5.2.4 Analiză matriceală

Majoritatea problemelor abordate matriceal presupun evaluarea unor proprietăţi şi testarea anumitor condiţii cu privire la matricele componente. În continuare se prezintă prin exemple funcţiile Matlab pentru analiză matriceală.

a) Calculul determinantului asociat unei matrice

Cu funcţia `det` se calculează determinantul unei matrice (pătrate) date cu instrucţiuni de forma `D=det(X)`, de exemplu:

```
>> X=[1 2 1;3 4 0;7 9 -8]

X =
     1     2     1
     3     4     0
     7     9    -8

>> D=det(X)

D =
    15
```

b) Calculul inversei unei matrice

Cu funcţia `inv` se calculează inversa unei matrice date: `Y=inv(X)` adică $Y=X^{-1}$. Prin definiţie, inversa unei matrici pătrate X este matricea X^{-1}, care satisface relaţia:

$$X \cdot X^{-1} = X^{-1} \cdot X = I,$$

unde I este matricea identitate. O matrice este inversabilă numai dacă este nesingulară, adică determinantul ei este diferit de zero.

De exemplu, pentru matricea X definită anterior se obţine:

```
>> Y=inv(X)

Y =
   -2.1333    1.6667   -0.2667
    1.6000   -1.0000    0.2000
   -0.0667    0.3333   -0.1333
```

c) Rangul unei matrice

Cu funcția `rank` se calculează rangul unei matrice date, cu sintaxa `r=rank(X)` sau `r=rank(X,tol)`.

Rangul unei matrice este dimensiunea/ordinul celui mai mic determinant nenul, ce poate fi izolat în matricea respectivă. Se spune că o matrice nenulă are rangul k, dacă aceasta are un minor nenul de ordin k, iar toți minorii de ordin mai mare decât k (dacă există) sunt nuli. Rangul unei matrice se interpretează ca numărul de linii sau coloane liniar independente ale acesteia şi este un număr scalar cu proprietatea: $1 \leq k \leq \min(m, n)$.

Apelată cu cea de-a doua sintaxă funcția `rank` returnează numărul de *valori singulare* ale matricei `X`, mai mari decât parametrul opțional `tol`. Acestea reprezintă un mijloc sigur pentru determinarea rangului unei matrice de formă generală. Iată câteva exemple de aplicare a funcției `rank`:

```
>>X=[1 2 1;3 4 0;7 9 -8]

X =
     1     2     1
     3     4     0
     7     9    -8
```

```
>> r=rank(X)

r =
     3
```

```
>> r1=rank(X,2)

r1 =
     2
```

```
>> r2=rank(X,10)

r2 =
     1
```

```
>> r3=rank(X,100)

r3 =
     0
```

d) Calculul valorilor singulare ale unei matrice

Numărul valorilor singulare reprezintă rangul unei matrice. Deci, calculul valorilor singulare constituie un mijloc sigur de determinare a rangului unei matrice.

Funcția `svd` se poate apela cu diferite sintaxe astfel:

`d=svd(X)`,

`[U,S,V]=svd(X)`,

sau `[U,S,V]=svd(X,0)`, pentru calcul mai rapid al valorilor singulare.

Potrivit sintaxelor de mai sus funcția returnează:

- d vector ce conține valorile singulare;
- S matrice diagonală cu dimensiunea lui X având elemente diagonale nenegative (care sunt şi valorile singulare) în ordine descrescătoare;
- U,V matrici care satisfac relația `X=U*S*V'`.

```
>> X=[1 2 1;3 4 0;7 9 -8;10 3 6]

X =
     1     2     1
     3     4     0
     7     9    -8
    10     3     6


>> r=rank(X)

r =
     3


>> d=svd(X)

d =
   15.8129
   10.7584
    2.0516


>> [U,S,V]=svd(X)

U =
   -0.1154   -0.0819    0.6711   -0.7278
   -0.3028   -0.0384    0.6734    0.6733
   -0.7934    0.5610   -0.2015   -0.1232
   -0.5153   -0.8229   -0.2356   -0.0430

S =
  15.8129          0          0
        0    10.7584          0
        0          0     2.0516
        0          0          0

V =
   -0.7418   -0.4182   -0.5242
   -0.6405    0.2103    0.7386
    0.1986   -0.8837    0.4239
```

e) Urma unei matrice

Urma unei matrice se defineşte ca *suma elementelor de pe diagonala principală* şi se calculează cu funcţia `trace` cu sintaxa `u=trace(A)`.

