

CURS 4

METODE NUMERICE PENTRU PROBLEMA DE VALORI PROPRII

Partea II

1. Algoritmul QR
2. Problema generalizată de valori proprii.

1 Algoritmul QR**1.1 Metoda**

QR este unul din cei mai utilizați algoritmi pentru a determina toate valorile proprii ale unei matrici. În numele metodei, Q desemnează o matrice ortogonală, iar R o matrice superior-triunghiulară (R vine de la *right*).

Considerăm o *matrice reală* nesingulară \mathbf{A} , simetrică sau nu.

Esența algoritmului *QR* constă în următoarea proprietate:

Dacă \mathbf{A} este factorizată în produsul $\mathbf{A} = \mathbf{QR}$, unde \mathbf{Q} este nesingulară, atunci matricea produsului în ordine inversă, $\mathbf{A}' = \mathbf{RQ}$, are aceleași valori proprii ca și \mathbf{A} (fiind similară cu \mathbf{A}) ■

Într-adevăr, avem: $\mathbf{R} = \mathbf{Q}^{-1}\mathbf{A}$, și $\mathbf{A}' = \mathbf{Q}^{-1}\mathbf{AQ}$.

Algoritmul *QR* realizează factorizări succesive ale șirului de matrici $\{\mathbf{A}_k\}$, unde

$\mathbf{A}_1 = \mathbf{A}$, definite de:

$$\mathbf{A}_1 = \mathbf{Q}_1\mathbf{R}_1; \quad \mathbf{A}_2 = \mathbf{R}_1\mathbf{Q}_1;$$

$$\mathbf{A}_2 = \mathbf{Q}_2\mathbf{R}_2; \quad \mathbf{A}_3 = \mathbf{R}_2\mathbf{Q}_2;$$

...

$$\mathbf{A}_k = \mathbf{Q}_k\mathbf{R}_k; \quad \mathbf{A}_{k+1} = \mathbf{R}_k\mathbf{Q}_k;$$

...

Matricile \mathbf{A}_k , și $\mathbf{Q}_k, \mathbf{R}_k$, au următoarele proprietăți:

- 1) Toate \mathbf{A}_k au aceleași valori proprii ca și \mathbf{A} .
- 2) Dacă \mathbf{A} este *simetrică*, sau *tridiagonală*, sau \mathbf{A} are forma *Hessenberg superioară*, matricile \mathbf{A}_k vor păstra aceste forme.
- 3) Fie $\bar{\mathbf{Q}}_k = \mathbf{Q}_1 \mathbf{Q}_2 \dots \mathbf{Q}_k$, atunci (prin inducție): $\mathbf{A}_{k+1} = \bar{\mathbf{Q}}_k^{-1} \mathbf{A}_1 \bar{\mathbf{Q}}_k$.
- 4) Fie $\bar{\mathbf{R}}_k = \mathbf{R}_k \dots \mathbf{R}_1$, atunci: $\bar{\mathbf{Q}}_k \bar{\mathbf{R}}_k = (\mathbf{A}_1)^k$

Matrici Hessenberg și Householder

Forma **Hessenberg** (superioară) este o matrice având elementele zero sub sub-diagonală. De exemplu, o matrice Hessenberg 6×6 arată astfel:

$$\mathbf{A} = \begin{bmatrix} a_{11} & \dots & & & & a_{16} \\ a_{21} & a_{22} & \dots & & & \vdots \\ 0 & a_{32} & a_{33} & \dots & & \\ 0 & 0 & a_{43} & a_{44} & \dots & \\ 0 & 0 & 0 & a_{54} & a_{55} & \\ 0 & 0 & 0 & 0 & a_{65} & a_{66} \end{bmatrix}$$

■

O matrice **Householder** $n \times n$ este definită prin

$$\mathbf{P} = \mathbf{I} - 2\mathbf{w}\mathbf{w}^T,$$

unde \mathbf{I} este matricea unitate $n \times n$, iar \mathbf{w} este un vector cu n coordonate, normalizat relativ la norma euclidiană, adică $\|\mathbf{w}\|_2 = 1$, sau $\mathbf{w}^T \mathbf{w} = 1$, și altfel arbitrar.

