Laborator 2

Rezolvarea ecuaţiilor matriceale liniare Sylvester şi Liapunov

2.1 Tema

Însuşirea tehnicilor de rezolvare a unor sisteme liniare, în general de mari dimensiuni, structurate în exprimări matriceale care permit dezvoltarea unor metode specifice de calcul. Concret, se vor elabora, edita şi testa programe MATLAB pentru rezolvarea eficientă a ecuaţiilor matriceale Sylvester şi Liapunov continue şi discrete.

2.2 Ecuaţii matriceale liniare. Algoritmi

Ecuaţiile matriceale liniare sunt sisteme de ecuaţii liniare care pot fi scrise compact în forma matriceală $\sum_{i=1}^{k} A_i X B_i = C$, unde $A_i \in \mathbb{R}^{p\times m}$, $B_i \in \mathbb{R}^{n\times q}$ şi $C \in \mathbb{R}^{p\times q}$ sunt matrice date iar $X \in \mathbb{R}^{m\times n}$ este matricea necunoscutelor. Evident, această ecuaţie poate fi scrisă într-o formă "desfăşurată" ca sistem de pq ecuaţii liniare cu mn necunoscute. O astfel de abordare eludează "structura matriceală" a sistemului iar aplicarea tehnicilor clasice de rezolvare pe sistemul desfăşurat, neexploatând structura internă a datelor de intrare, este, de cele mai multe ori, ineficientă. În continuare vom considera numai cazul "determinat", în care numărul ecuaţiilor este egal cu numărul necunoscutelor, i.e. $A_i \in \mathbb{R}^{m\times m}$, $B_i \in \mathbb{R}^{n\times n}$ sunt matrice pătrate iar $C, X \in \mathbb{R}^{m\times n}$.

Ecuaţiile matriceale liniare cele mai întâlnite în aplicaţii, e.g. în domeniul automaticii, se obţin pentru $k = 2$, i.e. sunt de forma $A_1XB_1 + A_2XB_2 = C$. Particularizările

$$AX + XB = C, \tag{2.1}$$

$$AXB + X = C. \tag{2.2}$$

sunt cunoscute sub denumirea de *ecuaţii matriceale Sylvester* continuă, respectiv discretă.

Particularizând şi mai mult, obţinem ecuaţiile matriceale liniare cunoscute sub denumirile de *ecuaţie Liapunov continuă* pentru

$$A^T X + XA = C, \tag{2.3}$$

respectiv *ecuaţie Liapunov discretă* pentru

$$A^T XA - X = C. \tag{2.4}$$

Condiţiile de existenţă şi unicitate ale soluţiilor acestor ecuaţii sunt date de

Teorema 2.1 *Ecuaţia matriceală Sylvester continuă (2.1) admite o soluţie $X \in \mathbb{R}^{m\times n}$ unică dacă şi numai dacă*

$$\lambda_i + \mu_j \neq 0, \quad \forall\, \lambda_i \in \lambda(A),\ \forall\, \mu_j \in \lambda(B) \tag{2.5}$$

sau, altfel spus, dacă şi numai dacă $\lambda(A) \cap \lambda(-B) = \emptyset$.

Ecuaţia matriceală Sylvester discretă (2.2) are o soluţie $X \in \mathbb{R}^{m\times n}$ unică dacă şi numai dacă

$$\lambda_i\mu_j \neq -1, \quad \lambda_i \in \lambda(A),\ \forall \mu_j\ \in \lambda(B). \tag{2.6}$$

În particular avem

Corolar 2.1 *Ecuaţia matriceală Liapunov continuă (2.3) admite o soluţie unică dacă şi numai dacă matricea A nu are nici o pereche de valori proprii opuse, i.e.*

$$\lambda_i + \lambda_j \neq 0, \quad \forall\, \lambda_i, \lambda_j \in \lambda(A), \tag{2.7}$$

În particular, $0 \notin \lambda(A)$ i.e. matricea A trebuie să fie nesingulară.

Ecuaţia matriceală Liapunov discretă (2.4) admite o soluţie unică dacă şi numai dacă matricea A nu are nici o pereche de valori proprii inverse, i.e.

$$\lambda_i\lambda_j \neq 1, \quad \forall\, \lambda_i, \lambda_j \in \lambda(A), \tag{2.8}$$

În particular, este necesar ca $\pm 1 \notin \lambda(A)$.