De exemplu, urma matricei X folosită în exemplele anterioare este:

```
>> u=trace(X)

u =
    -3
```

iar pentru inversa ei $Y=X^{-1}$ se obține:

```
>> u=trace(Y)

u =
   -3.2667
```

f) Calculul normelor pentru vectori şi matrice

Norma este un scalar care dă o măsură (o anumită evaluare) a *mărimii* elementelor unui vector sau matrice. Există diferite tipuri de norme care se aplică vectorilor şi matricelor şi pentru care Matlab pune la dispoziție funcții specifice de evaluare, astfel:

a) Pentru matrice se pot aplica următoarele funcții:

- `norm(X)` sau `norm(X,2)` – returnează *norma-2* reprezentând cea mai mare valoare singulară a lui X;

Din punct de vedere matematic norma-2 se defineşte sub forma:

$$\|X\|_2 = \sqrt{\rho(X * X)} = \max_{\lambda \in \sigma(X)} |\lambda|,$$

unde *spectrul matricei* X dat de mulțimea valorilor proprii este notat cu $\sigma(X)$, iar $\rho(X)$ şi λ sunt *raza spectrală* respectiv *valorile singulare* ale matricei X.

- `norm(X,1)` – returnează *norma-1* denumită şi *norma octaedrică*, reprezentând cea mai mare sumă a elementelor de pe coloană, adică:

$$\|X\|_1 = \max_{1 \le j \le N} \left(\sum_{k=1}^{N} |x_{jk}| \right)$$

- `norm(X,inf)` – returnează *norma infinită* reprezentând cea mai mare sumă a elementelor de pe linii, adică:

$$\|X\|_\infty = \max_{1 \le k \le N} \left(\sum_{j=1}^{N} |x_{jk}| \right)$$

- `norm(X,'fro')` – returnează *norma-F* (Frobenius), care se calculează cu relația următoare:

$$\|X\|_F = \sqrt{\sum_{j,k=1}^{N} |x_{jk}|^2} = \sqrt{|x_{11}|^2 + |x_{22}|^2 + \ldots + |x_{NN}|^2}$$

b) Pentru vectori se utilizează următoarele funcții:

- `norm(V,p)` returnează *norma euclidiană* şi se calculează cu expresia:

$$\|V\|_p = \sqrt[p]{\sum_{k=1}^{N} |V_k|^p} = \sqrt[p]{|V_1|^p + |V_2|^p + \ldots + |V_N|^p}$$

- `norm(V)` are acelaşi efect ca `norm(V,2)` prin particularizarea relației de mai sus pentru p=2.

- `norm(V)/sqrt(N)` – returnează valoarea rădăcinii medii pătratice (valoare efectivă) potrivit relației următoare:

$$RMS = \sqrt{\frac{1}{N}\sum_{k=1}^{N} |V_k|^2} = \sqrt{\frac{|V_1|^2 + |V_2|^2 + \ldots + |V_N|^2}{N}}$$

- `norm(V,inf)` – calculează *norma maximum* (sau cubică) şi returnează valoarea maximă în modul (elementul maxim în valoare absolută):

$$\|V\|_\infty = \max |V_k|_{1 \le k \le N}$$

- `norm(V,-inf)` – returnează valoarea minimă în modul (elementul minim în valoare absolută):

$$\|V\|_{-\infty} = \min |V_k|_{1 \le k \le N}$$

Observație. Funcțiile `norm(X)`, `norm(X,p)` şi `norm(X,'fro')` returnează scalari.

Aplicație. Să se calculeze distanța dintre punctele de coordonate A(1,2,0) şi B(2,-1,5).

