4. *Tiesinės algebros uždavinių sprendimo metodai*

Pagrindiniai tiesinės algebros uždaviniai yra šie:

— tiesinių lygčių sistemos $A\boldsymbol{x} = \boldsymbol{b}$ sprendimas; čia A— kvadratinė matrica, $\boldsymbol{x}$ ir $\boldsymbol{b}$ — vektoriai stulpeliai;

— determinanto apskaičiavimas;

— atvirkštinės matricos apskaičiavimas;

— matricos tikrinių reikšmių ir tikrinių vektorių apskaičiavimas.

Pirmiausia aptarkime pagrindines matricų algebros sąvokas bei veiksmus, kurie reikalingi sprendžiant šiuos uždavinius.

4.1 Pagrindinės matricų algebros sąvokos ir veiksmai

Simboliu R^n žymėsime n-mačių $(n \times 1)$ vektorių, o simboliu $R^{n\times m}$ — $n \times m$ formato matricų, kurių komponentės bei elementai yra realieji skaičiai, erdves.

Apibrėžimas. $m \times n$ formato matrica, - tai dvimatis realiųjų arba kompleksinių skaičių masyvas, turintis m eilučių ir n stulpelių, kuris paprastai žymimas didžiosiomis lotynų abėcėlės raidėmis, pvz., A, B ir pan., arba $A = \left[a_{ij}\right]_{i=\overline{1,m},\, j=\overline{1,n}}$, čia a_{ij} - matricos A elementas, esantis i-toje eilutėje ir j-tajame stulpelyje.

Jei matricos elementai yra realieji skaičiai, tai ji vadinama realiųjų skaičių matrica, o jei jos elementai yra kompleksiniai skaičiai, tai – kompleksinių skaičių matrica.

Realiosios matricos paprastai žymimos $A \in R^{m\times n}$, ***kompleksinės matric***os - $A \in C^{m\times n}$, čia simbolis $C^{m\times n}$ žymi $n \times m$ formato matricų, kurių elementai yra kompleksiniai skaičiai, erdvę.

Jei $m = n$, tai matrica vadinama n-tosios eilės ***kvadratine matrica***.

Jei n-tosios eilės kvadratinė matrica A tenkina sąlyga $a_{ij} = a_{ji}$, tai matrica A vadinama ***simetrine matrica***.

Veiksmai su matricomis.

Transponavimas. $A^t = \left[a_{ji}\right]_{i=\overline{1,m},\, j=\overline{1,n}}$, tai matrica A, kurioje eilutės sukeistos su stulpeliais vietomis. Aišku, kad $\left(A^t\right)^t = A$.

Jei A yra kompleksinė matrica, tai vietoj transponavimo operacijos naudojama jungtinė transponavimo operacija, žymima simboliu „H“. Matrica A^H gaunama taip: pirmiausia matrica A transponuojama, o po to kompleksiniai skaičiai keičiami jungtiniais kompleksiniais skaičiais. Kitaip tariant, $A^H = \left(\overline{A^t}\right)$.

Pavyzdys.

```
>> a=sqrt([-1 -2; -3 4; -8 -9])
a =     0 + 1.0000i        0 + 1.4142i
        0 + 1.7321i   2.0000
        0 + 2.8284i        0 + 3.0000i
>> b=a'
b =     0 - 1.0000i        0 - 1.7321i        0 - 2.8284i
        0 - 1.4142i   2.0000                  0 - 3.0000i
```

Jei $A = A^t$, tai matrica A ***yra simetrinė.***

Jei $A = -A^t$, tai matrica A ***yra antisimetrinė.***

Transponavimo savybė. $(A * B * C)^t = C^t * B^t * A^t$.

Sumavimas. To paties formato matricas galima sudėti: $C = A + B, c_{ij} = a_{ij} + b_{ij}$.

Sandauga iš skaičiaus. $C = \alpha A, c_{ij} = \alpha a_{ij}$.

Dviejų matricų sandauga. $m \times r$ formato matricą A galima padauginti iš $r \times n$ formato matricos B : $C = A * B, c_{ij} = \sum_{k=1}^{r} a_{ik} b_{kj}, i = \overline{1,m}, j = \overline{1,n}$. Kaip matome, matricų daugyba galima, jei pirmosios matricos stulpelių skaičius yra lygus antrosios matricos eilučių skaičiui.

Kaip MATLAB'e, išraiška $A(:, j)$ žymėsime matricos A j-tąjį stulpelį. Lengva patikrinti, kad matricos $C = A * B$ j-tasis stulpelis yra lygus: $C(:, j) = A * B(:, j)$.

Matricų sandaugai negalioja ***sukeitimo*** dėsnis, t.y. $A * B \neq B * A$.

Matricos A ir vektoriaus x sandauga yra vektorius: $y = A * x$. Kadangi vektorius x yra matrica stulpelis, tai tinkamo formato matricos ir vektoriaus sandauga yra anksčiau apibrėžta dviejų matricų sandauga. Tačiau kartais patogu į matricos A ir vektoriaus x sandaugą žiūrėti kaip į matricos A stulpelių tiesinį darinį, kai darinio koeficientai yra vektoriaus x komponentės: $y = A * x = x_1 A(:,1) + x_2 A(:,2) + \ldots + x_n A(:,n)$.

Atvirkštinė matrica. Jei matrica A yra neišsigimusi (matricos A determinantas nelygus 0), tai ji turi ***atvirkštinę matricą*** A^{-1}. Atvirkštinė matrica tenkina sąlygą: $A * A^{-1} = A^{-1} * A = E$, čia E – vienetinė matrica.

Jei $A^{-1} = A^t$, tai matrica A yra ***ortogonalioji***, o jei $A^{-1} = A^t = A$, tai matrica A yra ***simetrinė ir ortogonalioji.***

Kalbant apie kompleksines matricas, naudojamos kitos sąvokos.

Jei $A^H = A$, tai matrica A yra ***Ermito*** matrica, o jei $A^{-1} = A^H$, tai matrica A yra ***unitarioji*** matrica.

Matricų sandaugos atvirkštinė matrica. $(A * B * C)^{-1} = C^{-1} * B^{-1} * A^{-1}$.

Matricų sandaugos determinantas. $\det(A * B * C) = \det(A) * \det(B) * \det(C)$.

Tiesiškai nepriklausomi vektoriai. Panagrinėkime m-elemenčių vektorių $a_1, a_2, a_3, \ldots, a_n$ aibę. Šie vektoriai yra tiesiškai nepriklausomi, jei nei vieno iš jų negalima išreikšti likusių vektorių tiesiniu dariniu. Kitaip tariant, vektoriai yra tiesiškai nepriklausomi, jei lygybė $\alpha_1 a_1 + \alpha_2 a_2 + \alpha_3 a_3 + \ldots + \alpha_n a_n = 0$ teisinga tada ir tik tada, kai visi $\alpha_i, i = \overline{1,n}$ yra lygūs nuliui.

Matricos rangas. $m \times n$ formato matricos rangas yra didžiausias tiesiškai nepriklausomų matricos eilučių arba stulpelių skaičius. Matricos rangą apskaičiuoja MATLAB'o funkcija ***rank.***

Pavyzdys. Apskaičiuokime matricų A, B ir C rangus.

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, \quad A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 2 & 8 & 9 \end{pmatrix}, \quad A = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ -3 & -6 & -9 \end{pmatrix}.$$

```
>> rank(a)
ans = 2
>> rank(b)
ans = 3
>> rank(c)
ans = 1
```

Matricos A rangas yra lygus 2, nes $A(3,:) = -A(1,:) + 2 * A(2,:)$, matricos B rangas yra lygus 3, nes visos jos eilutės yra tiesiškai nepriklausomos, o matricos C rangas yra lygus 1, nes $C(2,:) = 2 * C(1,:)$ ir $C(3,:) = -3 * C(1,:)$.

Aišku, kad matricos A rangas tenkina nelygybę: $rank(A) \le \min(m,n)$.

Stulpelių erdvė. Matricos A nepriklausomi stulpeliai sudaro matricos stulpelių erdvės bazę, t.y. kiekvienas vektorius, kuris yra jų tiesinis darinys, priklauso šiai erdvei.

Nulio erdvė. Matricos A nulio erdvė (ji žymima $null(A)$) yra aibė visų vektorių x, tenkinančių sąlygą: $Ax = 0$.

MATLAB'as turi vidinę matricos nulio erdvės apskaičiavimo funkciją, kreipinys į kurią yra: ***z= null(A)***, čia ***z*** – nulio erdvės ortonormuota bazė.

Matrica su dominuojančia įstrižaine. Jei $n \times n$ formato matrica A tenkina sąlygą $|a_{ii}| > \sum_{j=1, j \ne i}^{n} |a_{ij}|$, tai sakoma, kad matrica A yra su griežtai dominuojančia įstrižaine.

Tokios matricos determinantas nelygus nuliui, t.y. ji yra neišsigimusi.

Teigiamai apibrėžta matrica. Matrica A yra teigiamai apibrėžta, jei ji yra simetrinė ir tenkina sąlygą: $(Ax, x) = x^t Ax > 0, x \ne 0$, čia išraiška (x, y) žymi vektorių x ir y skaliarinę sandaugą.

Remiantis apibrėžimu, sunku nustatyti ar matrica yra teigiamai apibrėžta. Tačiau laimei yra eilė lengviau patikrinamų kriterijų šiam faktui nustatyti.

1. ***Simetrinė matrica A yra teigiamai apibrėžta tada ir tiktai tada, kai jos visos tikrinės reikšmės yra teigiamos.*** Matricos tikrinių reikšmių apskaičiavimo uždavinys nagrinėjamas vėliau.
2. ***Jei simetrinė matrica A turi dominuojančią įstrižainę ir visi jos pagrindinės įstrižainės elementai yra teigiami, tai matrica A yra teigiamai apibrėžta.***
3. ***Simetrinė matrica A yra teigiamai apibrėžta tada ir tiktai tada, kai $\det A_i > 0, i = 1,2,\ldots,n$, čia $\det A_i$ yra matricos A i-os eilės pagrindinis determinantas.***

Teigiamai apibrėžtos matricos savybės.

1. ***Jei matrica A yra teigiamai apibrėžta, tai $\det A \ne 0$.***
2. ***Jei matrica A yra teigiamai apibrėžta, tai $a_{ii} > 0, i = 1,2,\ldots,n$.***
3. ***Jei matrica A yra teigiamai apibrėžta, tai $(a_{ij})^2 < a_{ii} a_{jj}, i \ne j$.***
4. ***Jei matrica A yra teigiamai apibrėžta, tai $\max_{1 \le k, j \le n} |a_{kj}| \le \max_{1 \le i \le n} |a_{ii}|$.***
5. ***Teigiamai apibrėžtų matricų suma yra teigiamai apibrėžta.***
6. ***Kiekviena teigiamai apibrėžtos matricos tikrinė reikšmė yra teigiama.***

Matricos singuliarinis skaidinys.

