots R^{(0)} .$$

Com  $V$  a matriz ortogonal dos auto-vetores de  $A$ , temos então que

$$A^k = V \Lambda^k V^T = V \Lambda^k LU = Q^{(0)} \dots Q^{(k-1)} R^{(k-1)} \dots R^{(0)} ,$$

de onde obtemos que (multiplicando à direita por  $U^{-1} \Lambda^{-k}$ )

$$V \Lambda^k L \Lambda^{-k} = Q^{(0)} \dots Q^{(k-1)} R^{(k-1)} \dots R^{(0)} U^{-1} \Lambda^{-k} .$$

Observemos agora que  $\Lambda^k L \Lambda^{-k}$  é triangular unitária inferior (com 1's na diagonal) e seus elementos abaixo da diagonal são  $l_{i,j} (\frac{\lambda_i}{\lambda_j})^k$ , com  $i > j$ . Segue que para  $k$  tendendo a infinito, esta matriz converge à identidade. Denotemos por  $\tilde{Q}^{(k)} = Q^{(0)} \dots Q^{(k-1)}$  e por  $\tilde{R}^{(k)} = R^{(k-1)} \dots R^{(0)} U^{-1} \Lambda^{-k}$ , notando que  $\tilde{Q}^{(k)}$  é ortogonal e  $\tilde{R}^{(k)}$  é triangular superior, ou seja  $\tilde{Q}^{(k)} \tilde{R}^{(k)}$  é a decomposição  $QR$  de  $V \Lambda^k L \Lambda^{-k}$ . Logo obtemos que:

$$V \Lambda^k L \Lambda^{-k} = \tilde{Q}^{(k)} \tilde{R}^{(k)} \rightarrow V .$$

No limite, temos que  $\tilde{Q}^{(k)} \tilde{R}^{(k)}$  tende à decomposição  $QR$  de  $V$  e portanto  $\tilde{Q}^{(k)}$  tende a  $V$  e  $\tilde{R}^{(k)}$  à identidade. Portanto  $Q^{(k)}$  tende à identidade e de (5) obtemos que  $A^{(k+1)} = (\tilde{Q}^{(k)})^T A \tilde{Q}^{(k)}$  tende a  $V^T A V = \Lambda$ , mostrando o que queríamos.

## 4.2 O algoritmo QR com deslocamentos espectrais

Nesta seção apresentaremos o algoritmo QR a ser implementado. Uma última melhoria será proposta. Como a taxa de convergência depende da razão  $|\lambda_{j+1}/\lambda_j|$  entre os módulos dos autovalores, ela pode ser lenta se estiver próxima de 1. Podemos acelerar a convergência se subtrairmos da matriz um múltiplo da identidade,  $\mu_k$  (variando a cada iteração) próximo a um autovalor.

Vamos argumentar heurísticamente. Sendo  $\alpha_i^{(k)}$ ,  $1 \leq i \leq n$ , e  $\beta_i^{(k)}$ ,  $1 \leq i \leq n-1$ , os coeficientes da matriz tridiagonal simétrica  $A^{(k)}$  da iteração  $k$  do algoritmo QR, se fizermos as iterações na forma

$$\begin{aligned} A^{(k)} - \mu_k I &\rightarrow Q^k R^{(k)} \\ A^{(k+1)} &= R^{(k)} Q^{(k)} + \mu_k I \end{aligned}$$

a taxa de convergência de  $\alpha_j^{(k)}$  para  $\lambda_j$  (ou de  $\beta_{j-1}^{(k)}$  para 0) será proporcional a  $|(\lambda_j - \mu_k)/(\lambda_{j-1} - \mu_k)|$ . Portanto, se  $\mu_k$  estiver perto de  $\lambda_n$ , esperamos que  $\beta_{n-1}^{(k)}$  tenda a zero mais rápido do que  $\beta_j^{(k)}$  para  $j < n-1$ , e que  $\alpha_n^{(k)}$  convirja rapidamente para  $\lambda_n$ . Mudamos  $\mu_k$  a cada passo, de acordo com a **heurística de Wilkinson**:

$$\text{com } d_k = (\alpha_{n-1}^{(k)} - \alpha_n^{(k)})/2 \text{ defina } \mu_k = \alpha_n^{(k)} + d_k - \text{sgn}(d_k) \sqrt{d_k^2 - (b_{n-1}^{(k)})^2} ,$$

onde  $\text{sgn}(d) = 1$  se  $d \geq 0$  e  $\text{sgn}(d) = -1$  caso contrário.