Rezolvare. Se observă că distanța dintre două puncte în sens euclidian este tocmai norma euclidiană aplicată diferenței vectorilor coordonatelor punctelor date:

$$\|V\|_2 = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$$

Astfel, problema se poate rezolva cu secvența de instrucțiuni următoare:

```
A =
       1     2     0

>> B=[2 -1 5]

B =
       2    -1     5


>> D=A-B

D =
      -1     3    -5


>> d=norm(D)

d =
    5.9161
```

g) Calculul numărului de condiționare al unei matrice

Aspecte teoretice

Numeroase probleme inginereşti se bazează pe modele algebrice liniare. Rezultatele obținute prin calcul numeric corespund în general unei *probleme perturbate*, datorită *erorilor de rotunjire*. Astfel, dacă avem de rezolvat un sistem liniar $A \cdot x = b$ datele A şi b pot suferi perturbațiile δA şi δb, inclusiv datorită erorilor de rotunjire.

În general, există însă numeroase motive de ordin tehnic în baza cărora putem considera că parametrii oricărui sistem sunt afectați de variații, ambiguitate şi imprecizie. Din acest motiv este necesar a se cunoaşte cât de sensibilă este soluția unor sisteme descrise de modele liniare în raport cu variația parametrică. Aceasta se face prin evaluarea *gradului de condiționare* al problemei, pe baza teoriei matematice a perturbațiilor.

Astfel, o problemă *rău condiționată* înseamnă:

$$X \cong X^* \Rightarrow f(X) \neq f(X^*),$$

iar pentru o problemă *bine condiționată* avem:

$$X \cong X^* \Rightarrow f(X) \cong f(X^*).$$

Exemplu aplicativ

Să se evalueze efectul proprietății de condiționare matriceală pentru o problemă de logistică modelată liniar. Fie sistemul de ecuații liniare:

$$\begin{cases} 6x_1 + 6{,}917x_2 = 198{,}34 \\ x_1 + 1{,}152x_2 = 33{,}04 \end{cases}$$

cu următoarea semnificație:

- x_1 – vehicule tip 1, x_2 – vehicule tip 2,
- matricea coeficienților semnifică consumurile specifice celor două categorii de vehicule pe două tipuri de trasee (drum foarte greu – prima ecuație, autostradă – a doua ecuație),
- vectorul coeficienților liberi reflectă consumurile acceptate (cantitățile disponibile) repartizate pentru cele două tipuri de trasee.

Soluția problemei este: `x1=10,x2=20.`

Dacă se consideră *sistemul perturbat* (vezi a treia zecimală de la coeficientul variabilei x_2, din prima ecuație):

$$\begin{cases} 6x_1 + 6{,}91\mathbf{5}x_2 = 198{,}34 \\ x_1 + 1{,}152x_2 = 33{,}04 \end{cases}$$

soluția problemei devine: `x1=16,6667,x2=33,333.`

Se observă astfel, că o mică perturbație parametrică a sistemului a condus la diferențe considerabile asupra soluțiilor problemei. Aceasta denotă o *slabă condiționare* a problemei, fapt ce trebuie cunoscut în prealabil, deoarece poate conduce la anomalii de interpretare a rezultatelor.

Gradul de condiționare al problemei se reflectă în numărul de condiționare asociat matricei sistemului de ecuații liniare, în cazul de față al matricei:

$$A = \begin{bmatrix} 6 & 6{,}917 \\ 1 & 1{,}152 \end{bmatrix}$$

Importanța cunoaşterii condiționării unei matrice decurge din faptul că rezultatele obținute prin calcul numeric corespund inevitabil unei probleme perturbate, ca urmare a efectelor erorilor de aproximare prin rotunjire.

Testarea matricelor din punct de vedere al condiționării se poate face cu ajutorul unor funcții Matlab destinate evaluării *numărului de condiționare*.

Numărul de condiționare al unei matrice indică *sensibilitatea soluției* sistemului de ecuații liniare şi se defineşte ca *raportul dintre cea mai mare şi cea mai mică valoare singulară a acesteia*. O matrice bine condiționată este o matrice relativ insensibilă la mici perturbații şi reciproc o matrice slab condiționată este sensibilă la micile perturbații ale valorilor elementelor sale.