Teorema. *Jei A – realioji $m \times n$ matrica, tai egzistuoja tokios ortogonaliosios matricos* $U = [u_1, u_2, \ldots, u_m] \in R^{m \times m}$ *ir* $V = [v_1, v_2, \ldots, v_n] \in R^{n \times n}$, *kad*

$A = U * diag(\sigma_1, \sigma_2, \ldots, \sigma_p) * V^t$, *čia* $p = \min(m,n)$, *o diag yra pseudoįstrižaininė* $m \times n$ *formato matrica, kurios pseudoįstrižainės elementai yra skaičiai* σ_i.

Skaičiai σ_i *yra matricos A singuliarinės reikšmės.*

Skaičiai σ_i yra neneigiamos kvadratinės šaknys iš matricos $A * A^t$ tikrinių reikšmių ir apibrėžiamos vieninteliu būdu.

Matricos *U* stulpeliai yra matricos $A * A^t$ tikriniai vektoriai, o matricos *V* stulpeliai yra matricos $A^t * A$ tikriniai vektoriai; abi vektorių sistemos išdėstytos atitinkamai tikrinėms reikšmėms σ_i^2.

Matricos *A* singuliarinis skaidinys duoda daug informacijos apie pačią matricą.

Tarkime, kad matricos *A* singuliarinės reikšmės tenkina sąlygą:

$\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \ldots \geq \sigma_r > \sigma_{r+1} = \ldots = \sigma_p = 0$. Tada matricos *A* rangas yra lygus *r*, t.y. $rank(A) = r$, be to $null(A) = span\{v_{r+1}, v_{r+2}, \ldots, v_n\}$, o $range(A) = span\{u_1, u_2, \ldots, u_r\}$, čia $range(A)$ yra matricos $A \in R^{m \times n}$ reikšmių sritis: $range(A) = \{y \in R^m : y = Ax, x \in R^n\}$. Kitaip tariant, $v_{r+1}, v_{r+2}, \ldots, v_n$ yra matricos *A* nulio erdvės ortonormuota bazė, o $u_1, u_2, \ldots, u_r$ yra matricos *A* stulpelių erdvės ortonormuota bazė.

MATLAB'o funkcija ***svd*** atlieka matricos singuliarinį skaidinį.

Pavyzdys.

```
>> a=[1 -3 5 6; 2 0 6 -4; 8 4 7 -9]
a =  1    -3     5     6
     2     0     6    -4
     8     4     7    -9
>> [u,s,v]=svd(a)
u =-0.0940   -0.9605    0.2618
    0.4201   -0.2767   -0.8643
    0.9026    0.0288    0.4296
s = 6.0047         0         0         0
         0    8.6020         0         0
         0         0    2.6184         0
v = 0.4978   -0.1492    0.7523   -0.4050
    0.2432    0.3484    0.3562    0.8322
    0.5229   -0.7279   -0.3321    0.2940
   -0.6478   -0.5714    0.4437    0.2386
```

Matricos ***a*** singuliarinis skaidinys rodo, kad jos rangas yra 3, matricos ***u*** stulpeliai yra matricos ***a*** stulpelių erdvės ortonormuota bazė, o matricos ***v*** ketvirtasis stulpelis yra matricos ***a*** nulio erdvės ortonormuota bazė.

4.2 Vektoriaus ir matricos normos

Vektoriaus bei matricos normos yra realiųjų skaičių modulio apibendrinimas.

Vektoriaus normos

Apibrėžimas. *n*-mačio vektoriaus $\boldsymbol{v} \in R^n$ norma $\|\boldsymbol{v}\|$ vadinamas skaičius, tenkinantis tokias sąlygas:

1) $\|\boldsymbol{v}\| > 0$, kai $\boldsymbol{v} \neq 0$, ir $\|\boldsymbol{v}\| = 0$, kai $\boldsymbol{v} = 0$;

2) $\|\alpha\, \boldsymbol{v}\| = |\alpha| \cdot \|\boldsymbol{v}\|$; čia α — skaičius;

3) $\|\boldsymbol{u}+\boldsymbol{v}\| \le \|\boldsymbol{u}\| + \|\boldsymbol{v}\|, \quad \boldsymbol{u},\boldsymbol{v} \in R^n$ (trikampio nelygybė).

Šis apibrėžimas vektoriaus normą apibūdina nevienareikšmiškai. Galima nurodyti be galo daug normų, tenkinančių apibrėžime iškeltas sąlygas.

Dažniausiai yra vartojamos tokios vektoriaus normos.

Tarkime, kad

$$\boldsymbol{v} = \left(v_1, v_2, \ldots, v_n\right)^t \in R^n$$

(simbolis *t* žymi transformavimo operaciją).

Tada

$$\|\boldsymbol{v}\|_\infty = \max_{1\le i\le n} |v_i| \quad \left(l_\infty \text{ norma}\right),$$

$$\|\boldsymbol{v}\|_1 = \sum_{i=1}^{n} |v_i| \qquad \left(l_1 \text{ norma}\right), \ \|\boldsymbol{v}\|_2 = \left(\sum_{i=1}^{n} v_i^{\,2}\right)^{\frac{1}{2}} \quad \left(l_2 \text{ norma}\right).$$

Visos šios normos yra atskiri atvejai l_p normos, apibrėžiamos taip:

$$\|\boldsymbol{v}\|_p = \left(\sum_{i=1}^{n} |v_i|^p\right)^{\frac{1}{p}}, \ 1 \le p \le \infty.$$

Matricos normos

Panagrinėkime matricos $A = \left(a_{ij}\right)$ $\left(\text{čia } i = \overline{1,n},\ j = \overline{1,n}\right)$ normas.

Į matricą *A* galima žiūrėti dvejopai:

1) kaip į vektorių $\left(a_{11}, a_{21}, \ldots, a_{n1}, a_{12}, \ldots, a_{nn}\right)^t$;

2) kaip į operatorių, transformuojantį erdvę R^n pačią į save.

Pirmuoju atveju gali būti vartojama bet kuri iš anksčiau apibrėžtų vektoriaus normų, pavyzdžiui, ***Frobeniuso norma*** $\|A\|_F$, kuri yra vektoriaus

$$\left(a_{11}, a_{21}, \ldots, a_{n1}, a_{12}, \ldots, a_{nn}\right)^t$$

l_2 norma, t. y.

$$\|A\|_F = \left(\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}^{\,2}\right)^{\frac{1}{2}}.$$

Yra žinoma, kad $\|A\|_F = \left(\sum_{i=1}^{n} \sigma_i^2\right)^{\frac{1}{2}}$, čia σ_i yra matricos *A* singuliarinės reikšmės.

Antruoju atveju matrica *A* kiekvieną vektorių $\boldsymbol{v}$ transformuoja į vektorių $A\boldsymbol{v}$. Norėtųsi žinoti vektoriaus $A\boldsymbol{v}$ normos ribas. Todėl matricos, kaip operatoriaus, norma apibrėžiama taip:

$$\|A\| = \max_{\boldsymbol{v}\in R^n,\, \boldsymbol{v}\ne 0} \left\{ \frac{\|A\boldsymbol{v}\|}{\|\boldsymbol{v}\|} \right\}. \qquad (4.1)$$

Tai skaičius, rodantis, kiek daugiausia kartų matrica A gali ištempti bet kurį vektorių erdvėje R^n. Šioje formulėje gali būti vartojama bet kuri anksčiau apibrėžta vektoriaus norma. (1) formule apibūdinta norma vadinama ***operatorine matricos norma***. Ją indukuoja vektoriaus norma $\|\cdot\|$. Jei (4.1) formulėje vartojama vektoriaus l_p norma, tai operatorinė matricos A norma žymima $\|A\|_p$. Aišku, kad, $\boldsymbol{u},\boldsymbol{v} \in R^n$ pakeitus į $A, B \in R^{n\times n}$, ji tenkina vektoriaus normos sąlygas.

(4.1) formulė nusako optimizavimo uždavinį. Jo sprendiniai, kai vartojamos l_1, l_2 ir l_∞ vektoriaus normos, yra žinomi:

$$\|A\|_1 = \max_{1\le j\le n}\left\{ \| A(:,j)\|_1 \right\} = \max_{1\le j\le n}\sum_{i=1}^{n}\left| a_{ij} \right|,$$

$\|A\|_2 =$ (didžiausia matricos $A^t \cdot A$ tikrinė reikšmė)$^{1/2}$=σ_1, čia σ_1 - matricos A pirmoji singuliarinė reikšmė.

$$\|A\|_\infty = \max_{1\le i\le n}\left\{ \| A(i,:)\|_1 \right\} = \max_{1\le i\le n}\sum_{j=1}^{n}\left| a_{ij} \right|.$$

Simbolis $A(i,:)$ čia žymi matricos A i-tąją eilutę, o simbolis $A(:,j)$ — j-tąjį stulpelį.

Matricų normos, be anksčiau minėtų sąlygų, tenkina nelygybę

$\|AB\| \le \|A\| \cdot \|B\|$.

Pavyzdžiai.

1. *Turime vektorių* $x = \begin{pmatrix} 6 & 3 & 2 & 5 & 7 & 8 & 4 & 1 \end{pmatrix}$**.**

Apskaičiuokime $\|x\|_1$ **,** $\|x\|_2$ ***ir*** $\|x\|_\infty$**.**

Remdamiesi anksčiau pateiktais apibrėžimais, turėsime:

$\|x\|_1 = 36,\ \|x\|_2 = 14.2829,\ \|x\|_\infty = 8.$

2*. Turime matricą* ***A:***

$$A = \begin{pmatrix} 24 & 2 & 7 & 6 & 20 \\ 8 & 22 & 4 & 12 & 18 \\ 21 & 16 & 3 & 23 & 13 \\ 25 & 19 & 11 & 9 & 17 \\ 15 & 10 & 14 & 5 & 1 \end{pmatrix}.$$

Apskaičiuokime $\|A\|_F$**,** $\|A\|_1$ **,** $\|A\|_2$ ***ir*** $\|A\|_\infty$**.**

Remdamiesi anksčiau pateiktais apibrėžimais, turėsime:

$\|A\|_F = 74.3303,\ \|A\|_1 = 93,\ \|A\|_2 = 68.9133,\ \|A\|_\infty = 81.$

4.3. Tiesinių lygčių sistemos sprendimo metodai

Tiesinių lygčių sistema matricinėje formoje užrašoma taip: $A\boldsymbol{x} = \boldsymbol{b}$; čia A — $m\times n$ formato matrica, o $\boldsymbol{x}$ yra $n\times 1$ formato, o $\boldsymbol{b}$ — $m\times 1$ formato vektoriai. Praktiniuose

uždaviniuose dažniausiai $m = n$. Vektorius $\boldsymbol{x}$ yra sistemos nežinomųjų vektorius, o vektorius $\boldsymbol{b}$ — sistemos laisvųjų narių vektorius.