Explicit,

$$\mathbf{P} = \begin{bmatrix} 1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1 \end{bmatrix} - 2 \begin{bmatrix} w_1 \\ \vdots \\ w_i \\ \vdots \\ w_n \end{bmatrix} [w_1 \quad \dots \quad w_j \quad \dots \quad w_n]$$

$$p_{ij} = \delta_{ij} - 2w_i w_j$$

Matricea \mathbf{P} este *simetrică* și *este propria sa inversă* (este unitară), întrucât avem

$$\mathbf{P}^T = \mathbf{P}, \text{ și } \mathbf{P}^{-1} = \mathbf{P}.$$

Dacă se consideră un vector *arbitrar* \mathbf{v} , definim

$$\mathbf{w} = \mathbf{v} / \|\mathbf{v}\|_2,$$

și \mathbf{P} devine:

$$\mathbf{P} = \mathbf{I} - 2 \frac{\mathbf{w}\mathbf{w}^T}{\|\mathbf{v}\|_2^2} \quad (1)$$

Sau, notând

$$H = \frac{1}{2} \|\mathbf{v}\|_2^2 = \frac{1}{2} \mathbf{v}^T \mathbf{v} = \frac{1}{2} \langle \mathbf{v}, \mathbf{v} \rangle, \quad (2)$$

$$\mathbf{P} = \mathbf{I} - \frac{\mathbf{w}\mathbf{w}^T}{H}$$

Proprietate

Pentru un vector dat \mathbf{x} , vrem să alegem \mathbf{v} astfel ca să avem

$$\mathbf{P}\mathbf{x} = \mu \mathbf{e}^{(1)},$$

unde $\mathbf{e}^{(1)}$ este prima coloană a matricii unitate $n \times n$. Adică:

$$\mathbf{P}\mathbf{x} = \begin{bmatrix} \mu \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Cu alte cuvinte, \mathbf{P} rotește \mathbf{x} pe direcția $\mathbf{e}^{(1)}$. Aceasta se realizează luând

$$\mathbf{v} = \frac{\mathbf{x}}{\|\mathbf{x}\|_2} \mathbf{e}^{(1)}.$$

Rezultă

$$\mathbf{P}\mathbf{x} = \frac{\mathbf{x}}{\|\mathbf{x}\|_2} \mathbf{e}^{(1)}$$

(Proprietatea se demonstrează astfel: se calculează $\langle \mathbf{v} - \mathbf{x}, \mathbf{v} - \mathbf{x} \rangle$ și se arată că avem $\langle \mathbf{v}, \mathbf{v} \rangle = 2 \langle \mathbf{v}, \mathbf{x} \rangle$; se calculează apoi, $\mathbf{P}\mathbf{x}$.)

■

1.2 Algoritm:

Algoritmul parcurge următoarele etape:

- I. Reducerea lui \mathbf{A} la forma tridiagonală dacă \mathbf{A} este simetrică, sau la forma Hessenberg dacă \mathbf{A} este nesimetrică.
- II. Reducerea lui \mathbf{A} la forma triunghiulară, sau bloc- triunghiulară (v. mai jos).
- III. Descompunerea \mathbf{QR} .

Algoritmul QR propriu-zis, constă în Etapele II și III.

a) Etapa I se realizează înainte de algoritmul QR pentru a aduce matricea \mathbf{A} la o forma care reduce efortul de calcul. Aceasta, pentru că *numărul de operații* pe un pas k al iterației QR (Etapele II și III), aplicată unei matrici $n \times n$ este de ordinul (Ralston & Rabinowitz, 1983):

- n^3 – pentru o matrice generală;
- n^2 – pentru o matrice Hessenberg;
- n – pentru o matrice tridiagonală.

Etapa I se realizează prin pre- și post-multiplicare cu matrici Householder \mathbf{P}_k .

b) Etapa II se realizează prin premultiplicare cu matrici Householder \mathbf{P}_k , cum urmează:

$$\mathbf{A}_1 = \mathbf{P}_1 \mathbf{A}; \quad \mathbf{A}_2 = \mathbf{P}_2 \mathbf{A}_1 = \mathbf{P}_2 \mathbf{P}_1 \mathbf{A}; \quad \dots; \quad \mathbf{A}_{n-1} = \mathbf{P}_{n-1} \dots \mathbf{P}_2 \mathbf{P}_1 \mathbf{A}.$$

Matricea \mathbf{A}_{n-1} are forma triunghiulară și este matricea \mathbf{R} .

Într-adevăr: Să notăm $\mathbf{Q} = \mathbf{P}_1 \mathbf{P}_2 \dots \mathbf{P}_{n-1}$, avem $\mathbf{A}_{n-1} = \mathbf{Q}^T \mathbf{A}$. \mathbf{Q} este o matrice ortogonală, deci avem $\mathbf{Q} \mathbf{A}_{n-1} = \mathbf{A}$. Cum \mathbf{Q} este ortogonală și \mathbf{A}_{n-1} este superior triunghiulară, urmează că $\mathbf{R} = \mathbf{A}_{n-1}$.