În Matlab există funcții pentru calculul numărului de condiționare al unei matrici, după cum urmează:

`c=cond(X)` – returnează raportul dintre cea mai mare valoare singulară a matricei X şi cea mai mică. O valoare mare a raportului (deci a numărului de condiționare) indică o matrice aproape singulară, adică o slabă condiționare a problemei.

`c=rcond(X)` – reprezintă o funcție mai performantă ce returnează un număr de condiționare care se interpretează astfel:

- dacă c≈eps matricea X este slab condiționată,
- dacă c≈1 matricea X este bine condiționată.

`c=condest(X)` – estimarea celui mai mic număr de condiționare.

Observație. Aceste funcții acceptă ca argument doar matrici pătrate (corespunzătoare sistemelor de ecuații determinate).

Revenind la exemplul dat, evaluarea numărului de condiționare al matricei A se calculează cu funcția:

```
c=rcond(A),
```

care returnează `c=4.7972e-005`, valoare ce confirmă o slabă condiționare a problemei.

Celelalte funcții pentru evaluarea condiționării problemei returnează:

```
cond(A)= 1.7234e+004
condest(A)= 2.0845e+004
```

Exercițiu propus. Să se verifice condiționarea următorului sistem liniar:

$$A = \begin{pmatrix} 1{,}2969 & 0{,}8648 \\ 0{,}2161 & 0{,}1441 \end{pmatrix},\ b = \begin{pmatrix} 0{,}8642 \\ 0{,}1440 \end{pmatrix}.$$

5.2.5 Descompunerea şi factorizarea matricelor

În cadrul acestui capitol vom trece în revistă funcțiile Matlab specifice pentru operațiile de descompunere şi factorizare a matricelor. Acestea sunt operații care se folosesc preponderent în analiza sistemelor lineare şi ca metode alternative în rezolvarea sistemelor de ecuații liniare.

a) Valori şi vectori proprii

Acestea sunt noțiuni matematice ce intervin în problemele de găsire a soluțiilor nebanale pentru sistemele de ecuații de forma:

$$F \cdot x = 0,$$

unde F este o funcție de matrice pătratică şi reală, de ordinul n, conținând un parametru λ.

În funcție de forma lui F, problemele de valori proprii pot fi de două tipuri, astfel:

- dacă $F = A - \lambda \cdot I$, avem o problemă de valori proprii *liniară particulară*;
- dacă $F = A - \lambda \cdot B$, avem o problemă de valori proprii *liniară generală*, în care matricele A şi B sunt pătrate, (I fiind matricea identitate).

În primul caz avem situația sistemului liniar $(A - \lambda \cdot I) \cdot x = 0$ adică ecuația matriceală $A \cdot x = \lambda \cdot x$, unde:

A – matrice pătratică
x – vector propriu
λ – scalari, numiți valori proprii.

În cazul al doilea sistemul liniar este $(A-\lambda\cdot B)\cdot x=0$ adică $A\cdot x=\lambda\cdot B\cdot x$, unde:

A, B – matrice pătrate
x – vector propriu generalizat
λ – valori proprii generalizate

Funcția Matlab care returnează aceste valori este `eig` şi este apelabilă cu una din sintaxele:

```
V=eig(A)
```

unde V – vector ce conține valorile proprii ale matricei A;

```
V=eig(A,B)
```

unde V – vector ce conține valorile proprii generalizate;

```
[V,D]=eig(A)
```

unde V – matrice ale cărei coloane sunt vectorii proprii corespunzători valorilor proprii astfel încât A·V=V·D,
D – matrice diagonală care conține valorile proprii ale matricei A;

```
[V,D]=eig(A,B)
```

respectiv

```
[V,D]=eig(A,'nobalance'),
```

în care ‘`nobalance`’ este o opțiune ce anulează *scalarea* valorilor elementelor mici ale matricei A (comparabile cu erorile de rotunjire) şi considerarea lor la fel de semnificativă ca şi celelalte elemente – lucru care ar conduce la vectori proprii incorecți.