Observemos que se  $A^{(k)} - \mu_k I = Q^k R^{(k)}$  então

$$A^{(k+1)} = R^{(k)} Q^{(k)} + \mu_k I = (Q^{(k)})^T (Q^{(k)} R^{(k)} + \mu_k I) Q^{(k)} = (Q^{(k)})^T A^{(k)} Q^{(k)},$$

e portanto as matrizes  $A^{(k)}$  são todas semelhantes a  $A$ .

Se chegarmos a  $\beta_{n-1}^{(k)} = 0$  então  $\alpha_n^{(k)}$  é auto-valor de  $A$ . Assim, na prática, quando  $\beta_{n-1}^{(k)} < \epsilon$  consideramos  $\beta_{n-1} = 0$  (convergência), já determinamos um dos auto-valores de  $A$  e podemos proceder da mesma forma com a submatriz tridiagonal  $n-1 \times n-1$ , com os correntes valores de  $\beta_j^{(k)}, j = 1, \dots, n-2$  e  $\alpha_j^{(k)}, j = 1, \dots, n-1$ . O segundo auto-valor estará determinado quando  $\beta_{n-2}^{(k)} < \epsilon$ . Considerando este valor como nulo, reduzimos a dimensão para  $n-2$  e assim por diante, até obtermos todos os auto-valores. O algoritmo então é o seguinte:

---

#### Algoritmo QR tridiagonal com deslocamento

---

- 1: Sejam  $A^{(0)} = A \in \mathbb{R}^{n \times n}$  uma matriz tridiagonal simétrica,  $V^{(0)} = I$  e  $\mu_0 = 0$ . O algoritmo calcula a sua forma diagonal semelhante  $A = V \Lambda V^T$ .
  - 2:  $k = 0$
  - 3: **para**  $m = n, n-1, \dots, 2$  **faça**
  - 4:   **repita**
  - 5:     se  $k > 0$  calcule  $\mu_k$  pela heurística de Wilkinson
  - 6:      $A^{(k)} - \mu_k I \rightarrow Q^{(k)} R^{(k)}$
  - 7:      $A^{(k+1)} = R^{(k)} Q^{(k)} + \mu_k I$
  - 8:      $V^{(k+1)} = V^{(k)} Q^{(k)}$
  - 9:      $k = k + 1$
  - 10:   **até que**  $\beta_{m-1}^{(k)} < \epsilon$
  - 11: **fim do para**
  - 12:  $\Lambda = A^{(k)}$
  - 13:  $V = V^{(k)}$
- 

## 5 Aplicação - Modos de vibração de sistemas massa-mola

Como aplicação do cálculo de auto-vetores e auto-valores de matrizes tridiagonais simétricas iremos considerar um sistema massa-mola. Sistemas massa-mola são utilizados na modelagem de vigas, alguns sistemas elétricos e outras aplicações. Os auto-vetores representam os modos naturais de vibração do sistema e os auto-valores determinam as frequências de vibração.

Inicialmente recordemos que uma equação diferencial do tipo

$$x''(t) + \lambda x(t) = 0$$

com  $\lambda$  positivo tem como solução geral

$$x(t) = a \cos(\omega t) + b \sin(\omega t)$$

com a frequência  $\omega = \sqrt{\lambda}$ . Dadas condições iniciais  $x(0) = x_0$  e  $x'(0) = d_0$  determina-se  $a$  e  $b$  de maneira única. Para um sistema de equações

$$X''(t) + AX(t) = 0$$

com  $X(t) = (x_1(t), x_2(t), \dots, x_n(t))^T$  e  $A$  uma matriz simétrica positiva definida (portanto real diagonalizável) consideremos inicialmente  $AQ = Q\Lambda$ , onde  $Q$  é a matriz ortogonal cujas colunas são os auto-vetores de  $A$  e  $\Lambda$  matriz diagonal composta pelos auto-valores de  $A$ . Temos portanto que

$$X''(t) + Q\Lambda Q^T X(t) = 0.$$

Definindo  $Y(t) = Q^T X(t)$  (ou seja,  $X(t) = QY(t)$ ), o sistema equivale a

$$Y''(t) + \Lambda Y(t) = 0,$$

que é composto por  $n$  equações escalares desacopladas como a descrita anteriormente. Cada uma destas componentes de  $Y(t)$  está associada a uma frequência determinada por um auto-valor de  $A$ . Assim, se  $X(0)$  é escolhido como um múltiplo de um dos auto-vetores de  $A$ , todas as massas se movem na mesma frequência, correspondente ao auto-valor respectivo. Temos então o modo de vibração associado a esta frequência.