■

Atunci, Etapele II și III se realizează prin următorii pași de iterare:

1) Pune $\mathbf{A}^{(1)} = \mathbf{A}$

- Triangularizează $\mathbf{A}^{(1)}$:

$$\mathbf{A}_1^{(1)} = \mathbf{P}_1^{(1)} \mathbf{A}; \quad \mathbf{A}_2^{(1)} = \mathbf{P}_2^{(1)} \mathbf{A}_1^{(1)}; \quad \dots; \quad \mathbf{A}_{n-1}^{(1)} = \mathbf{P}_{n-1}^{(1)} \mathbf{A}_{n-2}^{(1)}$$

- La sfârșitul pasului, avem:

$$\mathbf{R}^{(1)} = \mathbf{A}_{n-1}^{(1)}; \quad \mathbf{Q}^{(1)} = \mathbf{P}_1^{(1)} \mathbf{P}_2^{(1)} \dots \mathbf{P}_{n-1}^{(1)}.$$

2) Calculează $\mathbf{A}^{(2)} = \mathbf{R}^{(1)} \mathbf{Q}^{(1)}$

- Triangularizează $\mathbf{A}^{(2)}$:

$$\mathbf{A}_1^{(2)} = \mathbf{P}_1^{(2)} \mathbf{A}^{(2)}; \quad \mathbf{A}_2^{(2)} = \mathbf{P}_2^{(2)} \mathbf{A}_1^{(2)}; \quad \dots; \quad \mathbf{A}_{n-1}^{(2)} = \mathbf{P}_{n-1}^{(2)} \mathbf{A}_{n-2}^{(2)}$$

- Pune:

$$\mathbf{R}^{(2)} = \mathbf{A}_{n-1}^{(2)}; \quad \mathbf{Q}^{(2)} = \mathbf{P}_1^{(2)} \mathbf{P}_2^{(2)} \dots \mathbf{P}_{n-1}^{(2)}.$$

...

k) Calculează $\mathbf{A}^{(k)} = \mathbf{R}^{(k-1)} \mathbf{Q}^{(k-1)}$ ($k \geq 2$)

- Triangularizează $\mathbf{A}^{(k)}$:

$$\mathbf{A}_1^{(k)} = \mathbf{P}_1^{(k)} \mathbf{A}^{(k)}; \quad \mathbf{A}_2^{(k)} = \mathbf{P}_2^{(k)} \mathbf{A}_1^{(k)}; \quad \dots; \quad \mathbf{A}_{n-1}^{(k)} = \mathbf{P}_{n-1}^{(k)} \mathbf{A}_{n-2}^{(k)}$$

- Pune:

$$\mathbf{R}^{(k)} = \mathbf{A}_{n-1}^{(k)}; \quad \mathbf{Q}^{(k)} = \mathbf{P}_1^{(k)} \mathbf{P}_2^{(k)} \dots \mathbf{P}_{n-1}^{(k)}$$

unde $\mathbf{B}_j = \begin{bmatrix} b_{11}^{(j)} & b_{12}^{(j)} \\ b_{21}^{(j)} & b_{22}^{(j)} \end{bmatrix}$, iar \mathbf{C} conține elementele superioare. Avem $r + 2s = n$.

Valorile λ_i și blocurile \mathbf{B}_j , pot apare în orice ordine pe diagonală.

Notă

Cazul 3 se poate evita în practică, prin algoritmul QR cu deplasare – v. mai jos

■

1.3 Algoritmul QR cu deplasare:

Presupunem că \mathbf{A} are valori proprii *de module distincte* $|\lambda_1| > |\lambda_2| > \dots > |\lambda_n| > 0$.

Se arată că viteza de convergență la zero a elementelor sub-diagonale, și cea de convergență la valorile proprii a elementelor diagonale, depind de rapoartele $(\lambda_{i+1} / \lambda_i)^k$, $1 \leq i \leq n-1$.

Dacă două valori proprii λ_i și λ_{i+1} sunt apropiate una de alta, convergența va fi încetată. Atunci, se utilizează următoarea tehnică pentru a accelera convergența. Se aplică o deplasare s_k la valorile proprii (acestea devin $\lambda_i - s_k$), la fiecare etapă k .