Matrica A vadinama lygčių sistemos matrica, o matrica A^* (pagal MATLAB'o sintaksę $A^* = [\mathrm{A\ b}]$), gauta prie matricos A prijungiant laisvųjų narių stulpelį $\boldsymbol{b}$, vadinama sistemos išplėstine matrica.

Teorema. *Jei matricos A rangas rank(A) yra lygus išplėstinės matricos A^* rangui rank(A), tai lygčių sistema yra suderinta — turi bent vieną sprendinį; priešingu atveju, — sistema nesuderinta — neturi sprendinių. Jei rank(A) = rank(A^*) = n ,tai lygčių sistema turi vienintelį sprendinį, o jei rank(A) = rank(A^*) < n, tai lygčių sistema turi be galo daug sprendinių.*

Tiesinių lygčių sistemos sprendimo metodus galima suskirstyti į tiksliuosius ir apytikslius.

Tikslieji metodai — tai tokie metodai, kurie padeda rasti sprendinį, atlikus baigtinį skaičių aritmetinių veiksmų. Jei matricos A koeficientai bei lygčių sistemos laisvieji nariai yra tikslūs ir skaičiuojama be apvalinimo paklaidų, tai šiais metodais gaunami tikslūs sprendiniai. Prie tiksliųjų metodų priskiriamas Kramerio metodas, Gauso metodas, atvirkštinės matricos metodas bei daugelis kitų.

Apytiksliai metodai yra iteraciniai. Juos taikant, randamas apytikslis lygčių sistemos sprendinys. Renkantis šiuos metodus, svarbu žinoti, kokiomis sąlygomis ir kokiu greičiu jie konverguoja. Apytiksliams metodams priklauso paprastųjų iteracijų (Jakobio) metodas, Zeidelio metodas ir kt. Labai platus iteracijų metodų spektras pateiktas literatūroje: *R. Čiegis, V. Būda, Skaičiuojamoji matematika, TEV, Vilnius, 1997*.

4.3.1. Gauso metodas

Tiesinių lygčių sistemos sprendimas Kramerio bei atvirkštinės matricos metodais turi daugiau teorinę nei praktinę reikšmę. Vienas iš dažniausiai taikomų tiesinių lygčių sistemos sprendimo metodų yra Gauso metodas bei įvairios jo modifikacijos.

Tarkime, kad sprendžiame tiesinių lygčių sistemą $A\boldsymbol{x} = \boldsymbol{b}$; čia A — n-tosios eilės kvadratinė matrica, $\boldsymbol{x}$ ir $\boldsymbol{b}$ — vektoriai stulpeliai.

Išskleiskime duotąją sistemą:

$$\begin{cases} a_{11}x_1 + \ldots + a_{1k}x_k + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n = b_1, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{k1}x_1 + \ldots + a_{kk}x_k + \ldots + a_{kj}x_j + \ldots + a_{kn}x_n = b_k, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1 + \ldots + a_{nk}x_k + \ldots + a_{nj}x_j + \ldots + a_{nn}x_n = b_n. \end{cases}$$

Gauso metodas susideda iš dviejų etapų: tiesioginio ir atvirkštinio.

Tiesioginiame etape sistema $A\boldsymbol{x} = \boldsymbol{b}$, taikant elementarius pertvarkius (elementarus pertvarkis – tai veiksmas, kai iš i-tos lygties atimama k-oji lygtis padauginta iš pasirinkto daugiklio s), perskaičiuojama į trikampę išraišką $R\boldsymbol{x} = \boldsymbol{c}$; čia R— viršutinė trikampė matrica. Atvirkštiniame etape nuosekliai, pradedant n-tąja lygtimi, apskaičiuojami nežinomieji: $x_n, x_{n-1}, \ldots, x_1$.

Tiesioginį etapą sudaro n – 1 žingsnis. k-tuoju žingsniu pirmosios k lygtys nekeičiamos, o iš $k+1, \ldots, n$ lygčių pašalinamas kintamasis x_k. Tam tikslui iš i-tosios

($i = \overline{k+1, n}$) lygties atimama k-toji lygtis, padauginta iš tokio daugiklio s, su kuriuo $a_{ik} - a_{kk} \cdot s = 0$. Vadinasi, $s = \dfrac{a_{ik}}{a_{kk}}$. Kiti i-tosios lygties koeficientai bei laisvasis narys perskaičiuojami pagal formules: $a_{ij} := a_{ij} - s \cdot a_{kj}$ $\left(j = \overline{k+1, n}\right)$, $b_i := b_i - s \cdot b_k$.

Literatūroje nurodyta, kad, perskaičiuojant a_{ij} ir b_i, apvalinimo paklaidos mažiausios esti tada, kai s yra kuo mažesnis. Vadinasi, kiekviename žingsnyje a_{kk} turi būti k-tojo stulpelio didžiausio modulio elementas, t. y. $\left|a_{kk}\right| = \max\limits_{k \le i \le n} \left|a_{ik}\right|$. Tai vadinamasis pagrindinio elemento parinkimo būdas.

Pastaba. Be šio, dažniausiai taikomo pagrindinio elemento parinkimo būdo, yra ir kitų būdų:

1. a) Randamas $\left|a_{lt}\right| = \max\limits_{\substack{k \le i \le n, \\ k \le j \le n}} \left|a_{ij}\right|$;

b) sistemos l-toji ir k-toji lygtis, taip pat matricos A t-tasis bei k-tasis stulpelis sukeičiami vietomis, dėl to elementas a_{lt} pakyla į elemento a_{kk} vietą. Šiuo atveju tenka pernumeruoti kintamuosius.

2. a) Apskaičiuojamas $\left|a_{it}\right| = \max\limits_{k \le j \le n} \left|a_{ij}\right|$, $i = \overline{k, n}$, t. y. randamas kiekvienos lygties didžiausio modulio koeficientas;

b) randamas $\left|s_l\right| = \max\limits_{k \le i \le n} \left|s_i\right|$; čia $s_i = \sum\limits_{j=k}^{n} \left|\dfrac{a_{ij}}{a_{it}}\right|$;

c) k-toji ir l-toji lygtys sukeičiamos vietomis.

1. Gauso metodu išspęsime lygčių sistemą: $\begin{cases} x_1 - 2x_2 + 3x_3 = 2, \\ -x_1 + x_2 + 3x_3 = 3, \\ 2x_1 - x_2 + x_3 = 2. \end{cases}$

Tiesioginis etapas.

1-as žingsnis

Šiame žingsnyje iš (2) ir (3) lygčių pašalinsime x_1. Tam tikslui iš antrosios lygties atimsime pirmąją lygtį padauginta iš -1, o po to, iš trečiosios lygties atimsime pirmąją lygtį padaugintą iš 2. Turėsime:

$$\begin{cases} x_1 - 2x_2 + 3x_3 = 2, \\ 0x_1 - x_2 + 6x_3 = 5, \\ 0x_1 + 3x_2 - 5x_3 = -2. \end{cases}$$

2-as žingsnis

Šiame žingsnyje iš trečiosios lygties pašalinsime x_2. Tam tikslui iš trečiosios lygties atimsime antrąją lygtį padaugintą iš -3. Turėsime:

$$\begin{cases} x_1 - 2x_2 + 3x_3 = 2, \\ 0x_1 - x_2 + 6x_3 = 5, \\ 0x_1 + 0x_2 + 13x_3 = 13. \end{cases}$$

Tiesioginis etapas baigtas,- lygčių sistema perskaičiuota į trikampį pavidalą.

Atvirkštinis etapas.

Iš trečiosios lygties apskaičiuosime x_3.

$$x_3 = \frac{13}{13} = 1.$$

Turėdami x_3, iš antrosios lygties apskaičiuosime x_2.

$$x_2 = \frac{5 - 6x_3}{-1} = \frac{5 - 6*1}{-1} = 1.$$

Turėdami x_3 ir x_2, iš pirmosios lygties apskaičiuosime x_1.

$$x_1 = \frac{2 + 2x_2 - 3x_3}{1} = \frac{2 + 2*1 - 3*1}{1} = 1.$$

Vadinasi, sistemos sprendinys yra: $x_1 = 1, x_2 = 1, x_3 = 1$.

2. Gauso metodu išspęsime lygčių sistemą:

$$\begin{cases} x_1 - 3x_2 + 3x_3 = 1, \\ -x_1 + 2x_2 + 3x_3 = 4, \\ x_1 - 4x_2 + 9x_3 = 6. \end{cases}$$

Tiesioginis etapas.

1-as žingsnis

Šiame žingsnyje iš (2) ir (3) lygčių pašalinsime x_1. Tam tikslui iš antrosios lygties atimsime pirmąją lygtį padauginta iš -1, o po to, iš trečiosios lygties atimsime pirmąją lygtį padaugintą iš 1. Turėsime:

$$\begin{cases} x_1 - 3x_2 + 3x_3 = 1, \\ 0x_1 - x_2 + 6x_3 = 5, \\ 0x_1 - x_2 + 6x_3 = 5. \end{cases}$$

2-as žingsnis

Šiame žingsnyje iš trečiosios lygties pašalinsime x_2. Tam tikslui iš trečiosios lygties atimsime antrąją lygtį padaugintą iš 1. Turėsime:

$$\begin{cases} x_1 - 3x_2 + 3x_3 = 1, \\ 0x_1 - x_2 + 6x_3 = 5, \\ 0x_1 + 0x_2 + 0x_3 = 0. \end{cases}$$

Atvirkštinis etapas.

Iš trečiosios lygties turėtume apskaičiuoti x_3. Kadangi x_3 koeficientas yra lygus nuliui, o taip pat laisvasis narys lygus nuliui, tai reiškia, kad x_3 reikšmė gali būti bet koks skaičius. Vadinasi, ši lygčių sistema turi be galo daug sprendinių.

3. Gauso metodu išspęsime lygčių sistemą:

$$\begin{cases} x_1 - 3x_2 + 5x_3 = 3, \\ -x_1 + 2x_2 + 3x_3 = 4, \\ x_1 - 4x_2 + 13x_3 = 6. \end{cases}$$

Tiesioginis etapas.