2.2.1 Rezolvarea ecuaţiilor matriceale Sylvester

Rezolvarea ecuaţiilor matriceale Sylvester triunghiulare şi cvasitriunghiulare

Ecuaţiile Sylvester sau Liapunov *triunghiulare* sunt cele la care matricele A şi B au o structură triunghiulară şi permit un calcul simplu al necunoscutelor printr-o procedură de substituţie numerică directă dacă se respectă o ordine de calcul determinată. Pentru subliniere, vom folosi notaţii specifice. Fie, de exemplu, ecuaţia Sylvester continuă

$$SX + XT = C \tag{2.9}$$

unde matricele $S \in \mathbb{R}^{m\times m}$ şi $T \in \mathbb{R}^{n\times n}$ sunt *superior triunghiulare* cu $s_{ii} + t_{jj} \neq 0$, $\forall i = 1 : m$, $\forall j = 1 : n$. Din egalitatea $(SX + XT)_{ij} = c_{ij}$ rezultă

$$s_{ii}x_{ij} + x_{ij}t_{jj} = c_{ij} - \sum_{k=i+1}^{m} s_{ik}x_{kj} - \sum_{k=1}^{j-1} x_{ik}t_{kj}, \quad i = 1 : m,\ j = 1 : n \tag{2.10}$$

Având în vedere că în membrul drept al relaţiei (2.10) apar elementele matricei necunoscute X aflate în "stânga" şi "sub" necunoscuta curentă x_{ij} (vezi fig. 2.2.1), rezultă că valorile lor vor fi disponibile la momentul curent dacă ordinea de calcul este $i = m, m-1, \ldots, 1$ şi $j = 1, 2, \ldots, n$, în care caz putem utiliza relaţia de calcul

$$x_{ij} = \frac{c_{ij} - \sum_{k=i+1}^{m} s_{ik}x_{kj} - \sum_{k=1}^{j-1} x_{ik}t_{kj}}{s_{ii} + t_{jj}}. \tag{2.11}$$

Rezultă următoarele două versiuni (pe "linii', respectiv "pe coloane") ale algoritmului de rezolvare a ecuaţiiei matriceale Sylvester (2.9).

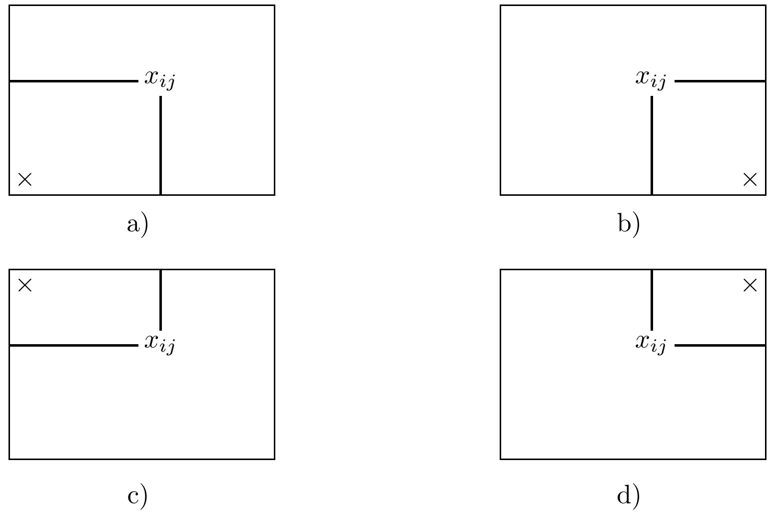

Figura 2.1: Dependenţele necunoscutei curente x_{ij} de alte elemente ale matricei X la ecuaţiile Sylvester triunghiulare continue $SX + XT = C$ în următoarele situaţii: a) S şi T superior triunghiulare; b) S superior triunghiulară şi T inferior triunghiulară; c) S inferior triunghiulară şi T superior triunghiulară; d) S şi T inferior triunghiulare. Cu $\times$ s-a marcat elementul care trebuie calculat primul.

Algoritmul 2.1 (Date matricele superior triunghiulare $S \in \mathbb{R}^{m\times m}$, $T \in \mathbb{R}^{n\times n}$, cu $s_{ii} + t_{jj} \neq 0,\ \ \forall\, i, j$, precum şi matricea $C \in \mathbb{R}^{m\times n}$, algoritmul calculează soluţia $X \in \mathbb{R}^{m\times n}$ a ecuaţiei Sylvester continue $SX + XT = C$.)

% I. Versiunea pe coloane

1. `Pentru` $j = 1 : n$
   1. `Pentru` $i = m : -1 : 1$
      1. $x_{ij} = \dfrac{c_{ij} - \sum_{k=i+1}^{m} s_{ik}x_{kj} - \sum_{k=1}^{j-1} x_{ik}t_{kj}}{s_{ii} + t_{jj}}$.

% II. Versiunea pe linii

1. `Pentru` $i = m : -1 : n$
   1. `Pentru` $j = 1 : n$
      1. $x_{ij} = \dfrac{c_{ij} - \sum_{k=i+1}^{m} s_{ik}x_{kj} - \sum_{k=1}^{j-1} x_{ik}t_{kj}}{s_{ii} + t_{jj}}$.