Exemplu: $\lambda_i = 2.2$, $\lambda_{i+1} = 2.1$; avem $\lambda_{i+1} / \lambda_i = 0.95$. Cu $s = 2.0$, raportul are valoarea $0.1/0.2 = 0.5$.

Adică, punem

$$\mathbf{A}_1 = \mathbf{A}, \text{ și}$$

$$\mathbf{A}^{(k)} - s_k \mathbf{I} = \mathbf{Q}^{(k)} \mathbf{R}^{(k)},$$

$$\mathbf{A}^{(k+1)} = \mathbf{R}^{(k)} \mathbf{Q}^{(k)} + s_k \mathbf{I}$$

Există două strategii pentru a alege deplasarea s_k (s_k est o aproximație a lui λ_n):

$$1) \quad s_k = a_{nn}^{(k)}$$

Pentru o *matrice simetrică* (reală), această strategie asigură o convergență cubică, chiar în prezența unor valori proprii multiple (Wilkinson, 1965).

2) Se calculează valorile proprii ale submatricii 2×2 cea mai de jos din $\mathbf{A}^{(k)}$:

$$\begin{bmatrix} a_{n-1,n-1}^{(k)} & a_{n-1,n}^{(k)} \\ a_{n,n-1}^{(k)} & a_{nn}^{(k)} \end{bmatrix}$$

Dacă valorile proprii sunt reale, se alege:

$$s_k = \text{valoarea proprie care este cea mai apropiată de } a_{nn}^{(k)}.$$

Dacă valorile proprii sunt complexe, se ia:

$$s_k = \text{partea reală a valorii proprii.}$$

3) Experimentele numerice au condus la o a treia strategie, și anume:

$$s_k = \min(a_{ii}^{(k)}), \quad i = \overline{1, n}$$

Pentru unele matrici simetrice, această strategie se dovedește cea mai bună, conducând la un număr mai mic de iterații, și o eroare mai mică în $\mathbf{Ay} - \lambda\mathbf{y}$.

1.4 Programul QR

Programul QR din ANA, calculează sistemul propriu (valori și vectori proprii), în succesiunea următoare:

- I. Transformarea preliminară a matricii \mathbf{A} (Opțional).
- II. Valorile proprii, prin reducerea matricii la forma bloc-triunghiulară. (Programul recunoaște numai blocuri 2×2 .)
- III. Vectorii proprii (Opțional).

Se calculează, mai întâi, vectorii proprii ai matricii bloc-triunghiulare. Aceștia se aduc, prin transformarea de similaritate, la vectorii proprii ai lui \mathbf{A} .

IV. Rafinarea sistemului propriu prin iterație inversă (Opțional).

V. Verificarea sistemului propriu (Opțional):

Precizia sistemului propriu se verifică prin listarea erorii maxime (în modul), în $\mathbf{Ay} - \lambda\mathbf{y}$ (unde λ este valoarea proprie, iar \mathbf{y} vectorul propriu asociat cu λ).

Exemplu – 1: Matrice simetrică

Considerăm matricea:

$$\mathbf{A} = \begin{bmatrix} 1 & -3 & -2 & 1 \\ -3 & 10 & -3 & 6 \\ -2 & -3 & 3 & -2 \\ 1 & 6 & -2 & 1 \end{bmatrix}$$

Programul QR, rulat cu $eps = 1E-7$ (toleranță pentru elementele non-diagonale), $eps_lambda = 1E-6$ (toleranța la λ), *deplasare* – strategia 1, produce următoarele rezultate (matricile sunt listate în format 1pG15.7, iar valorile proprii în format 1pG24.16):

Matrix A - Tridiagonal form:

1.000000	3.741657	-4.0811097E-16	2.8277989E-16
3.741657	2.785714	-5.246476	0.000000
-4.0811097E-16	-5.246476	10.19927	-4.479586
2.8277989E-16	0.000000	-4.479586	1.015014

Iteration 19:

Reduced Matrix - Block-Triangular Form

14.32951	8.3206414E-08	-1.0862088E-10	6.0791490E-16
8.3206414E-08	4.456957	-3.9733936E-03	-2.9646362E-11
-1.0862122E-10	-3.9733936E-03	-3.415088	-2.2709916E-08
-8.1045051E-19	-2.9646498E-11	-2.2709916E-08	-0.3713752

Eigenvalues

1	14.32950642539378
2	-3.415090280621962
3	4.456959098788067
4	-0.3713752435599103

Programul calculează și vectorii proprii \mathbf{y} . Diferența maximă în $\mathbf{A}\mathbf{y} - \lambda\mathbf{y}$ este 5.766E-08.