În continuare se dau câteva exemple numerice pentru lucrul cu valori şi vectori proprii.

```
>> A=[6 6.917;1 1.152]

A =
    6.0000    6.9170
    1.0000    1.1520


>>V=eig(A)

V =
    7.1527
   -0.0007


>> [V,D]=eig(A)

V =
    0.9864   -0.7554
    0.1644    0.6553

D =
    7.1527         0
         0   -0.0007


>>[V,D]=eig(A,'nobalance')

V =
    1.0000   -1.0000
    0.1666    0.8675

D =
    7.1527         0
         0   -0.0007
```

```
>> B=randn(2)

B =
   -0.4326    0.1253
   -1.6656    0.2877


>> V1=eig(A,B)

V1 =
   -0.0004
  149.7197


>> [V,D]=eig(A,B)

V =
    1.0000   -0.1674
   -0.8674   -1.0000

D =
   -0.0004         0
         0         149.7197
```

b) Factorizarea Cholesky

Factorizarea unei matrice A constă în descompunerea ei de forma $R^t \cdot R = A$ unde R, numit *factorul Cholesky* este o matrice unică, superior triunghiulară şi nesingulară.

Factorizarea Cholesky este o metodă directă de rezolvare a unui sistem de ecuații liniare de forma A·x=b, cu condiția ca matricea A să fie pozitiv definită, adică: $A=A^t$ şi `det(A)>0`. Prin urmare, dacă se factorizează în sens Cholesky matricea coeficienților sistemului A, ecuația matriceală ce descrie sistemul devine:

$$R^t \cdot R \cdot x = b\ ,$$

iar soluția acesteia se obține prin rezolvarea succesivă a sistemelor: $R^t \cdot y = b$ şi apoi $R \cdot x = y$.

Funcția Matlab pentru factorizarea Cholesky se apelează cu sintaxele:

```
R=chol(A);
[R,p]=chol(A);
```

unde A – o matrice (pozitiv definită)

R – o matrice superior triunghiulară, astfel încât `R'*R=A`

p – un scalar de test, egal cu zero dacă matricea A este pozitiv definită şi un întreg pozitiv în caz contrar.

Exemplu. Rezolvarea unui sistem de ecuații liniare în Matlab pe baza factorizării Cholesky cu forma `R'*R*x=b` se va efectua parcurgând următoarele etape:

- se rezolvă sistemul $R^t \cdot y=b$ cu instrucțiunea `y=R'\b`:
- se rezolvă sistemul $R \cdot x=y$ cu instrucțiunea `x=R\y`.

Aceste etape se rezolvă cu instrucțiunile:

```
R=chol(A),
X=R\(R'\b)
```

```
>> A=[10 -1 1; -1 5 0;1 0 10]
A =
    10    -1     1
    -1     5     0
     1     0    10


>> b'=[10 -5 0]
b =
    10
    -5
     0


>>  [R,p]=chol(A)
R =
    3.1623   -0.3162    0.3162
     0        2.2136    0.0452
     0         0        3.1461

p =
     0


>> x=R\(R'\b)
x =
       0.9278
      -0.8144
      -0.0928
```

Verificare

```
>> b=A*x

b =

    10
    -5
     0
```

c) Factorizarea LU (lower-upper)

Acest tip de factorizare realizează o descompunere a unei matrici pătrate ca produs a două matrice triunghiulare: una inferior triunghiulară, cu elemente 1 pe diagonala principală (matricea L) şi cealaltă superior triunghiulară (matricea U).

Sintaxe de apelare:

`[L,U]=lu(X),`

returnează o matrice superior triunghiulară U şi o matrice inferior triunghiulară permutată L, astfel încât X=L*U;

`[L,U,P]=lu(X),`

furnizează o matrice superior triunghiulară U, o matrice inferior triunghiulară L şi permutarea matriceală P, astfel încât `L*U=P*X`.