1-as žingsnis

Šiame žingsnyje iš (2) ir (3) lygčių pašalinsime x_1. Tam tikslui iš antrosios lygties atimsime pirmąją lygtį padauginta iš -1, o po to, iš trečiosios lygties atimsime pirmąją lygtį padaugintą iš 1. Turėsime:

$$\begin{cases} x_1 - 3x_2 + 5x_3 = 3, \\ 0x_1 - x_2 + 8x_3 = 7, \\ 0x_1 - x_2 + 8x_3 = 3. \end{cases}$$

2-as žingsnis

Šiame žingsnyje iš trečiosios lygties pašalinsime x_2. Tam tikslui iš trečiosios lygties atimsime antrąją lygtį padaugintą iš 1. Turėsime:

$$\begin{cases} x_1 - 3x_2 + 5x_3 = 3, \\ 0x_1 - x_2 + 8x_3 = 7, \\ 0x_1 + 0x_2 + 0x_3 = -4. \end{cases}$$

Atvirkštinis etapas.

Iš trečiosios lygties turėtume apskaičiuoti x_3. Kadangi x_3 koeficientas yra lygus nuliui, o laisvasis narys nelygus nuliui, tai reiškia, kad nėra tokios x_3 reikšmės, kad ją padauginus iš nulio turėtume laisvąjį narį: -4. Vadinasi, ši lygčių sistema yra nesuderinta, t.y. neturi sprendinių.

Gauso metodo algoritmo tiesioginio etapo k-tasis žingsnis $\left(k = \overline{1, n-1}\right)$

Atlikę k – 1 žingsnį, turime lygčių sistemą

$$\begin{cases} a_{11}x_1 + \ldots + a_{1k}x_k + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n = b_1, \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{kk}x_k + \ldots + a_{kj}x_j + \ldots + a_{kn}x_n = b_k, \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{ik}x_k + \ldots + a_{ij}x_j + \ldots + a_{in}x_n = b_i, \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{nk}x_k + \ldots + a_{nj}x_j + \ldots + a_{nn}x_n = b_n. \end{cases}$$

1) Randame k-tojo stulpelio didžiausio modulio elementą

$|a_{lk}| = \max\limits_{k \le i \le n} |a_{ik}|$.

Jei $|a_{lk}| = 0$, tai lygčių sistema arba ***nesuderinta***, arba turi ***be galo daug sprendinių***. Šiuo atveju sprendimas baigtas. Priešingu atveju, jei $k \neq l$, tai k-tąją ir l-tąją lygtis sukeičiame vietomis.

2) Perskaičiuojame $k+1, \ldots, n$ lygtis pagal formules:

$s = \frac{a_{ik}}{a_{kk}},\ a_{ij} := a_{ij} - s \cdot a_{kj}\ \left(j = \overline{k,n}\right), b_i := b_i - s \cdot b_k,\ i = k+1, k+2, \ldots, n.$

Atliekant Gauso metodo tiesioginio etapo (*n-1*)-ąjį žingsnį, nuliu verčiamas $a_{n,n-1}$ koeficientas. Jei lygčių sistemoje $A\boldsymbol{x} = \boldsymbol{b}$ yra viena lygtis, tiesiškai priklausoma nuo kitų lygčių, tai (*n-1*)-me žingsnyje perskaičiuotas koeficientas a_{nn} bus lygus nuliui. Vadinasi, prieš pradėdami Gauso metodo atvirkštinį etapą turime patikrinti ar koeficientas a_{nn} yra lygus nuliui. Jei $a_{nn} = 0$, tai reiškia, kad lygčių sistema $A\boldsymbol{x} = \boldsymbol{b}$ yra arba ***nesuderinta*** (neturi sprendinių), arba turi ***be galo daug sprendinių***. Abiems atvejais lygčių sistemos sprendimą nutraukiame.

Atvirkštinis etapas

Po Gauso metodo tiesioginio etapo turime lygčių sistemą:

$$\begin{cases} a_{11}x_1 + \ldots + a_{1i}x_i + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n = b_1, \\ \qquad\qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \qquad\qquad a_{ii}x_i + \ldots + a_{ij}x_j + \ldots + a_{in}x_n = b_i, \\ \qquad\qquad\qquad\qquad\qquad\qquad \ldots\ldots\ldots\ldots \\ \qquad\qquad\qquad\qquad\qquad\qquad a_{nn}x_n = b_n, \end{cases}$$

Aišku, kad atvirkštinis Gauso metodo etapas yra aprašomas formule:

$$x_i = \frac{b_i - \sum_{j=i+1}^{n} a_{ij}x_j}{a_{ii}}, i = n, n-1, \cdots, 2, 1.$$

Žemiau patalpinta šį algoritmą realizuojanti funkcija.

```
function [x,salsk,p] = gausom(a,b)
% GAUSOM funkcija sprendžia tiesinių lygčių sistemą Gauso metodu.
% Įėjimo parametrai
% a -sistemos matrica,
% b – laisvųjų narių eilute arba stulpelis.
% Išėjimo parametrai
% x     - sistemos sprendinys,
% salsk - sistemos sąlygotumo skaičius,
% p     - požymis: jei p=1, tai sistema turi vienintelį sprendinį;
%                  jei p=0, tai sistema arba nesuderinta,
%                           arba turi be galo daug sprendiniu.

salsk=cond(a);
n=length(b);
[m,t]=size(b);
if m == 1
    b=b';
end;
x=zeros(1,n);
% Tiesioginis etapas
p=1; k=1;
while (p ==1 ) & (k <= n-1)
  [d,l]=max(abs(a(k:n,k))); % pagrindinio elemento apskaičiavimas
  l=l+k-1;
  if d <= eps
```

```
            p=0;
            disp('sistema nesuderinta arba turi daug sprendinių')
        else
            if l ~= k % l ir k lygtis keičiame vietomis
                z=a(k,:); a(k,:)=a(l,:); a(l,:)=z;
                z=b(k); b(k)=b(l); b(l)=z;
             end; % perskaičiuojame k+1,...,n lygtis
                s=a(k+1:n,k)/a(k,k);
                a(k+1:n,k)=0;
                a(k+1:n,k+1:n)=a(k+1:n,k+1:n)-s*a(k,k+1:n);
                b(k+1:n)=b(k+1:n)-s*b(k);
            end;
   k=k+1;
 end;
% Atvirkštinis etapas
          if abs(a(n,n)) <= eps
            p=0;
             disp('sistema nesuderinta arba turi daug sprendinių ')
         else
            for i = n:-1:1
                x(i)=(b(i)-a(i,i+1:n)*x(i+1:n)')/a(i,i);
            end;
        end;
```

Gauso metodo skaičiavimo apimtis. Aptarkime, kiek aritmetinių veiksmų reikia atlikti sprendžiant tiesinių lygčių sistemą Gauso metodu.

Iš pateikto algoritmo (tiesioginiame etape jis turi trigubą, o atvirkštiniame — dvigubą ciklą) matyti, kad veiksmų skaičius proporcingas n^3, t. y. $O\left(n^3\right)$. Kaip nurodyta literatūroje *R. Čiegis, V. Būda, Skaičiuojamoji matematika, TEV, Vilnius, 1997*, tikslus aritmetinių veiksmų skaičius apskaičiuojamas pagal šias formules:

sudėties veiksmų skaičius $s = \dfrac{n(n-1)(2n+5)}{6}$,

daugybos veiksmų skaičius $d = \dfrac{n\left(n^2+3n-1\right)}{3}$.

Jei n — didelis skaičius, tai $s \approx d \approx \dfrac{n^3}{3}$. Vadinasi, Gauso metodo skaičiavimo apimtis yra $O\left(\dfrac{2}{3}n^3\right)$.

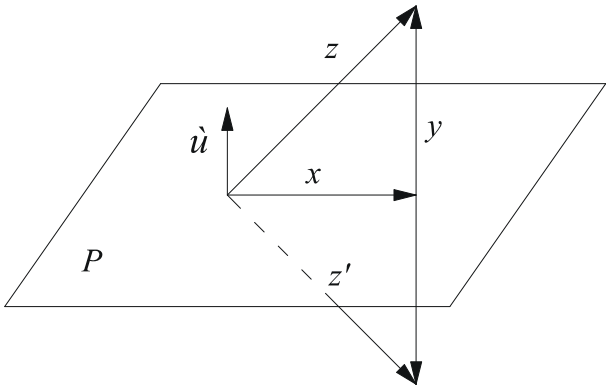
Simbolis "*O"* reiškia žodžius: "*tos pačios eilės skaičius*". Vadinasi, simbolis $O\left(\dfrac{2}{3}n^3\right)$ reiškia, kad, sprendžiant lygčių sistemą $A\boldsymbol{x} = \boldsymbol{b}$ Gauso metodu, atliekamų veiksmų skaičius yra tos pačios eilės skaičius, kaip ir skaičius $\dfrac{2}{3}n^3$. Šis skaičius rodo, kad Gauso metodas yra efektyvus. Pavyzdžiui, jei sistemos lygčių skaičius lygus 100, tai, ją sprendžiant Gauso metodu, reikės atlikti apie 10^6 aritmetinių veiksmų. Šiuolaikiniai kompiuteriai tokį veiksmų skaičių atlieka per sekundės dalis.

4.3.2. QR metodas

Šis metodas literatūroje dar vadinamas atspindžio metodu. Sprendžiant sistemą *QR* metodu, kaip ir Gauso metode, pirmiausia lygčių sistema perskaičiuojama į trikampį pavidalą, o po to, taikant Gauso metodo atvirkštinį etapą, apskaičiuojami $x_n, x_{n-1}, \ldots, x_1$. Pagrindinis *QR* metodo skirtumas nuo Gauso metodo yra tas, kad perskaičiuojant sistemą į trikampį pavidalą vietoje elementarių pertvarkių yra naudojamos atspindžio matricos.

Pirmiausia išnagrinėkime atspindžio transformaciją ir kai kurias jos savybes. Tarkime, kad erdvėje R^3 duota plokštuma P, o vienetinis vektorius $\boldsymbol{\omega}$ yra plokštumos normalės vektorius. Raskime bet kurio erdvės R^3 vektoriaus $\boldsymbol{z}$ atspindžio plokštumos P atžvilgiu vektorių $\boldsymbol{z}'$ (žr. 4.1 pav.).

Matrica, kuri vektorių $\boldsymbol{z}$ transformuoja į vektorių $\boldsymbol{z}'$, vadinama atspindžio matrica ir nusakoma formule: $Q = E - 2\boldsymbol{\omega}\cdot\boldsymbol{\omega}^t$.



4.1 pav. Atspindžio transformacija erdvėje R^3. $\boldsymbol{z}'$ *— vektoriaus* $\boldsymbol{z}$ *atspindžio vektorius.*

Analogiškai atspindžio transformacija nusakoma *n*-matėje erdvėje R^n.