Programul rulat *cu deplasare* – strategia 3, termină în 9 iterații, cu diferența maximă $1.228\text{E-}12$. Pentru acest exemplu, strategia 3 de alegere a deplasării este cea mai bună.

■

Exemplu – 1 bis: Rafinare

Considerăm matricea și parametrii din Exemplu-1, și facem rafinarea sistemului propriu cu $ns = 4$. Se obțin rezultatele (valorile proprii sunt listate în format `1pg24.16`):

	Iterations	Test Value	Tolerance
1	4	4.4408920985E-16	1.776E-15
2	4	2.7194799110E-16	4.441E-16
3	3	0.000000000	8.882E-16
4	4	1.5324295440E-14 (stationary)	

Eigenvalues

1	14.32950642539381
2	-3.415090280621964
3	4.456959098788065
4	-0.3713752435599113

Diferența maximă în $\mathbf{Ay} - \lambda\mathbf{y}$ este acum, $1.110\text{E-}15$.

■

Exemplu – 2: Matrice nesimetrică

Considerăm matrice:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 4 & 5 & 6 & 7 \\ 2 & 1 & 5 & 0 \\ 4 & 2 & 1 & 0 \end{bmatrix}$$

Programul QR, cu $eps = 1E-7$, $eps_lambda = 1E-8$, *deplasare* – strategia 1, produce rezultatele:

Matrix A - Hessenberg form:

1.000000	-5.000000	-1.539516	-1.276672
-6.000000	8.555556	0.1084754	3.698593
0.000000	-5.918166	-2.168879	-1.142756
0.000000	0.000000	-0.1427564	3.613324

Iteration 24:

Reduced Matrix - Block-Triangular Form

4.153893	10.18835	2.745848	2.753036
5.464962	3.096039	3.428772	2.606012
6.2957059E-08	-5.5017566E-09	0.1764519	0.1516238
-2.9451447E-16	2.5737335E-17	1.5892009E-08	3.573617

Eigenvalues

1	-3.855588238007097
2	11.10551971006653
3	0.1764519119325265
4	3.573616616008061

Diferența maximă în $Ay - \lambda y$ este 4.403E-08.

Cu strategia 3, programul termină în 22 iterații și diferența maximă este 8.755E-08.

■

Exemplu – 2 bis: Rafinare

Considerăm matricea și parametrii din Exemplu-2, și facem rafinarea sistemului propriu cu $ns = 4$. Se obțin rezultatele:

	Iterations	Test Value	Tolerance
1	4	3.1401849174E-16	4.441E-16
2	4	2.2204460493E-16	1.776E-15
3	5	0.000000000	2.776E-17

4 5 0.000000000 4.441E-16

Eigenvalues

1 -3.855588220333913

2 11.10551973067810

3 0.1764518729384592

4 3.573616616717359

Diferența maximă în $\mathbf{A}\mathbf{y} - \lambda\mathbf{y}$ este 1.776E-15.

■

2 Problema generalizată

2.1 Problema

Problema determinării valorilor și vectorilor proprii ai unei matrici \mathbf{A} , definită de $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$, sau de ecuația

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = \mathbf{0}, \quad (1)$$

se va numi problema *standard*.

În Dinamica Structurilor apare următoarea problemă:

$$(\mathbf{K} - \lambda\mathbf{M})\mathbf{x} = \mathbf{0}, \quad (2)$$

unde: $\lambda = \omega^2$, iar \mathbf{K} și \mathbf{M} sunt matrice simetrice și pozitiv definite (\mathbf{K} este matricea de rigiditate, \mathbf{M} matricea de masă sau de inerție, și ω este pulsația proprie).

Problema (2) se zice problema *generalizată* de valori proprii. Mai general considerând două matrici $n \times n$ \mathbf{A} și \mathbf{B} , unde \mathbf{B} este nesingulară, problema generalizată are forma

$$(\mathbf{A} - \lambda\mathbf{B})\mathbf{x} = \mathbf{0} \quad (2')$$

În continuare, vom trata problema (2).

2.2 Reducerea la problema standard

Problema (2) poate fi adusă la problema standard (1), prin înmulțirea la stânga a ecuației (2) cu matricea \mathbf{K}^{-1} și notând $\mathbf{D} = \mathbf{K}^{-1}\mathbf{M}$, adică,

$$\left(\mathbf{D} - \frac{1}{\lambda}\mathbf{I}\right)\mathbf{x} = \mathbf{0}$$

Dezavantajul acestei formulări este că, în general, matricea \mathbf{D} nu mai este simetrică și astfel, metodele pentru matrici simetrice (de exemplu, Jacobi, iterații simultane) – nu mai pot fi utilizate.