În cazul ecuaţiei Sylvester discrete triunghiulare

$$SXT + X = C \tag{2.12}$$

cu matricele $S \in \mathbb{R}^{m\times m}$ şi $T \in \mathbb{R}^{n\times n}$, de exemplu, *superior triunghiulare* cu $s_{ii}t_{jj} \neq -1$, $\forall i = 1 : m$, $\forall j = 1 : n$, din egalitatea $(SXT + X)_{ij} = c_{ij}$ rezultă

$$s_{ii}x_{ij}t_{jj} + x_{ij} = c_{ij} - \sum_{l=1}^{j}\sum_{k=\kappa}^{m} s_{ik}x_{kl}t_{lj}, \quad i = 1 : m,\ j = 1 : n \tag{2.13}$$

unde

$$\kappa = \begin{cases} i & \text{pt.} \quad l < j \\ i+1 & \text{pt.} \quad l = j \end{cases} \tag{2.14}$$

şi de unde putem exprima, dacă sunt satisfăcute condiţiile de existenţă ale soluţiei, necunoscuta x_{ij} în raport cu celelalte necunoscute

$$x_{ij} = \frac{c_{ij} - \sum_{l=1}^{j} \sum_{k=\kappa}^{m} s_{ik} x_{kl} t_{lj}}{s_{ii} t_{jj} + 1}. \tag{2.15}$$

Dependenţa necunoscutei ”curente” de celelalte necunoscute este prezentată în fig. 2.2.1 a). Se observă uşor că, respectând aceeaşi ordine de calcul ca în cazul continuu, toate necunoscutele pot fi calculate printr-o substituţie numerică directă. Prezentăm varianta ”pe coloane” a algoritmului de calcul.

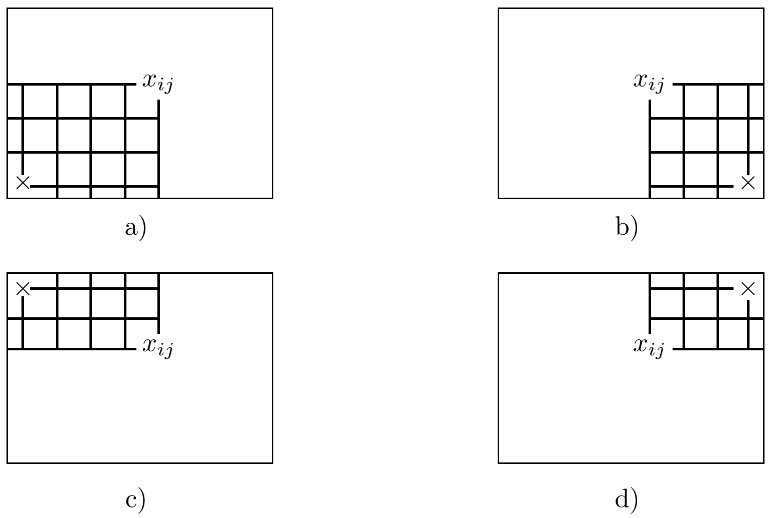

Figura 2.2: Dependenţele necunoscutei curente x_{ij} de alte elemente ale matricei X la ecuaţiile Sylvester triunghiulare discrete $SXT + X = C$ în următoarele situaţii: a) S şi T superior triunghiulare; b) S superior triunghiulară şi T inferior triunghiulară; c) S inferior triunghiulară şi T superior triunghiulară; d) S şi T inferior triunghiulare. Cu $\times$ s-a marcat elementul care trebuie calculat primul.

Algoritmul 2.2 (Date matricele superior triunghiulare $S \in \mathbb{R}^{m\times m}$, $T \in \mathbb{R}^{n\times n}$, cu $s_{ii}t_{jj} \neq -1$, $\forall\, i, j$, precum şi matricea $C \in \mathbb{R}^{m\times n}$, algoritmul calculează soluţia $X \in \mathbb{R}^{m\times n}$ a ecuaţiei Sylvester discrete $SXT + X = C$.)

1. `Pentru` $j = 1 : n$
   1. `Pentru` $i = m : -1 : 1$
      1. $x_{ij} = \dfrac{c_{ij} - \sum_{l=1}^{j} \sum_{k=\kappa}^{m} s_{ik} x_{kl} t_{lj}}{s_{ii} t_{jj} + 1}.$

În cazul altor altor structuri triunghiulare ordinea impusă de calcul este aceeaşi în cazurile continuu şi discret şi poate fi dedusă simplu din diagramele din fig.2.2.1. Pentru necesităţile ulterioare, introducem următoarele sintaxe

$$X = \textbf{sylv\_st\_c}(S, T, C), \qquad X = \textbf{sylv\_st\_d}(S, T, C)$$

pentru una din versiunile algoritmului 2.1, respectiv algoritmului 2.2.