Atspindžio matricos savybės:

1. Atspindžio transformacija nekeičia transformuojamo vektoriaus ilgio.
2. Atspindžio matrica yra simetrinė ir ortogonalioji.
3. Žinant to paties ilgio vektorius $\boldsymbol{z}$ ir $\boldsymbol{z}'$, galima apskaičiuoti plokštumą *P,* atžvilgiu kurios šie vektoriai yra simetriški. Aišku, kad plokštumos *P* vienetinis normalės vektorius apskaičiuojamas pagal formule $\omega = (z - z')/|z - z'|$, o atspindžio matricą *Q*, transformuojanti vektorių $\boldsymbol{z}$ į vektorių $\boldsymbol{z}'$ ($\boldsymbol{z}' = Q\boldsymbol{z}$), yra: $Q = E - 2\boldsymbol{\omega}\cdot\boldsymbol{\omega}^t$.

QR metodo idėja. Turime tiesinių lygčių sistemą $A\boldsymbol{x} = B$. Matricą *A* transformuokime į viršutinę trikampę matricą *R*, $n-1$ kartą iš kairės daugindami matricą *A* iš atspindžio matricų Q_k. Kiekviena matrica Q_k verčia nuliais matricos $Q_{k-1}\ldots Q_1 A$ *k*-tojo stulpelio elementus, esančius žemiau pagrindinės įstrižainės, ir nekeičia pirmųjų $(k-1)$ matricos stulpelių. Po tokios transformacijos turime $Q_{n-1}Q_{n-2}\ldots Q_1 A = R$. Pažymėję $B^{\bullet} = Q_{n-1}\ldots Q_2 Q_1 B$, turėsime sistemą $R\boldsymbol{x} = B^{\bullet}$. Ji sprendžiama taikant atvirkštinį Gauso metodo etapą.

QR metodas, lyginant su Gauso metodu, turi tą privalumą, kad formuojamos matricos R elementai yra tos pačios eilės, kaip ir matricos A, nes $\|R\|_2 = \|QR\|_2 = \|A\|_2$. Taigi skaičiavimo požiūriu šis metodas yra labai stabilus.

Aptarsime, kaip ieškoma atspindžio matrica Q_k, tenkinanti anksčiau minėtas sąlygas. Šis uždavinys ekvivalentus tokiam uždaviniui: duoti vienodo ilgio vektoriai $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^t$ ir $\mathbf{y} = -sign(x_1)|\mathbf{x}|(1,0,\ldots,0)^t$ (taip paėmus vektorių y skaičiavimo paklaidos bus mažesnės); reikia rasti tokį vektorių $\boldsymbol{u}$, kuris būtų plokštumos P, kurios atžvilgiu vektoriai $\boldsymbol{x}$ ir $\boldsymbol{y}$ yra simetriški, normalės vektorius.

Pagal atspindžio matricos 3 savybę, vektorius $\boldsymbol{u} = \boldsymbol{x} - \boldsymbol{y}$. Norint apskaičiuoti atspindžio matricą, kuri vektorių $\boldsymbol{x}$ transformuoja į vektorių $\boldsymbol{y}$, reikia turėti vektoriaus $\boldsymbol{u}$ vienetinį vektorių. Vadinasi, $\omega = (x - y)/|x - y|$.

Iš pateikto nagrinėjimo matyti, kad matrica Q_k apskaičiuojama taip.

1. sudarome vektorių $u \in R^n$ pagal formulę: $u = (0,0,\ldots,0,a_{kk},a_{k+1,k},\ldots,a_{nk})$,
2. perskaičiuojame elementą $u(k,k) = u(k,k) + sign(u(k,k)) * |u|$,
3. apskaičiuojame vektorių $\omega = u/|u|$,
4. apskaičiuojame atspindžio matricą $Q_k = E - 2\omega\omega^t$.

Žemiau pateikta funkcija *qrm,* kuri realizuoja aptartą *QR* metodą. Kadangi vektoriaus ω apskaičiavimas nepriklauso nuo vektoriaus *u* ilgio, tai vektorius *u* yra normuojamas tam, kad skaičiavimo paklaidos būtų mažesnės. Be to, dauginant iš kairės matricą A iš atspindžio matricos Q_k patogu pasinaudoti tuo, kad matricų sandaugos *j*-sis stulpelis yra lygus Q_k ir matricos $Q_{k-1}\ldots Q_1 A$ *j*-jo stulpelio sandaugai.

```
function [x,salsk,p] = qrm(a,b)
% QRM funkcija QR metodu sprendžia tiesinių lygčių sistemą a*x=b.
% Įėjimo parametrai
% a - n-ios eiles kvadratinė matrica,
% b - laisvųjų narių eilutė arba stulpelis;
% Išėjimo parametrai
% x     - sistemos sprendinys,
% salsk - sistemos sąlygotumo skaičius.
% p     - požymis: jei p=1, tai sistema turi vieninteli sprendinį;
%                  jei p=0, tai sistema arba nesuderinta,
%                           arba turi be galo daug sprendiniu.

salsk=cond(a);
n=numel(b);
if size(b,1) == 1 % formuojame laisvųjų narių stulpelį
   b=b';
end;
a1=[a b]; % laisvųjų narių stulpelį prijungiame prie matricos a
x=zeros(1,n); % išskiriame sprendiniui vietą
% Tiesioginis etapas
p=1; k=1;
while (p ==1 ) & (k <= n-1)
  [d,l]=max(abs(a1(k:n,k)));
  l=l+k-1;
  if d <= eps
```

```
            p=0;
            disp('sistema nesuderinta arba turi daug sprendinių')
            return;
         else
           u=a1(k:n,k)./d; % normuojame vektorių u
           t=norm(u,2); % apskaičiuojame vektoriaus u ilgį
           a1(k,k)=-sign(a1(k,k))*t*d;% transformuojame k-jį stulpelį
% Vektoriaus omega apskaičiavimas
           u(1)=u(1)+sign(u(1))*t;
           t=norm(u,2);
           omega=u/t;
% Sandaugos (E-2*omega*omega')*a apskaičiavimas
 a1(k:n,k+1:n+1)=a1(k:n,k+1:n+1)-2*omega*(omega'*a1(k:n,k+1:n+1));
           end;
   k=k+1;
 end;
   % Atvirkštinis etapas
       b=a1(:,n+1);
       if abs(a1(n,n)) <= eps
            p=0;
             disp('sistema nesuderinta arba turi daug sprendinių')
             return;
          else
            for i = n:-1:1
                 x(i)=(b(i)-a1(i,i+1:n)*x(i+1:n)')/a1(i,i);
            end;
         end;
```

4.3.3. Skaidos metodai

LU metodas. Skaidos metodai pagrįsti tuo, kad lygčių sistemos $A\boldsymbol{x} = \boldsymbol{b}$ matrica A išskaidoma į dviejų trikampių matricų L ir U sandaugą: $A = L \cdot U$; čia L — apatinė trikampė matrica, U — viršutinė trikampė matrica.

Įrašę matricos A skaidinį į duotąją sistemą, turime tokią lygčių sistemą:

$LU\boldsymbol{x} = \boldsymbol{b}$.

Pažymėję $U\boldsymbol{x} = \boldsymbol{y}$, turime $L\boldsymbol{y} = \boldsymbol{b}$. Vadinasi, norėdami rasti sistemos $A\boldsymbol{x} = \boldsymbol{b}$ sprendinį, turime nuosekliai vieną po kitos išspręsti sistemas $L\boldsymbol{y} = \boldsymbol{b}$ ir $U\boldsymbol{x} = \boldsymbol{y}$ su trikampėmis matricomis

$$L = \begin{pmatrix} l_{11} & 0 & 0 & \cdots & 0 \\ l_{21} & l_{22} & 0 & \cdots & 0 \\ l_{31} & l_{32} & l_{33} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & l_{nn} \end{pmatrix}, \quad U = \begin{pmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & u_{22} & u_{23} & \cdots & u_{2n} \\ 0 & 0 & u_{33} & \cdots & u_{3n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & u_{nn} \end{pmatrix}.$$

Tarkime, kad $\Delta_1 = |a_{11}| \neq 0$, $\Delta_2 = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} \neq 0$, …, $\Delta_n = \det(A) \neq 0$. Tada matrica A gali būti išskaidyta į L ir U sandaugą, tačiau šis skaidinys nėra vienintelis. Kad jis būtų vienintelis, reikia fiksuoti arba matricos L, arba matricos U pagrindinės įstrižainės elementus.

Fiksuodami matricos L elementus $l_{ii} = 1\left(i = \overline{1,n}\right)$, matricas L ir U galime apskaičiuoti pritaikę Gauso metodo tiesioginį etapą.

Sudaryti skaidos metodų algoritmus bus patogiau, jei matricas L ir U padėsime matricos A vietoje. Kadangi $l_{ii} = 1\left(i = \overline{1,n}\right)$, tai išskaidę turėsime:

$$a_{ij} = \begin{cases} l_{ij}, i > j, \\ u_{ij}, i \le j. \end{cases}$$

Sistemos matricai A pritaikę Gauso metodo tiesioginį etapą, viršutinė trikampė matricos A dalis apibrėš matricą U. Matricą L apibrėš daugikliai $s = \frac{a_{ik}}{a_{kk}}$, t.y. $l_{ik} = \frac{a_{ik}}{a_{kk}}, k = \overline{1,n-1}, i = \overline{k+1,n}$. Aišku, kad l_{ik} reikšmės turi būti patalpintos elementų a_{ik} vietoje.

Tada sistemos $L\boldsymbol{y} = \boldsymbol{b}$ sprendinį galėsime apskaičiuoti pagal formulę $y_i = b_i - \sum_{j=1}^{i-1} a_{ij} \cdot y_j, i = \overline{1,n}$, o sistemos $U\boldsymbol{x} = \boldsymbol{y}$ sprendinį — pagal formulę $x_i = y_i - \sum_{j=i+1}^{n} a_{ij} \cdot x_j, i = n, n-1, \ldots, 1$.