Pe de altă parte:

- Valorile proprii ale unei matrici simetrice sunt bine-condiționate, pe când cele ale unei matrici nesimetrice pot fi rău-condiționate.
- La utilizarea metodei QR, numărul de operații (pe un pas al iterației QR) pentru o matrice simetrică este mult mai mic decât pentru o matrice generală.

Ținând cont de definiția pozitivă a matricii \mathbf{M} , problema generalizată (2) poate fi transformată în problema standard pentru o matrice simetrică cum urmează.

1. Se face descompunerea Cholesky a lui \mathbf{M} :

$$\mathbf{M} = \mathbf{S}^T \mathbf{S}, \quad (3)$$

unde \mathbf{S} este o matrice superior triunghiulară. (Descompunerea inferior triunghiulară poate fi de asemenea utilizată). Atunci, (2) scrisă în forma

$$\mathbf{K}\mathbf{x} = \lambda\mathbf{M}\mathbf{x} \quad (4)$$

devine $\mathbf{K}\mathbf{x} = \lambda\mathbf{S}^T \mathbf{S}\mathbf{x}$, și se scrie din nou ca și

$$\mathbf{K}\mathbf{S}^{-1}(\mathbf{S}\mathbf{x}) = \lambda\mathbf{S}^T(\mathbf{S}\mathbf{x}).$$

2. Se notează

$$\mathbf{y} = \mathbf{S}\mathbf{x} \quad (5)$$

și ecuația precedentă devine

$$\mathbf{KS}^{-1}\mathbf{y} = \lambda\mathbf{S}^T\mathbf{y}.$$

Premultiplicând cu $(\mathbf{S}^T)^{-1} = (\mathbf{S}^{-1})^T = \mathbf{S}^{-T}$ (ultima egalitate este o notație), obținem

$$\mathbf{S}^{-T}\mathbf{KS}^{-1}\mathbf{y} = \lambda\mathbf{I}\mathbf{y}$$

3. Acum, definim

$$\mathbf{R} = \mathbf{S}^{-T}\mathbf{KS}^{-1} \quad (6)$$

\mathbf{R} este o matrice *simetrică și pozitiv definită* – v. mai jos.

4. Problema (2) devine problema standard pentru matricea \mathbf{R} , și anume,

$$(\mathbf{R} - \lambda\mathbf{I})\mathbf{y} = \mathbf{0} \quad (7)$$

Problemele (7) și (2) au aceleași valori proprii.

5. După rezolvarea lui (7) pentru λ și \mathbf{y} , vectorii proprii ai problemei originale (2) sunt dați (v. (5)) de

$$\mathbf{x} = \mathbf{S}^{-1}\mathbf{y} \quad (8)$$

■

Se verifică imediat că matricea \mathbf{R} este simetrică și pozitiv definită, la fel cum sunt matricile \mathbf{M} și \mathbf{K} . Într-adevăr, avem:

$$\mathbf{R}^T = (\mathbf{S}^{-T}\mathbf{KS}^{-1})^T = \mathbf{S}^{-T}\mathbf{K}^T\mathbf{S}^{-1} = \mathbf{S}^{-T}\mathbf{KS}^{-1} = \mathbf{R}$$

Apoi, pentru orice $\mathbf{x} \neq \mathbf{0}$, avem:

$$\mathbf{x}^T\mathbf{R}\mathbf{x} = \mathbf{x}^T(\mathbf{S}^{-T}\mathbf{KS}^{-1})\mathbf{x} = (\mathbf{S}^{-1}\mathbf{x})^T\mathbf{K}(\mathbf{S}^{-1}\mathbf{x}) = \mathbf{u}^T\mathbf{K}\mathbf{u} > 0,$$

întrucât $\mathbf{u} = \mathbf{S}^{-1}\mathbf{x}$ este arbitrar și nonzero, la fel ca și \mathbf{x} .

Urmează că valorile proprii ale lui \mathbf{R} sunt reale și pozitive, așa cum trebuie să avem conform cu $\lambda_i = \omega_i^2$. Astfel, pentru a rezolva (7), orice metodă pentru problema de valori proprii ale unei matrici simetrice și pozitiv definite poate fi aplicată lui \mathbf{R} .