<u>*Notă*</u>. În Matlab factorizarea LU este utilizată pentru calculul inversei şi determinantului în cadrul algoritmului funcţiei `inv`, respectiv `det`. De asemenea, acest tip de factorizare constituie baza rezolvării sistemelor de ecuaţii liniare prin împărţire matriceală cu operatorii \ şi /, astfel:

- inversa unei matrice se poate calcula cu: `X=inv(U)*inv(L);`
- determinantul se calculează cu: `D=det(L)*det(U);`
- rezolvarea sistemelor de ecuaţii liniare A·x=b prin factorizarea LU presupune soluţionarea succesivă a sistemelor: `y=L\b` şi `x=U\y` cu instrucţiunile: `[L,U]=lu(A)` şi `x=U\(L\b).`

d) Factorizarea QR

Factorizarea QR este o descompunere a unei matrice ca produs între o matrice ortonormală Q (adică $Q^t \cdot Q=I$) şi o matrice superior triunghiulară R. Se mai numeşte *descompunerea ortogonal-triunghiulară* şi se realizează cu funcţia `qr` apelată cu una dintre sintaxele:

```
[Q,R]=qr(X),
[Q,R,E]=qr(X),
```

unde: R – matrice superior triunghiulară de aceeaşi dimensiune ca X;
Q – matrice ortogonală;
E – matricea permutată a matricei superior triunghiulară R cu elementele diagonalei descrescătoare astfel încât: `X*E=Q*R`, iar `triu(qr(X))=R`.

Notă. Aceast tip de factorizare se foloseşte pentru rezolvarea sistemelor de ecuaţii liniare supradeterminate (care au mai multe ecuaţii decât necunoscute). Prin utilizarea factorizării QR soluţia sistemului este calculată în doi paşi:

```
y=Q'*b
x=R\y
```

Exemplu. Fie sistemul supradeterminat descris de matricea A= [10 2 1 ; -1 5 0 ;1 0 10 ; -2 3 7] şi vectorul termenilor liberi b=[10 5 0 1]. Să se determine o soluţie a sistemului folosind factorizarea QR potrivit secvenţei de instrucţiuni:

```
[Q,R]=qr(A)
x=R\(Q'*b)
```

```
>> A=[10 2 1 ; -1 5 0 ;1 0 10 ; -2 3 7]

A =
    10     2     1
    -1     5     0
     1     0    10
    -2     3     7


>> b=[10 5 0 1]

>> b=b'

b =
    10
     5
     0
     1

>> [Q,R]=qr(A)

Q =
   -0.9713   -0.1886    0.0224    0.1433
    0.0971   -0.8333    0.2584   -0.4789
   -0.0971    0.0139   -0.8568   -0.5062
    0.1943   -0.5195   -0.4457    0.7027

R =

  -10.2956   -0.8742   -0.5828
         0   -6.1021   -3.6857
         0         0  -11.6652
         0         0         0


>> x=R\(Q'*b)

x =
   0.7864
   1.1324
  -0.0918
```

Verificare

```
>> b=A*x

b =

  10.0371
   4.8758
  -0.1312
   1.1822
```

e) Pseudoinversa unei matrice sau inversa generalizată Moore-Penrose

Pseudoinversa unei matrice se utilizează, de regulă, pentru rezolvarea sistemelor de ecuații liniare supradeterminate, în sensul *celor mai mici pătrate*. Aceasta se calculează cu funcția `pinv` cu sintaxa:

`X=pinv(A),`

returnând matricea pseudoinversă X care are aceeaşi dimensiune ca şi matricea A^t. Matricea pseudoinversă X a unei matrice A verifică relațiile: `A*X*A=A` şi `X*A*X=X`.

Exemplu. Fie $A \cdot x=B$ un sistem supradeterminat cu $A(4\times3)$ şi $B(4\times1)$. Să se afle o soluție cu ajutorul pseudoinversei.

```
>> A=[10 2 1 ; -1 5 0 ;1 0 10 ; -2 3 7]

A =
    10     2     1
    -1     5     0
     1     0    10
    -2     3     7


>> b=[10 5 0 1]

b =
    10     5     0     1

>> b=b'

b =
    10
     5
     0
     1

>> X=pinv(A)

X =
    0.0917   -0.0209    0.0092   -0.0263
    0.0321    0.1499   -0.0466    0.0621
   -0.0019   -0.0222    0.0734    0.0382

>> x=X*b'

x =
    0.7864
    1.1324
   -0.0918
```