Toliau pateikiame funkciją ***LUmet***, realizuojančią aprašytą metodą.

```
function [x,salsk,p] = LUmet(a,b)
 % LUMET funkcija sprendžia tiesinių lygčių sistemą LU metodu.
 %Įėjimo ir išėjimo parametrai sutampa su funkcijos Gausom
 %parametrais
 salsk=cond(a); n=length(b); [m,t]=size(b);
if m == 1 % formuojame laisvųjų narių stulpelį
     b=b';
 end;
x=zeros(1,n); y=zeros(1,n); % išskiriame sprendiniams vietą
 % Tiesioginis etapas
 p=1; k=1;
 while (p ==1 ) & (k <= n-1)
   [d,l]=max(abs(a(k:n,k)));
   l=l+k-1;
   if d <= eps
              p=0;
              disp('sistema nesuderinta arba turi daug sprendinių')
          else
              if l ~= k
                   z=a(k,:); a(k,:)=a(l,:); a(l,:)=z;
                   z=b(k); b(k)=b(l); b(l)=z;
               end;
                   s=a(k+1:n,k)/a(k,k);
                   a(k+1:n,k)=s;
                   a(k+1:n,k+1:n)=a(k+1:n,k+1:n)-s*a(k,k+1:n);
              end;
    k=k+1;
  end;
 % Atvirkštinis etapas
          if abs(a(n,n)) <= eps
              p=0;
              disp('sistema nesuderinta arba turi daug sprendinių')
```

```
else
     y(1)=b(1);
     for i = 2:n
         y(i)=b(i)-a(i,1:i-1)*(y(1:i-1))';
     end;
     for i = n:-1:1
          x(i)=(y(i)-a(i,i+1:n)*x(i+1:n)')/a(i,i);
     end;
 end;
```

Kaip matome, *LU* metodas yra vienas iš Gauso metodo variantų. Šis metodas yra pranašesnis už Gauso metodą, jei tenka spręsti keletą tiesinių lygčių sistemų $A\boldsymbol{x}_i = \boldsymbol{b}_i$, kurių matrica A ta pati, o laisvųjų narių $\boldsymbol{b}_i$ vektorius apskaičiuojamas tik išsprendus sistemas su kitais laisvaisiais nariais $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_{i-1}$. Tada, matricą A vieną kartą išskaidę į matricų L ir U sandaugą, toliau kiekvieną kartą spręstume sistemas: $L\boldsymbol{y} = \boldsymbol{b}$ ir $U\boldsymbol{x} = \boldsymbol{y}$.

Choleckio metodas. Kai lygčių sistemos matrica A yra teigiamai apibrėžta, taikomas kitas, dvigubai spartesnis, skaidos metodas, vadinamas Choleckio metodu.

Jei simetrinė matrica A yra teigiamai apibrėžta, tai ją galima išskaidyti į matricų L ir U sandaugą, kai $U = L^t$:

$$\begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} = \begin{pmatrix} l_{11} & 0 & \ldots & 0 \\ l_{21} & l_{22} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{pmatrix} \cdot \begin{pmatrix} l_{11} & l_{12} & \ldots & l_{1n} \\ 0 & l_{22} & \cdots & l_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & l_{nn} \end{pmatrix}.$$

Palyginę šios matricinės išraiškos dešiniosios ir kairiosios pusės atitinkamus elementus ir atsižvelgę į tai, kad matrica A yra simetrinė, turime:

$$a_{ij} = \sum_{k=1}^{i} l_{ik} \cdot l_{jk} = l_{ii} \cdot l_{ji} + \sum_{k=1}^{i-1} l_{ik} \cdot l_{jk} \text{ su visais } i = \overline{1,n},\ j = \overline{i,n}.$$

Jei matricas L ir U dedame matricos A vietoje, t. y. jei

$$a_{ij} = \begin{cases} l_{ij}, i \geq j, \\ u_{ij}, i \leq j \end{cases} \text{ (aišku, kad } l_{ii} = u_{ii},\ i = \overline{1,n}),$$

tai matricos A skaidinio elementus galime apskaičiuoti pagal formulę

$$a_{ij} = \begin{cases} \sqrt{a_{ii} - s}, & i = j, \\ \left(a_{ij} - s\right) / a_{ii}, & i \neq j, \end{cases} a_{ji} = a_{ij} \text{ su visais } i = \overline{1,n},\ j = \overline{i,n};$$

čia $s = \sum_{k=1}^{i-1} a_{ik} \cdot a_{jk}$.

Toliau pateikiame Choleckio metodą realizuojančią funkciją.

```
function [x,salsk] = cholmet(a,b)
% CHOLMET Choleckio metodą realizuojanti funkcija.
 % Įėjimo parametrai
 % a -sistemos matrica,
 % b - laisvųjų narių eilute arba stulpelis.
 % Išėjimo parametrai
 % x     - sistemos sprendinys,
 % salsk - sistemos sąlygotumo skaičius,
```

```
salsk=cond(a); n=length(b); [m,t]=size(b);
if m == 1
    b=b';
end;
y=zeros(1,n); x=zeros(1,n);
% Matricos a išskaidymas į L*U matricų sandaugą
for i=1:n
    for j=i:n
        s=sum(a(i,1:i-1).*a(j,1:i-1));
        if j == i
            a(i,j)=sqrt(a(i,i)-s);
        else
            a(i,j)=(a(i,j)-s)/a(i,i);
            a(j,i)=a(i,j);
        end
    end
end
% Sprendžiama lygtis L*y=b
y(1)=b(1)/a(1,1);
for i = 2:n
    y(i)=(b(i)-a(i,1:i-1)*(y(1:i-1))')/a(i,i);
end;
% Sprendžiama lygtis U*x=y
for i = n:-1:1
    x(i)=(y(i)-a(i,i+1:n)*x(i+1:n)')/a(i,i);
end;
```

4.3.4. Gauso ir Zeidelio bei paprastųjų iteracijų metodai

Kaip jau buvo minėta, labai didelis tiesinių lygčių sistemos sprendimo iteracinių metodų rinkinys pateiktas K. Čiegio ir V. Būdos vadovėlyje „Skaičiuojamoji matematika“. Todėl čia aptarsime tik pačius paprasčiausius apytikslius tiesinių lygčių sistemos sprendimo metodus: paprastųjų iteracijų bei Gauso ir Zeidelio metodus.

Tiesinių lygčių sistemą

$$a_{i1}x_1 + \ldots + a_{i,i-1}x_{i-1} + a_{ii}x_i + a_{i,i+1}x_{i+1} + \ldots + a_{in}x_n = b_i \ \left(i = \overline{1, n}\right)$$

perrašykime išreikšdami x_i iš *i*-tosios lygties, t. y.

$$x_i = \frac{1}{a_{ii}}\left(b_i - \sum_{\substack{j=1,\\ j\neq i}}^{n} a_{ij}x_j\right), \ i = \overline{1, n}.$$

Tarkime, kad $x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}$ yra pradinis duotosios lygčių sistemos sprendinys. Jeigu apie jį nieko nežinoma, galima daryti prielaidą, kad $x_i^{(0)} = 0 \ \left(i = \overline{1, n}\right)$. Tada paprastą iteracinį procesą bus galima nusakyti formule

$$x_i^{(k)} = \frac{1}{a_{ii}}\left(b_i - \sum_{\substack{j=1,\\ j\neq i}}^{n} a_{ij}x_j^{(k-1)}\right), \ i = \overline{1, n},$$

o Gauso ir Zeidelio iteracinį procesą — formule

$$x_i^{(k)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} \right), \ i = \overline{1, n}.$$

Gauso ir Zeidelio iteracinio proceso *k*-tojoje iteracijoje patikslintos x_i reikšmės naudojamos toje pačioje iteracijoje kitoms x_j $\left(j = \overline{i+1, n}\right)$ reikšmėms tikslinti. Todėl Gauso ir Zeidelio procesas konverguoja greičiau nei paprastas iteracinis procesas.

Kuriai sąlygai esant, šie iteraciniai procesai konverguoja? Yra žinoma, kad pakankamoji konvergavimo sąlyga yra tokia:

$$|a_{ii}| \geq \sum_{\substack{j=1,\\ j\neq i}}^{n} |a_{ij}| \text{ su visais } i = \overline{1, n} \quad \text{ir } |a_{ii}| > \sum_{\substack{j=1,\\ j\neq i}}^{n} |a_{ij}| \text{ bent su vienu } i.$$

Skaičiavimas paprastai baigiamas, kai $\max\limits_{1\leq i\leq n} \left| \dfrac{x_i^{(k)} - x_i^{(k-1)}}{x_i^{(k)}} \right| < \varepsilon$; čia ε — sprendinio tikslumas.

Pavyzdys. Paprastųjų iteracijų bei Gauso ir Zeidelio metodais išspręskime lygčių sistemą

$$\begin{cases} 2x + y = 2, \\ x - 2y = -2. \end{cases}$$

Gauso ir Zeidelio metodo darbo formulės yra tokios:

$$x^{(k)} = \frac{1}{2}\left(2 - y^{(k-1)}\right), \ y^{(k)} = \frac{1}{2}\left(2 + x^{(k)}\right),$$

o paprastųjų iteracijų metodo — tokios:

$$x^{(k)} = \frac{1}{2}\left(2 - y^{(k-1)}\right), \ y^{(k)} = \frac{1}{2}\left(2 + x^{(k-1)}\right).$$

Skaičiavimas parodytas 4.1 lentelėje.

4.1 lentelė. Lygčių sistemos $\begin{cases} 2x + y = 2, \\ x - 2y = -2 \end{cases}$ *sprendimas*

Gauso ir Zeidelio bei paprastųjų iteracijų metodais

	Gauso ir Zeidelio metodas		*Paprastųjų iteracijų metodas*	
k	*x*	*y*	*x*	*y*
0	0	0	0	0
1	1	3/2	1	1
2	1/4	9/8	1/2	3/2
3	7/16	39/32	1/4	5/4
4	25/64	153/128	3/8	9/8

Tikslus sistemos sprendinys yra *x* = 2/5, *y* = 6/5. Kaip matyti iš skaičiavimo rezultatų, Gauso ir Zeidelio metodas konverguoja greičiau nei paprastųjų iteracijų metodas.

Žemiau pateikta Gauso ir Zeidelio metodą realizuojanti funkcija ***gauszeid.***

```
function [x,it,p] = gauszeid(a,b,prec,itmax)
% GAUSZEID Gauso-Zeidelio metodu sprendžia tiesinių lygčių sistemą.
% Įėjimo parametrai
% a     - yra n-osios eiles kvadratinė matrica,
% b     - laisvųjų narių eilutė arba stulpelis,
% prec  - apskaičiuoto sprendinio tikslumas,
% itmax - maksimalus iteracijų skaičius.
% Išėjimo parametrai
% x     - sistemos sprendinys,
% it    - atliktų iteracijų skaičius,
% p     - požymis: jei p=1, tai sistema tenkina konvergavimo sąlygą;
%                  jei p=0, priešingu atveju.

n=length(b); [m,t]=size(b);
if m == 1
    b=b';
end;
% Ar tenkinama pakankama konvergavimo sąlyga?
p=1;
if any(sum(abs(a'))-2*abs(diag(a))'>0)
    p=0; disp('netenkinama konvergavimo sąlyga');
  x=nan; it=nan;
    return
end
t=diag(a);
a=a./repmat(t,1,n); b=b./t;
% Matricos a ir laisvųjų narių b paruošimas darbui
x=zeros(1,n)'; % pradinis sistemos sprendinys
it=0; ep=1+prec;
% Iteracijos
while (it <= itmax) & (ep > prec)
    x1=b-a*x+x;
    ep=norm((x1-x)./x1);
    x=x1; it=it+1;
 end
```

4.3.5. Pradinės informacijos paklaidos įtaka sprendinio tikslumui

Išsiaiškinkime praktiniu požiūriu labai svarbų klausimą — kokią įtaką matricos A ir laisvųjų narių vektoriaus b pradinės paklaidos turi sprendinio $\boldsymbol{x}$ tikslumui. Tarkime, kad matricos A elementai yra tikslūs, o laisvųjų narių paklaida lygi Δb. Dėl to atsiranda sprendinio paklaida $\Delta \boldsymbol{x}$, t. y.

$$A(\mathbf{x}+\Delta\mathbf{x}) = b + \Delta b.$$

Iš čia $A \cdot \Delta\mathbf{x} = \Delta b$.