În situaţia unor matrici de coeficienţi $S = A$ şi $T = B$ bloc-triunghiulare, fie superior bloc-triunghiulare (de exemplu forme Schur a unor matrice)

$$S = \begin{bmatrix} S_{11} & S_{12} & \cdots & S_{1p} \\ 0 & S_{22} & \cdots & S_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & S_{pp} \end{bmatrix}, \qquad T = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1p} \\ 0 & T_{22} & \cdots & T_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & T_{pp} \end{bmatrix}, \tag{2.16}$$

care impun partiţii bloc corespunzătoare ale matricelor C şi X

$$C = \begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1q} \\ C_{21} & C_{22} & \cdots & C_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ C_{p1} & C_{p2} & \cdots & C_{pq} \end{bmatrix}, \qquad X = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1q} \\ X_{21} & X_{22} & \cdots & X_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ X_{p1} & X_{p2} & \cdots & X_{pq} \end{bmatrix}, \tag{2.17}$$

unde $S_{ij} \in \mathbb{R}^{m_i \times m_j}$, $T_{ij} \in \mathbb{R}^{n_i \times n_j}$, $C_{ij}, X_{ij} \in \mathbb{R}^{m_i \times n_j}$. Evident, dacă S, T sunt forme Schur reale atunci $m_k, n_k \in \{1, 2\}$. În acest caz, relaţiile (2.10), respectiv (2.13), rămân valabile la nivel de bloc

$$S_{ii}X_{ij} + X_{ij}T_{jj} = C_{ij} - \sum_{k=i+1}^{p} S_{ik}X_{kj} - \sum_{k=1}^{j-1} X_{ik}T_{kj}, \quad i = 1:p,\ j = 1:q \tag{2.18}$$

pentru cazul continuu, respectiv

$$S_{ii}X_{ij}T_{jj} + X_{ij} = C_{ij} - \sum_{l=1}^{j} \sum_{k=\kappa}^{p} S_{ik}X_{kl}T_{lj}, \quad i = 1:p,\ j = 1:q \tag{2.19}$$

pentru cazul discret. În ipoteza că adoptăm o ordine de calcul care asigură posibilitatea determinării membrului drept, ecuaţiile (2.18) şi (2.19) sunt ecuaţii Sylvester continue, respectiv discrete, *netriunghiulare*, care, pentru m_k, n_k mici, pot fi rezolvate prin desfăşurarea lor [1] sub forma unor sisteme de $m_i n_j$ ecuaţii liniare şi tot atâtea necunoscute şi utilizarea unor metode standard cum ar fi eliminarea gaussiană.

Scrierea algoritmilor pentru rezolvarea ecuaţiilor Sylvester cvasitriunghiulare este transparentă şi este lăsată în sarcina studentului. Ne mărginim aici să fixăm sintaxele de utilizare

$$X = \mathbf{sylv\_sct\_c}(S, T, C), \qquad X = \mathbf{sylv\_sct\_d}(S, T, C)$$

pentru rezolvarea ecuaţiilor *superior* cvasitriunghiulare continue, respectiv discrete.

Rezolvarea ecuaţiilor matriceale Sylvester generale

Modalităţile eficiente de rezolvare a ecuaţiilor matriceale Sylvester constau în transformări care reduc problema la rezolvarea unor ecuaţii Sylvester triunghiulare sau cvasitriunghiulare. Pentru aceasta se utilizează forma Schur complexă sau reală a matricelor A şi B. În acest

[1] Dacă $\bar{x} \in \mathbb{R}^{mn}$ şi $\bar{c} \in \mathbb{R}^{mn}$ sunt vectorii definiţi, de exemplu, prin concatenarea coloanelor matricelor X respectiv C ecuaţia $AX + XB = C$ se poate scrie în forma $(I_n \otimes A + B^T \otimes I_m)\bar{x} = \bar{c}$ iar ecuaţia $AXB + X = C$ în forma $(B^T \otimes A + I_{mn})\bar{x} = \bar{c}$. În relaţiile de mai sus $\otimes$ semnifică produsul Kronecker definit în felul următor: dacă $U \in \mathbb{R}^{p\times q}$, $V \in \mathbb{R}^{r\times s}$ atunci $W \stackrel{\text{def}}{=} U \otimes V \in \mathbb{R}^{pr\times qs}$ este matricea având structura bloc $W = [\, W_{ij}\,]_{i=1:p,\, j=1:q}$ cu $W_{ij} = u_{ij}V$.

context amintim că oricare ar fi matricele $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{C}^{n\times n}$, există matricele unitare $U \in \mathbb{C}^{m\times m}$ şi $V \in \mathbb{C}^{n\times n}$ astfel încât

$$U^H AU = S \in \mathbb{C}^{m\times m}, \qquad V^H BV = T \in \mathbb{C}^{n\times n} \tag{2.20}$$

sunt matrice superior triunghiulare numite *forme Schur complexe* ale matricelor A respectiv B. De asemenea, există matricele ortogonale $U \in \mathbb{R}^{m\times m}$ şi $V \in \mathbb{R}^{n\times n}$ astfel încât

$$U^T AU = S \in \mathbb{R}^{m\times m}, \qquad V^T BV = T \in \mathbb{R}^{n\times n} \tag{2.21}$$

sunt matrice superior cvasitriunghiulare numite *forme Schur reale* ale matricelor A respectiv B. În MATLAB forma Schur reală precum şi matricea de transformare se obţin cu funcţia **schur**:

$$[U, S] = \mathbf{schur}(A)$$

iar forma Schur complexă, printr-un apel suplimentar al funcţiei **rsf2csf**

$$[U, S] = \mathbf{rsf2csf}(U, S).$$

Înlocuind în ecuaţia matriceală Sylvester continuă $AX + XB = C$ expresiile $A = USU^H$ şi $B = VTV^H$ aceasta devine $USU^H X + XVTV^H = C$, de unde avem $SU^H XV + U^H XVT = U^H CV$. Notând cu $Y = U^H XV$, $\tilde{C} = U^H CV$ obţinem o ecuaţie Sylvester triunghiulară $SY + YT = \tilde{C}$.