Observație

Cel mai simplu caz este acela în care matricea \mathbf{M} este diagonală (model de mase concentrate), $\mathbf{M} = \text{diag}(m_i)$. Atunci, avem $\mathbf{S} = \mathbf{S}^T = \text{diag}(\sqrt{m_i})$, și

$\mathbf{S}^{-1} = \mathbf{S}^{-T} = \text{diag}(1/\sqrt{m_i})$. Astfel, elementele lui \mathbf{R} și \mathbf{x} sunt date de:

$$r_{ij} = \frac{k_{ij}}{\sqrt{m_i} \sqrt{m_j}}, \quad i, j = \overline{1, n},$$

și

$$x_i = \frac{y_i}{\sqrt{m_i}}, \quad i = \overline{1, n}.$$

Dacă \mathbf{M} nu este diagonală: notând $\mathbf{Z} = \mathbf{S}^{-1}$, matricea \mathbf{Z} se calculează ușor din

$\mathbf{S}\mathbf{Z}^{(j)} = \mathbf{e}^{(j)}$, $j = \overline{1, n}$ prin substituție înapoi, întrucât \mathbf{S} este superior triunghiulară.

Apoi, $\mathbf{R} = \mathbf{Z}^T \mathbf{K} \mathbf{Z}$ și $\mathbf{x} = \mathbf{Z} \mathbf{y}$.

■

Exemplu

Fie matricile:

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2.5 \end{bmatrix}; \quad \mathbf{K} = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 3 & -2 \\ 0 & -2 & 6 \end{bmatrix}$$

Avem:

$$\mathbf{S} = \begin{bmatrix} 1 & & \\ & \sqrt{2} & \\ & & \sqrt{2.5} \end{bmatrix}; \quad \mathbf{S}^{-1} = \begin{bmatrix} 1 & & \\ & 1/\sqrt{2} & \\ & & 1/\sqrt{2.5} \end{bmatrix}$$

Rezultă:

$$\mathbf{R} = \begin{bmatrix} 1 & -1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1.5 & -2/\sqrt{5} \\ 0 & -2/\sqrt{5} & 2.4 \end{bmatrix}$$

Vectorii proprii ai lui \mathbf{R} sunt calculați analitic, din sistemul (7). Se obține:

$$\mathbf{y} = \begin{bmatrix} \sqrt{2/5} \\ 2/\sqrt{5}(1-\lambda) \\ \lambda^2 - 2.5\lambda + 1 \end{bmatrix}$$

Sau, împărțind cu prima coordonată

$$\mathbf{y} = \begin{bmatrix} 1 \\ \sqrt{2}(1-\lambda) \\ \sqrt{2.5}(\lambda^2 - 2.5\lambda + 1) \end{bmatrix}$$

Cu acesta, se obține

$$\mathbf{x} = \mathbf{S}^{-1}\mathbf{y} = \begin{bmatrix} 1 \\ 2(1-\lambda) \\ \lambda^2 - 2.5\lambda + 1 \end{bmatrix}$$

Valorile proprii ai lui \mathbf{R} se pot calcula rezolvând ecuația caracteristică

$$p(\lambda) = -\lambda^3 + 4.9\lambda^2 - 6.2\lambda + 1.6 = 0. \text{ Se obține (în simplă precizie):}$$

$$\lambda_1 = 3.025604; \lambda_2 = 1.528400; \lambda_3 = 0.3459958.$$

■

2.3 Ortogonalitate

Se poate arăta că vectorii proprii \mathbf{x} ai problemei (2), sunt ortogonali relativ la ambele matrici \mathbf{K} și \mathbf{M} .

1) \mathbf{M} -ortogonalitate:

Vectorii \mathbf{y} sunt ortogonali, adică,

$$\mathbf{y}^{(i)T}\mathbf{y}^{(j)} = 0, \quad \text{pentru } i \neq j.$$

Substituind $\mathbf{y} = \mathbf{S}\mathbf{x}$ rezultă $\mathbf{x}^{(i)T}\mathbf{S}^T\mathbf{S}\mathbf{x}^{(j)} = 0$, sau, ținând cont de (3),

$$\mathbf{x}^{(i)T}\mathbf{M}\mathbf{x}^{(j)} = 0. \tag{9}$$

2) \mathbf{K} -ortogonalitate:

Vectorii \mathbf{y} sunt ortogonali relativ la matricea \mathbf{R} , adică

$$\mathbf{y}^{(i)T} \mathbf{R} \mathbf{y}^{(j)} = 0, \quad \text{pentru } i \neq j.$$

Cum $\mathbf{y} = \mathbf{S} \mathbf{x}$, urmează că $\mathbf{x}^{(i)T} \mathbf{S}^T \mathbf{R} \mathbf{S} \mathbf{x}^{(j)} = 0$. Din (6) avem $\mathbf{S}^T \mathbf{R} \mathbf{S} = \mathbf{K}$, și astfel, obținem

$$\mathbf{x}^{(i)T} \mathbf{K} \mathbf{x}^{(j)} = 0, \quad (10)$$

care probează ortogonalitatea relativ la \mathbf{K} . Concluzia (10) se poate obține și din (4), premultiplicând cu $\mathbf{x}^{(i)T}$ și utilizând (9).