Tada

$$\|\Delta\mathbf{x}\| = \left\|A^{-1}\Delta b\right\| \le \left\|A^{-1}\right\| \cdot \|\Delta b\|, \quad \|b\| = \|A\mathbf{x}\| \le \|A\| \cdot \|\mathbf{x}\|.$$

Sudauginę šias nelygybes, turime: $\|\Delta\mathbf{x}\| \cdot \|b\| \le \|A\| \cdot \left\|A^{-1}\right\| \cdot \|\Delta b\| \cdot \|\mathbf{x}\|$.

Taigi

$$\frac{\|\Delta \mathbf{x}\|}{\|\mathbf{x}\|} \le \|A\| \cdot \|A^{-1}\| \frac{\|\Delta b\|}{\|b\|}.$$

Daugiklis $\|A\| \cdot \|A^{-1}\|$ vadinamas lygčių sistemos ***sąlygotumo skaičiumi*** ir žymimas ***cond***(***A***). Jis rodo, kiek kartų sprendinio paklaida yra didesnė už lygčių sistemos pradinės informacijos paklaidą.

Rezultatas bus analogiškas, jei laikysime, kad laisvieji nariai yra tikslūs, o matricos A elementai turi paklaidą ΔA. Šiuo atveju turėsime lygybę: $(A + \Delta A)(\mathbf{x} + \Delta \mathbf{x}) = b.$

Iš jos išplaukia, kad: $\mathbf{x} + \Delta \mathbf{x} = (A + \Delta A)^{-1} b.$

Kadangi $\mathbf{x} = A^{-1} \cdot b$, tai $\Delta \mathbf{x} = (A + \Delta A)^{-1} b - A^{-1} b = \left((A + \Delta A)^{-1} - A^{-1}\right) b.$

Toliau nagrinėdami, remsimės tokia matricų savybe:

$B^{-1} - A^{-1} = A^{-1}(A - B)B^{-1}$.

Tada $\Delta \mathbf{x} = A^{-1}(A - A - \Delta A)(A + \Delta A)^{-1} b$, arba $\Delta \boldsymbol{x} = -A^{-1} \Delta A \left(\boldsymbol{x} + \Delta \boldsymbol{x}\right)$

ir

$$\|\Delta \boldsymbol{x}\| \le \|A^{-1}\| \cdot \|\Delta A\| \cdot \|\boldsymbol{x} + \Delta \boldsymbol{x}\|.$$

Iš čia $\frac{\|\Delta \boldsymbol{x}\|}{\|\boldsymbol{x} + \Delta \boldsymbol{x}\|} \le \|A\| \cdot \|A^{-1}\| \frac{\|\Delta A\|}{\|A\|}$.

Iš pateikto nagrinėjimo matyti, kad sprendinio paklaida priklauso nuo *cond*(*A*). Sistemos, kurių *cond*(*A*) yra didelis, vadinamos ***blogai sąlygotomis***. Tokių sistemų sprendiniai, nesvarbu, kuriuo metodu rasti, turi dideles paklaidas. Todėl labai svarbu apskaičiuoti arba įvertinti lygčių sistemos sąlygotumo skaičių.

Pavyzdžiui, jei sistemos sąlygotumo skaičius yra 10^6 eilės, o pradinės informacijos - maticos *A* ir vektoriaus *b* elementų - paklaidos yra 10^{-8} eilės, tai apskaičiuoto sprendinio *x* paklaida bus 10^{-2} eilės. Vadinasi, sistemos ***sąlygotumo skaičius*** yra labai svarbus parametras, nusakantis sprendinio tikslumą.

MATLAB‘as turi sąlygotumo skaičių apskaičiuojančią funkciją ***cond***.

4.3.5. Perkelties metodas

Sprendžiant praktinius uždavinius, pavyzdžiui, apskaičiuojant interpoliacinius kubinius splainus, tenka ieškoti tiesinių lygčių sistemų su triįstrižaine (tridiagonaliąja) matrica *A* sprendinių. Tokios matricos visi elementai, išskyrus trijų pagrindinių įstrižainių elementus, lygūs nuliui. Pavyzdžiui, matrica

$$A = \begin{pmatrix} * & * & 0 & 0 & 0 \\ * & * & * & 0 & 0 \\ 0 & * & * & * & 0 \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$

yra triįstrižainė penktosios eilės matrica.

4.3.1 paragrafe pateikto Gauso metodo apimtis yra $O(n^3)$. Kai matrica *A* yra triįstrižainė, Gauso metodą galima modifikuoti taip, kad jo apimtis būtų $O(n)$. Toks metodas vadinamas ***perkelties metodu***.

Gauso metodu, nekeisdami lygčių tvarkos, išspręskime tiesinę lygčių sistemą su triįstrižaine matrica

$$a_i x_{i-1} - b_i x_i + c_i x_{i+1} = d_i; \qquad (4.1)$$

čia $i = \overline{1, n}$, $a_1 = c_n = 0$. Po tiesioginio sprendimo etapo turėsime sistemą

$$x_i = s_{i+1} x_{i+1} + e_{i+1} \quad \left(i = \overline{1, n}\right). \qquad (4.2)$$

(4.2) formulę, sumažinę jos indeksą i vienetu, įrašykime į (4.1) lygtį:

$$a_i\left(s_i x_i + e_i\right) - b_i x_i + c_i x_{i+1} = d_i \quad \left(i = \overline{1, n}\right).$$

Išreiškę x_i, turime:

$$x_i = \frac{c_i}{b_i - a_i s_i} x_{i+1} + \frac{a_i e_i - d_i}{b_i - a_i s_i} \quad \left(i = \overline{1, n}\right). \qquad (4.3)$$

Kad (4.2) formulė sutaptų su (4.3), turi būti

$$s_{i+1} = \frac{c_i}{b_i - a_i s_i}, \quad e_{i+1} = \frac{a_i e_i - d_i}{b_i - a_i s_i} \quad \left(i = \overline{1, n}\right). \qquad (4.4)$$

Kadangi $a_1 = 0$, tai s_1 ir e_1 gali būti bet kokie. Imame $s_1 = e_1 = 0$. Tuomet perkelties metodą galėsime apibūdinti toliau pateikiamomis formulėmis.

Tiesioginis etapas

$$s_{i+1} = c_i / \left(b_i - a_i s_i\right),$$

$$e_{i+1} = \left(a_i e_i - d_i\right) / \left(b_i - a_i s_i\right), \quad i = \overline{1, n},$$

$$s_1 = e_1 = 0.$$

Atvirkštinis etapas

$$x_i = s_{i+1} x_{i+1} + e_{i+1}, \quad i = \overline{n, 1},$$

$$x_{n+1} = 0, \text{ nes } c_n = 0.$$

Šios formulės nusako Gauso metodą, kurį taikant lygčių tvarka nekeičiama. Kyla klausimas: ar nebus dalybos iš nulio? Sąlyga $|b_i| \geq |a_i| + |c_i|$ su visais $i = \overline{1, n}$ ir $|b_i| > |a_i| + |c_i|$ nors su vienu i yra pakankama, kad, sprendžiant lygčių sistemą, nebūtų dalybos iš nulio.

Pavyzdys. Perkelties metodu išspręskime lygčių sistemą

$$\begin{cases} x_1 + 2x_2 = 3, \\ x_1 - 3x_2 + x_3 = -3, \\ x_2 + 4x_3 - x_4 = -2, \\ x_3 + x_4 = -2. \end{cases}$$

Jos koeficientų A, B, C ir D masyvai yra tokie: $A = (0, 1, 1, 1)$, $B = (-1, 3, -4, -1)$, $C = (2, 1, -1, 0)$, $D = (3, -3, -2, -2)$.

Tiesioginio etapo skaičiavimai pateikti 4.2 lentelėje.

4.2 lentelė. Lygčių sistemos sprendimas perkelties metodu

i	s_i	e_i	$b_i - a_i s_i$	$a_i e_i - d_i$
1	0	0	−1	−3
2	−2	3	5	6

3	1/5	6/5	–21/5	16/5
4	5/21	–16/21	–26/21	26/21
5	0	–1	–	–

Iš atvirkštinio etapo formulių apskaičiuojame:

$$x_4 = -1;\ x_3 = \frac{5}{21}(-1) + \left(-\frac{16}{21}\right) = -1;\ x_2 = \frac{1}{5}(-1) + \frac{6}{5} = 1;\ x_1 = -2 \cdot 1 + 3 = 1.$$

Žemiau pateikta perkelties metodą realizuojanti funkcija ***perkelt***.

```
function x=perkelt(a,b,c,d)
% PERKELT sprendžia triįstrižainininę tiesinių lygčių sistemą.
% Įėjimo parametrai
% a -  sistemos šalutinė (apatinė) įstrižainė,
% b -  sistemos pagrindinė įstrižainė,
% c -  sistemos šalutinė (viršutinė) įstrižainė,
% d - laisvųjų narių masyvas.
% Išėjimo parametrai
% x - sistemos sprendinys

% Tiesioginis etapas
n=numel(b); s(1)=0; e(1)=0;
for i=1:n
    t=b(i)-a(i)*s(i);
    s(i+1)=c(i)/t;
    e(i+1)=(a(i)*e(i)-d(i))/t;
end
% Atvirkstinis etapas
x(n)=e(n+1);
for i=n-1:-1:1
    x(i)=s(i+1)*x(i+1)+e(i+1);
end
```

4.3.6. MATLAB'o tiesinių lygčių sistemos sprendimo funkcijos

Apie MATLAB'o funkcijas, skirtas tiesinės algebros uždaviniams spręsti, galima sužinoti surinkus komandą „***help matfun***". Čia aptarsime dvi tiesinių lygčių sistemos sprendimo MATLAB'o funkcijas: „\" (***backslash or left division***) ir „***linsolve***".