Redactarea algoritmului de calcul este următoarea.

Algoritmul 2.3 (Date matricele $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{n\times n}$, $C \in \mathbb{R}^{m\times n}$ cu $\lambda(A) \cap \lambda(-B) = \emptyset$, algoritmul calculează soluţia $X \in \mathbb{R}^{m\times n}$ a ecuaţiei Sylvester continue $AX + XB = C$ prin aducerea matricelor A şi B la forma Schur complexă.)

1. $[\,U,\, S\,] = \mathbf{schur}(A)$
2. $[\,U,\, S\,] = \mathbf{rsf2csf}(U, S)$
3. $[\,V,\, T\,] = \mathbf{schur}(B)$
4. $[\,V,\, T\,] = \mathbf{rsf2csf}(V, T)$
5. $C \leftarrow \tilde{C} = U^H CV$
6. $X = \mathbf{sylv\_st\_c}(S, T, C)$
7. $X = \mathbf{real}(UXV^H)$

Modalităţile de calcul ale soluţiei ecuaţiei matriceale Sylvester discrete $AXB + X = C$ sunt absolut similare şi, în consecinţă, ne mărginim să prezentăm direct algoritmul de calcul.

Algoritmul 2.4 (Date matricele $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{n\times n}$ astfel încât $\lambda_i \mu_j \neq -1$ pentru toţi $\lambda_i \in \lambda(A)$ şi $\mu_j \in \lambda(B)$ şi matricea $C \in \mathbb{R}^{m\times n}$ algoritmul calculează soluţia X a ecuaţiei Sylvester discrete $AXB + X = C$ prin reducerea matricelor A şi B la forma Schur complexă.)

1. $[\,U,\, S\,] = \mathbf{schur}(A)$
2. $[\,U,\, S\,] = \mathbf{rsf2csf}(U, S)$
3. $[\,V,\, T\,] = \mathbf{schur}(B)$
4. $[\,V,\, T\,] = \mathbf{rsf2csf}(V, T)$
5. $C \leftarrow \tilde{C} = U^H CV$
6. $X = \mathbf{sylv\_st\_d}(S, T, C)$

7. $X = \mathbf{real}(UXV^H)$

Ultima instrucţiune din ultimii doi algoritmi este necesară întrucât calculele intermediare cu numere complexe conduc la apariţia unor reziduuri imaginare care pot şi trebuie să fie neglijate.

O metodă mai performantă se obţine dacă se utilizează în exclusivitate o aritmetică reală. În acest caz, utilizarea formelor Schur reale $S = U^T AU$ şi $T = V^T BV$ conduce la necesitatea rezolvării unor ecuaţii Sylvester cvasisuperior triunghiulare conform următorilor doi algoritmi.

Algoritmul 2.5 (Algoritmul **Bartels-Stewart**. Date matricele $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{n\times n}$, $C \in \mathbb{R}^{m\times n}$ cu $\lambda(A) \cap \lambda(-B) = \emptyset$, algoritmul calculează soluţia $X \in \mathbb{R}^{m\times n}$ a ecuaţiei Sylvester continue $AX + XB = C$ prin aducerea matricelor A şi B la forma Schur reală.)

1. $[U, S] = \mathbf{schur}(A)$
2. $[V, T] = \mathbf{schur}(B)$
3. $C \leftarrow \tilde{C} = U^T CV$
4. $X = \mathbf{sylv\_sct\_c}(S, T, C)$
5. $X = UXV^T$.

Algoritmul 2.6 (Algoritmul **Kitagawa-Barraud**. Date matricele $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{n\times n}$ astfel încât $\lambda_i\mu_j \neq -1$ pentru toţi $\lambda_i \in \lambda(A)$ şi $\mu_j \in \lambda(B)$ şi matricea $C \in \mathbb{R}^{m\times n}$ algoritmul calculează soluţia X a ecuaţiei Sylvester discrete $AXB+X = C$ prin reducerea matricelor A şi B la forma Schur reală.)

1. $[U, S] = \mathbf{schur}(A)$
2. $[V, T] = \mathbf{schur}(B)$
3. $C \leftarrow \tilde{C} = U^T CV$
4. $X = \mathbf{sylv\_sct\_d}(S, T, C)$
5. $X = UXV^T$.