## 5.1 Descrição do sistema com $n$ massas e $n+1$ molas

Consideremos um sistema com  $n$  massas pontuais iguais (com  $m$  kg), localizadas numa linha reta entre dois anteparos fixos. Cada duas massas consecutivas são conectadas por uma mola, e há também uma mola em cada extremo conectando a primeira e última mola aos anteparos. Temos assim  $n+1$  molas no sistema, cada uma com sua respectiva constante elástica  $k_i$ . O deslocamento de cada massa em relação à posição de equilíbrio no instante  $t$  é dado pela função  $x_i(t)$ , para  $i = 1, \dots, n$ . As massas se movimentam sobre uma superfície plana sem qualquer atrito ou outra força de amortecimento. Assim, as massas se movem em linha reta, entre os dois anteparos.

O sistema de equações diferenciais que descreve o movimento das  $n$  massas é dado por

$$\begin{aligned} x_1''(t) &= \frac{1}{m}(-k_1 x_1(t) + k_2(x_2(t) - x_1(t))), \\ x_i''(t) &= \frac{1}{m}(k_i(x_{i-1}(t) - x_i(t)) + k_{i+1}(x_{i+1}(t) - x_i(t))), \quad (\text{para } 2 \leq i \leq n-1) \\ x_n''(t) &= \frac{1}{m}(k_n(x_{n-1}(t) - x_n(t)) + k_{n+1}x_n(t)), \end{aligned} \quad (6)$$

que em forma matricial se escreve como

$$X''(t) + AX(t) = 0 \quad (7)$$

onde  $X(t) = (x_1(t), x_2(t), \dots, x_n(t))^T$ . A matriz  $A$  é dada por

$$A = \frac{1}{m} \begin{bmatrix} k_1 + k_2 & -k_2 & 0 & 0 & \dots & 0 & 0 \\ -k_2 & k_2 + k_3 & -k_3 & 0 & \dots & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 & \\ \vdots & \vdots & \vdots & \vdots & \dots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & -k_{n-1} & k_{n-1} + k_n & -k_n \\ 0 & 0 & \vdots & \vdots & 0 & -k_n & k_n + k_{n+1} \end{bmatrix},$$

e é tridiagonal simétrica positiva definida.

## 5.2 Tarefa

Vocês devem implementar o algoritmo QR com deslocamento espectral para o cálculo dos auto-valores e auto-vetores de uma matriz tridiagonal simétrica  $n \times n$ .

**a)** Como um primeiro teste, considere as matrizes com diagonal principal constante igual a 2 ( $\alpha_k = 2$ ) e subdiagonal igual a -1 ( $\beta_k = -1$ ). Os auto-valores e auto-vetores desta matriz são conhecidos analiticamente. Temos que

$$\lambda_j = 2(1 - \cos(\frac{j\pi}{n+1})), j = 1, \dots, n$$

com os auto-vetores correspondentes

$$v_j = (\sin(\frac{j\pi}{n+1}), \sin(\frac{2j\pi}{n+1}), \dots, \sin(\frac{nj\pi}{n+1})) .$$



Calcule os auto-valores e auto-vetores neste caso para  $n=4,8,16$  e  $32$ , usando o valor de  $\epsilon = 10^{-6}$  no critério de parada do esquema. Compare o número de iterações até a convergência nestes casos com o obtido pelo método QR sem deslocamentos (a única diferença é que neste caso o valor de  $\mu_k$  é definido como zero em todas as etapas).

**b)** Considere o sistema massa-mola com 5 massas de 2kg e as molas com  $k_i = (40 + 2i)$  N/m,  $i = 1, \dots, 6$ . Faça gráficos da evolução da solução, mostrando os deslocamentos de cada mola em relação a sua posição de equilíbrio, por 1 minuto iniciando com os seguintes deslocamentos, com velocidade inicial nula:

- $X(0) = -2, -3, -1, -3, -1$
- $X(0) = 1, 10, -4, 3, -2$
- $X(0)$  correspondente ao modo de maior frequência

Determine as frequências e seus respectivos modos de vibração.

**c)** Considere o sistema massa-mola com 10 massas de 2kg e as molas com  $k_i = (40 + 2(-1)^i)$  N/m,  $i = 1, \dots, 11$ . Realize as mesmas tarefas do item anterior (com velocidade inicial nula), sendo que nas definições de  $X(0)$  use  $x_{i+5} = x_i$ , com os valores  $x_i, i = 1, \dots, 5$  dados no item anterior.