În final, să introducem matricea vectorilor proprii

$$\mathbf{X} = [\mathbf{x}^{(1)} \mid \mathbf{x}^{(2)} \mid \dots \mid \mathbf{x}^{(n)}] = \begin{bmatrix} x_1^{(1)} & x_1^{(2)} & \dots & x_1^{(n)} \\ x_2^{(1)} & x_2^{(2)} & \dots & x_2^{(n)} \\ \vdots & \vdots & \vdots & \vdots \\ x_n^{(1)} & x_n^{(2)} & \dots & x_n^{(n)} \end{bmatrix},$$

și să presupunem că \mathbf{X} este normalizată, adică $\|\mathbf{x}^{(i)}\|_2 = 1$. Să notăm:

$$\mathbf{x}^{(i)T} \mathbf{M} \mathbf{x}^{(i)} = M_i, \quad (11)$$

$$\mathbf{x}^{(i)T} \mathbf{K} \mathbf{x}^{(i)} = K_i. \quad (12)$$

Se pot da formule explicite pentru M_i și K_i . Observați că, în acord cu definiția pozitivă a lui \mathbf{M} și \mathbf{K} , avem $M_i > 0$ și $K_i > 0$. Mai mult, din (4) obținem $K_i = \lambda M_i$.

Cu (11) și (12), relațiile de ortogonalitate (9) și (10) pot fi scrise, respectiv, ca

$$\mathbf{X}^T \mathbf{M} \mathbf{X} = \text{diag}(M_i),$$

și

$$\mathbf{X}^T \mathbf{K} \mathbf{X} = \text{diag}(K_i).$$

■

*NOTE**1. Algoritm și Rutine*

Rutinele pentru calculul pulsațiilor și vectorilor proprii prin metoda de mai sus, se găsesc în folderele din \ANA\EIGEN\ – Biblioteca ANA .

Calculul va parcurge următorii pași:

- 1) Calculul matricii **R**: \Eigen\General_R (sau General_R1)
- 2) Calculul valorilor și vectorilor proprii ai matricii **R**:
 \Eigen\QR; \Eigen\Jacobi_D; \Eigen\Sim_Iter_D;
- 3) Regăsirea valorilor și vectorilor proprii ai problemei generalizate:
 \Eigen\Retrieve_Eigen_from_R

Toate rutinele menționate lucrează în dublă precizie.

2. Note pentru Analiza structurilor

- Matricea de rigiditate **K** va fi furnizată de programul de analiză statică – cu încărcările date de greutatea maselor.
- Matricea de masă **M** va fi:
 - Generată direct, pentru mase concentrate;
 - Furnizată de programul de analiză dinamică, pentru mase distribuite. (Acesta va fi rulat pe un interval mic, cu condiții inițiale nenule, și încărcări nule.)

3. Unități de măsură

Pentru ca pulsațiile să rezulte în s^{-1} trebuie ca elementele matricii **M** să fie în kg , iar cele

ale matricii **K** în $\frac{N}{m}$.

Mai general, în unități compatibile, astfel ca ω^2 să rezulte în $1/s^2$, iar ω în $1/s$:

Luând forța (F) sau lungimea (L) în alte unități (decît unitățile SI), masa va fi mărime derivată, anume:

$$M = \frac{F}{L/t^2} .$$

Programul `General_R` cere, la citirea din fișier a matricilor **M** și **K**, introducerea unor factori care înmulțesc aceste matrici. Acești factori vor asigura transformarea unităților folosite la generarea matricilor, în unități SI (sau, în unități compatibile).

4. *Pulsații sau frecvențe*

În particular, `General_R` permite opțiunea de a calcula pulsații sau frecvențe. Frecvențele sunt legate de pulsații prin

$$f_i = \omega_i / (2\pi) \quad [\text{rad/s}]$$

Perioadele de vibrație sunt:

$$T_i = \frac{1}{f_i} \quad [\text{s}]$$

■