Cea mai eficientă metodă de calcul ”de transformare” este considerată aşa numita metodă ”Hessenberg-Schur”, în care matricea A este redusă la forma Hessenberg iar matricea B la forma Schur complexă. În acest caz rezolvarea ecuaţiilor Sylvester se reduce la rezolvarea, de exemplu prin eliminare gaussiană, a n sisteme superior Hessenberg de $m$ecuaţii cu m necunoscute.

2.2.2 Rezolvarea ecuaţiilor matriceale Liapunov

Întrucât ecuaţiile Liapunov continuă $A^T X + XA = C$ şi discretă $A^T XA - X = C$ sunt cazuri particulare ale ecuaţiilor Sylvester corespunzătoare şi, în consecinţă, algoritmii din paragraful precedent pot fi utilizaţi pentru rezolvarea lor. Apar totuşi unele particularităţi şi simplificări care sunt prezentate mai în continuare.

Rezolvarea ecuaţiilor matriceale Liapunov triunghiulare şi cvasitriunghiulare

Dacă matricea $T \in \mathbb{R}^{m\times m}$ este superior triunghiulară atunci matricea $S = T^T$ este inferior triunghiulară şi ecuaţia $T^T X + XT = C$ scrisă pe componente $(T^T X + XT)_{ij} = c_{ij}$ conduce la

$$t_{ii}x_{ij} + x_{ij}t_{jj} = c_{ij} - \sum_{k=1}^{i-1} t_{ki}x_{kj} - \sum_{k=1}^{j-1} x_{ik}t_{kj}, \quad i = 1 : m,\ j = 1 : n \tag{2.22}$$

de unde, în condiţiile corolarului 2.1, avem

$$x_{ij} = \frac{c_{ij} - \sum_{k=1}^{i-1} t_{ki}x_{kj} - \sum_{k=1}^{j-1} x_{ik}t_{kj}}{t_{ii} + t_{jj}}. \tag{2.23}$$

care pot fi calculate dacă se respectă ordinea $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, m$ (vezi fig.2.2.1 c)). Rezultă algoritmul

Algoritmul 2.7 (Date matricea $T \in \mathbb{R}^{m\times m}$, cu $t_{ii} + t_{jj} \neq 0,\ \ \forall\, i, j$, precum şi matricea $C \in \mathbb{R}^{m\times m}$, algoritmul calculează soluţia $X \in \mathbb{R}^{m\times m}$ a ecuaţiei Liapunov continue $T^T X + XT = C$.)

1. `Pentru` $j = 1 : m$
   1. `Pentru` $i = 1 : m$
      1. $x_{ij} = \dfrac{c_{ij} - \sum_{k=1}^{i-1} t_{ki}x_{kj} - \sum_{k=1}^{j-1} x_{ik}t_{kj}}{t_{ii} + t_{jj}}$.

Absolut similar, ecuaţia Liapunov discretă $T^T XT - X = C$, cu matricea $T \in \mathbb{R}^{m\times m}$ superior triunghiulară, scrisă pe componente $(T^T XT - X)_{ij} = c_{ij}$, conduce la

$$x_{ij} = \frac{c_{ij} - \sum_{l=1}^{j} \sum_{k=1}^{\sigma} t_{ki}x_{kl}t_{lj}}{t_{ii}t_{jj} - 1}, \tag{2.24}$$

unde

$$\sigma = \begin{cases} i & \text{pentru} \quad l \neq j \\ i-1 & \text{pentru} \quad l = j \end{cases}. \tag{2.25}$$

Obţinem următorul algoritm.

Algoritmul 2.8 (Date matricea superior triunghiulară $T \in \mathbb{R}^{m\times m}$, cu $t_{ii}t_{jj} \neq 1,\ \ \forall\, i, j$, precum şi matricea $C \in \mathbb{R}^{m\times m}$, algoritmul calculează soluţia $X \in \mathbb{R}^{m\times m}$ a ecuaţiei Liapunov discrete $T^T XT - X = C$.)

1. `Pentru` $j = 1 : m$
   1. `Pentru` $i = 1 : m$
      1. $x_{ij} = \dfrac{c_{ij} - \sum_{l=1}^{j} \sum_{k=1}^{\sigma} t_{ki}x_{kl}t_{lj}}{t_{ii}t_{jj} - 1}$.

Introducem următoarele sintaxe de utilizare pentru cei doi algoritmi de mai sus

$$X = \textbf{liap\_st\_c}(A, C), \quad \textit{respectiv} \quad X = \textbf{liap\_st\_d}(A, C).$$

Un caz important este cel în care matricea C este simetrică ($C = C^T$), situaţie în care şi matricea X este simetrică. Prin urmare, se calculează numai triunghiul inferior (sau cel superior) iar algoritmii de mai sus devin.

Algoritmul 2.9 (Date matricea $T \in \mathbb{R}^{m\times m}$, cu $t_{ii} + t_{jj} \neq 0,\ \ \forall\, i, j$, precum şi matricea *simetrică* $C \in \mathbb{R}^{m\times m}$, algoritmul calculează triunghiul inferior al soluţiei $X \in \mathbb{R}^{m\times m}$ a ecuaţiei Liapunov continue $T^T X + XT = C$.)

1. `Pentru` $j = 1 : m$
   1. `Pentru` $i = j : m$
      1. $x_{ij} = \dfrac{c_{ij} - \sum_{k=1}^{i-1} t_{ki}x_{kj} - \sum_{k=1}^{j-1} x_{ik}t_{kj}}{t_{ii} + t_{jj}}$.

Algoritmul 2.10 (Date matricea superior triunghiulară $T \in \mathbb{R}^{m\times m}$, cu $t_{ii}t_{jj} \neq 1, \ \forall\, i, j$, precum şi matricea *simetrică* $C \in \mathbb{R}^{m\times m}$, algoritmul calculează triunghiul inferior al soluţiei $X \in \mathbb{R}^{m\times m}$ a ecuaţiei Liapunov discrete $T^T XT - X = C$.)

1. `Pentru` $j = 1 : m$
   1. `Pentru` $i = j : m$
      1. $x_{ij} = \dfrac{c_{ij} - \sum_{l=1}^{j}\sum_{k=1}^{\sigma} t_{ki}x_{kl}t_{lj}}{t_{ii}t_{jj} - 1}$.

Introducem următoarele sintaxe de utilizare pentru cei doi algoritmi de mai sus

$$X = \textbf{liap\_st\_c\_sim}(A, C), \quad \textit{respectiv} \quad X = \textbf{liap\_st\_d\_sim}(A, C).$$

Cazul ecuaţiilor Liapunov cvasitriunghiulare este propus ca exerciţiu.

Rezolvarea ecuaţiilor matriceale Liapunov generale

Considerăm mai întâi cazul *ecuaţiei Liapunov continue* $A^T X + XA = C$ şi fie reducerea $S = U^H AU$ a matricei $A \in \mathbb{R}^{n\times n}$ la *forma Schur complexă*. Rezultă $A^T = A^H = US^H U^H$, cu care ecuaţia Liapunov devine

$$US^H U^H X + XUSU^H = C \tag{2.26}$$

respectiv

$$S^H Y + YS = \tilde{C} \tag{2.27}$$

unde

$$Y = U^H XU, \quad \tilde{C} = U^H CU. \tag{2.28}$$

În condiţiile corolarului 2.1, aceste sisteme sunt nesingulare şi pot fi rezolvate în ordinea $j = 1, 2 ..., n$. Cu observaţia că soluţia X a ecuaţiei iniţiale se calculează apoi din prima relaţie (2.28) rezultă următorul algoritm.

Algoritmul 2.11 (Date matricele $A \in \mathbb{R}^{n\times n}$, satisfăcând $\lambda_i + \lambda_j \neq 0, \forall\lambda_i, \lambda_j \in \lambda(A)$, şi $C \in \mathbb{R}^{n\times n}$ oarecare, algoritmul calculează soluţia X a ecuaţiei Liapunov $A^T X + XA = C$ pe baza reducerii matricei A la forma Schur complexă.)

1. $[U, S] = \textbf{schur}(A)$
2. $[U, S] = \textbf{rsf2csf}(U, S)$
3. $C \leftarrow U^H CU$
4. $X = \textbf{liap\_st\_c\_sim}(A, C)$.
5. $X = \textbf{real}(UXU^H)$.

În cazul în care matricea C este *simetrică* apar simplificări suplimentare datorită faptului că şi matricea soluţie este *simetrică*. Acest fapt poate fi exploatat observând că, datorită simetriei matricelor C şi X, matricele Y şi $\tilde{C}$ din (2.28) sunt hermitice şi în consecinţă este suficient să se calculeze, de exemplu, numai triunghiul superior al matricelor $\tilde{C}, Y$ şi, în final, matricea X.

Algoritmul de rezolvare a *ecuaţiei Liapunov discrete* (2.10), bazat pe reducerea matricei A la *forma Schur complexă*, urmăreşte îndeaproape schema de mai sus.

Algoritmul 2.12 (Date matricele $A \in \mathbb{R}^{m\times m}$, satisfăcând $\lambda_i\lambda_j \neq 1, \forall\lambda_i \in \lambda(A)$ şi $C \in \mathbb{R}^{m\times m}$ oarecare, algoritmul calculează soluţia X a ecuaţiei Liapunov $A^T XA - X = C$ pe baza reducerii matricei A la forma Schur complexă.)

1. $[U, S] = \mathbf{schur}(A)$
2. $[U, S] = \mathbf{rsf2csf}(U,S)$
3. $C \leftarrow U^H C U$
4. $X = \mathbf{liap\_st\_d\_sim}(A,C)$.
5. $X = \mathbf{real}(UXU^H)$.

În încheiere menţionăm faptul că rezolvarea unei ecuaţii Liapunov continue poate fi redusă la rezolvarea unei ecuaţii corespondente discrete şi reciproc utilizând transformarea (omografică)

$$B = (A - I_n)^{-1}(A + I_n), \tag{2.29}$$

ecuaţia $A^T XA - X = C$ devine $B^T X + XB = D$ unde $D = \frac{1}{2}(B^T - I_n)C(B - I_n)$.

Programe MATLAB disponibile

Pentru rezolvarea ecuaţiilor matriceale Sylvester şi Liapunov continue este disponibilă funcţia **lyap**, iar pentru ecuaţia Liapunov discretă se utilizează funcţia **dlyap**.

2.3 Sarcini de lucru

A. În laborator

1. Se vor scrie programele MATLAB pentru implementarea algoritmilor de rezolvare a ecuaţiilor Sylvester şi Liapunov superior triunghiulare şi se vor testa pe exemple numerice semnificative ($m, n > 20$). Se va aprecia acurateţea soluţiilor prin calculul normelor reziduurilor $\|C - SX - XT\|$, respectiv $\|C - SXT + X\|$. De asemenea se vor compara soluţiile calculate cu programe proprii cu cele oferite de funcţiile MATLAB disponibile.

2. Se vor scrie programele MATLAB pentru implementarea algoritmilor "Schur-Schur" de rezolvare a ecuaţiilor Sylvester şi Liapunov generale şi se vor testa pe exemple numerice semnificative ($m, n > 20$). Se va aprecia acurateţea soluţiilor prin calculul normelor reziduurilor $\|C - SX - XT\|$, respectiv $\|C - SXT + X\|$ şi se vor compara soluţiile calculate cu programe proprii cu cele oferite de funcţiile MATLAB disponibile.

3. Se vor analiza sursele funcţiilor MATLAB **lyap** şi **dlyap** şi se vor identifica metodele folosite.

B. Acasă

1. Se vor scrie programele MATLAB pentru implementarea algoritmilor de rezolvare a ecuaţiilor Sylvester şi Liapunov superior cvasitriunghiulare (forme Schur reale) şi se vor testa pe exemple numerice semnificative ($m, n > 20$). Se va aprecia acurateţea soluţiilor prin calculul normelor reziduurilor $\|C - SX - XT\|$, respectiv $\|C - SXT + X\|$. De asemenea se or compara soluţiile calculate cu programe proprii cu cele oferite de funcţiile MATLAB disponibile.

2. Se vor scrie programele MATLAB pentru implementarea algoritmilor "Schur-Schur" *reali*, i.e. algoritmii Bartels-Stewart şi Kitagawa-Barraud, de rezolvare a ecuaţiilor Sylvester şi Liapunov generale şi se vor testa pe exemple numerice semnificative ($m, n > 20$). Se va

aprecia acurateţea soluţiilor prin calculul normelor reziduurilor $\|C - SX - XT\|$, respectiv $\|C - SXT + X\|$ şi se vor compara soluţiile calculate cu aceste programe proprii cu cele oferite de programele ce folosesc formele Schur complexe precum şi de funcţiile MATLAB disponibile.

3. **Exerciţii şi teme de casă**

E 2.1 Fiind date matricele $A \in \mathbb{R}^{m \times m}$, $B \in \mathbb{R}^{n \times n}$ care satisfac condiţiile de existenţă si unicitate a soluţiei ecuaţiei implicate, şi $C \in \mathbb{R}^{m \times n}$, să se scrie algoritmii Hessenberg – Schur de rezolvare a ecuaţiei matriceale Sylvester continue $AX + XB = C$, respectiv a celei discrete $AXB + X = C$.

E 2.2 Se poate demonstra că dacă matricea A satisface condiţia $\mathbf{real}\lambda_i < 0$, $\forall \lambda_i \in \lambda(A)$ şi matricea $D = -C$ este simetrică şi pozitiv definită, atunci soluţia X a ecuaţiei Liapunov continue este simetrică şi pozitiv definită. Cu ipotezele de mai sus satisfăcute scrieţi un algoritm care să rezolve următoarea problemă de calcul: dat factorul Cholesky al matricei D să se calculeze factorul Cholesky al matricei X fără a calcula explicit nici matricea D nici matricea X (algoritmul Hammarling). *Indicaţie:* vezi [2].

E 2.3 Se poate demonstra că dacă matricea A satisface condiţia $|\lambda_i| < 1$, $\forall \lambda_i \in \lambda(A)$ şi matricea $D = -C$ este simetrică şi pozitiv definită, atunci soluţia X a ecuaţiei Liapunov discrete este simetrică şi pozitiv definită. Cu ipotezele de mai sus satisfăcute scrieţi un algoritm care să rezolve următoarea problemă de calcul: dat factorul Cholesky al matricei D să se calculeze factorul Cholesky al matricei X fără a calcula explicit nici matricea D nici matricea X. *Indicaţie:* vezi [2].

Bibliografie

[1] **Ionescu V., Varga A.** *Teoria Sistemelor. Sinteza Robustă. Metode Numerice de Calcul*, Ed. ALL, Bucureşti, 1994.

[2] **Jora B., Popeea C., Barbulea S.** *Metode de Calcul Numeric în Automatică*, Ed. Enciclopedică, Bucureşti, 1996.