Funkcija „\" (***backslash or left division***).

Norint išspręsti tiesinę lygčių sistemą $Ax = b$, čia A – kvadratinė matrica, reikia parašyti komandą: $x = A \setminus b$. Sprendinys bus apskaičiuojamas Gauso metodu.

Jei matrica A yra blogai sąlygota, tai apie šį faktą bus atspausdintas įspėjimas. Daugiau informacijos apie nagrinėjamą funkciją galima gauti surinkus komandą: „***help slash***".

Pavyzdys. Pasinaudodami funkcija „\" (***backslash***), išspręskime 4.3.1 paragrafo 1-ą ir 2-ą pavyzdžius.

1-o pavyzdžio sprendimas.

```
a=[1 -2 3;-1 1 3;2 -1 1];
>> b=[2 3 2 ]';
>> x=a\b
x =  1
     1
     1
```

2-o pavyzdžio sprendimas.

```
>> a=[1 -3 3;-1 2 3;1 -4 9];
>> b=[1 4 6]';
```

```
>> x=a\b
Warning: Matrix is singular to working precision.
x =
   NaN
   NaN
   NaN
```

Funkcija linsolve.

Ši MATLAB'o funkcija taip pat yra skirta spręsti tiesinių lygčių sistemą $Ax = b$. Pagrindinis kreipinys į šią funkciją yra: ***x=linsolve(A,b)***.

Jei matrica A yra kvadratinė, tai sistema sprendžiama ***LU*** metodu; priešingu atveju naudojama ***QR*** faktorizacija. Yra atsiunčiamas pranešimas, jei matrica A yra kvadratinė ir blogai sąlygota arba išsigimusi. Daugiau informacijos apie funkciją galima gauti surinkus komandą: „***help linsolve***".

Pavyzdys. Pasinaudodami funkcija ***linsolve***, išspręskime 4.3.1 paragrafo 1-ą ir 2-ą pavyzdžius.

1-o pavyzdžio sprendimas.

```
a=[1 -2 3;-1 1 3;2 -1 1];
>> b=[2 3 2 ]';
>> x=linsolve(a,b)
x =  1
     1
     1
```

2-o pavyzdžio sprendimas.

```
>> a=[1 -3 3;-1 2 3;1 -4 9];
>> b=[1 4 6]';
>> x=linsolve(a,b)
Warning: Matrix is singular to working precision.
x =
   NaN
   NaN
   NaN
```

4.4. Determinanto apskaičiavimas

Tarkime, kad $a_1, a_2, \ldots, a_n$ yra aibės $\{1, 2, \ldots, n\}$ kėlinys. Jei $i < j$ ir $a_i > a_j$, tai pora (a_i, a_j) vadinama kėlinio $a_1, a_2, \ldots, a_n$ inversija. Pavyzdžiui, kėlinys 3142 turi 3 inversijas: (3, 1), (3, 2) ir (4, 2).

Kiekviena inversija — tai pora, „sujaukianti tvarką". Vadinasi, vienintelis kėlinys, neturintis inversijų, yra $1, 2, \ldots, n$.

Inversijos sąvoką 1750 m. pirmasis pavartojo Šveicarijos matematikas G. Krameris (Kramer), aprašydamas tiesinių lygčių sistemos sprendimo metodą, kuris dabar vadinamas jo vardu.

G. Krameris *n*-tosios eilės determinantą apibrėžė taip:

$$\det\begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix} = \sum (-1)^{I(k_1, k_2, \ldots, k_n)} \cdot a_{1k_1} a_{2k_2} \ldots a_{nk_n};$$

čia sudedama pagal visus aibės $\{1, 2, \ldots, n\}$ kėlinius, o $I(k_1, k_2, \ldots, k_n)$ yra kėlinio $k_1, k_2, \ldots, k_n$ inversijų skaičius. Pavyzdžiui,

$$\det\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = (-1)^{I(1,2,3)} a_{11}a_{22}a_{33} + (-1)^{I(2,1,3)} a_{12}a_{21}a_{33} +$$

$$+ (-1)^{I(2,3,1)} a_{12}a_{23}a_{31} + (-1)^{I(3,1,2)} a_{13}a_{21}a_{32} +$$

$$+ (-1)^{I(3,2,1)} a_{13}a_{22}a_{31} + (-1)^{I(1,3,2)} a_{11}a_{23}a_{32} =$$

$$= (-1)^0 a_{11}a_{22}a_{33} + (-1)^1 a_{12}a_{21}a_{33} + (-1)^2 a_{12}a_{23}a_{31} +$$

$$+ (-1)^2 a_{13}a_{21}a_{32} + (-1)^3 a_{13}a_{22}a_{31} + (-1)^1 a_{11}a_{23}a_{32}.$$

Antros ir trečios eilės determinantai paprastai skaičiuojami pagal Kramerio pateiktą determinanto apibrėžimą.

Apskaičiuoti aukštesnės eilės determinantą pagal apibrėžimą labai neracionalu, nes reikia atlikti $n!$ $a_{1k_1} \ldots a_{nk_n}$ daugybos ir tiek pat sudėties veiksmų.

Tam, kad apskaičiuotume aukštesnės eilės determinantą, pasinaudosime determinanto savybėmis.

Yra žinoma, kad

1. determinanto reikšmė nepakinta, jei iš kurios nors determinanto eilutės atimsime kitą eilutę, padauginta iš kokio nors skaičiaus;
2. determinantas keičia ženklą, jei sukeisime jo kokias nors dvi eilutes vietomis;
3. trikampės matricos determinantas yra lygus pagrindinės įstrižainės elementų sandaugai.

Vadinasi, determinantas skaičiuojamas paprasčiau, kai, taikant Gauso metodo tiesioginį etapą, determinantui suteikiama trikampė išraiška. Tada pradinio determinanto reikšmė lygi perskaičiuoto determinanto reikšmei, o ši — pagrindinės įstrižainės elementų sandaugai.

Perskaičiuojant determinantą, reikia atsižvelgti į tai, kad:

1) keičiant eilutes vietomis, keičiasi determinanto ženklas;

2) jei $\max\limits_{k \le i \le n} (a_{ik}) = 0,$ tai determinantas lygus nuliui.

Determinanto ženklo įvertinimą, kečiant eilutes vietomis, patogu atlikti taip:

1) pradžioje kintamajam *D* suteikiame reikšmę 1;
2) keičiant *k*-ąją ir *l*-ąją eilutes vietomis, operatoriumi $D = -D$ invertuojame kintamojo *D* turinį. Vadinasi, jei eilučių sukeitimų skaičius bus lyginis, tai *D* turinys bus lygus 1; o jei eilučių sukeitimų skaičius bus nelyginis, tai *D* turinys bus lygus -1. Abiem atvejais *D* turinys įvertins determinanto ženklą, kurį iššaukia determinanto eilučių sukeitimas vietomis.

Pavyzdys. *Apskaičiuokime determinantą:*

$$D=\begin{vmatrix} 1 & 3 & 0 & -1 \\ 2 & 7 & 4 & -3 \\ -1 & -4 & 2 & -3 \\ 0 & 2 & 4 & 1 \end{vmatrix}.$$

Pirmasis žingsnis

Iš antrosios eilutės atimsime pirmąją eilutę padaugintą iš 2. Iš trečiosios eilutės atimsime pirmąją eilutę padaugintą iš(-1). Turėsime:

$$D=\begin{vmatrix} 1 & 3 & 0 & -1 \\ 0 & 1 & 4 & -1 \\ 0 & -1 & 2 & -4 \\ 0 & 2 & 4 & 1 \end{vmatrix}.$$

Antrasis žingsnis

Iš trečiosios eilutės atimsime antrąją eilutę padaugintą iš (-1). Iš ketvirtosios eilutės atimsime antrąją eilutę padaugintą iš 2. Turėsime:

$$D=\begin{vmatrix} 1 & 3 & 0 & -1 \\ 0 & 1 & 4 & -1 \\ 0 & 0 & 6 & -5 \\ 0 & 0 & -4 & 3 \end{vmatrix}$$

Trečiasis žingsnis

Iš ketvirtosios eilutės atimsime trečiąją eilutę padaugintą iš 2/3. Turėsime:

$$D=\begin{vmatrix} 1 & 3 & 0 & -1 \\ 0 & 1 & 4 & -1 \\ 0 & 0 & 6 & -5 \\ 0 & 0 & 0 & -\frac{1}{3} \end{vmatrix}.$$

Baigiamasis žingsnis

Kadangi skaičiuojant eilutės tarpusavyje nebuvo keičiamos, tai determinantas yra lygus pagrindinės įstrižainės elementų sandaugai. Vadinasi, $D = 1*1*6*\frac{-1}{3} = -2$.

Čia nepateikiame determinanto apskaičiavimo funkcijos, nes ją lengva parašyti, modifikuojant tiesinių lygčių sistemos sprendimo Gauso metodu tiesioginį etapą. Be to, MATLAB‘as turi vidinę determinanto apskaičiavimo funkciją ***det***, kuri ir apskaičiuoja determinantą Gauso metodu. Kreipinys į funkciją yra: ***d=det(a)***, čia ***a*** – matrica.

Pavyzdys. *Pasinaudodami funkcija* ***det***, *apskaičiuokime tą patį determinantą:*

$$D=\begin{vmatrix} 1 & 3 & 0 & -1 \\ 2 & 7 & 4 & -3 \\ -1 & -4 & 2 & -3 \\ 0 & 2 & 4 & 1 \end{vmatrix}.$$

```
>> a=[1 3 0 -1;2 7 4 -3;-1 -4 2 -3;0 2 4 1];
>> d=det(a)
d = -2
```

4.5. Atvirkštinės matricos apskaičiavimo metodai

Tarkime, kad A yra n-tosios eilės neišsigimusioji kvadratinė matrica. Tada matricos A atvirkštinė matrica yra tokia matrica A^{-1}, su kuria $A \cdot A^{-1} = A^{-1} \cdot A = E$; čia E — vienetinė n-tosios eilės matrica.

Yra žinoma analizinė matricos A^{-1} išraiška:

$$A^{-1} = \frac{1}{\det(A)} \begin{pmatrix} A_{11} & \dots & A_{i1} & \dots & A_{n1} \\ \dots & \dots & \dots & \dots & \dots \\ A_{1j} & \dots & A_{ij} & \dots & A_{nj} \\ \dots & \dots & \dots & \dots & \dots \\ A_{1n} & \dots & A_{in} & \dots & A_{nn} \end{pmatrix}; \qquad (4.6)$$

čia A_{ij} — matricos A elemento a_{ij} adjunktas.

Skaičiuoti atvirkštinę matricą pagal (4.6) formulę neracionalu, nes reikia rasti $n^2 + 1$ determinantą. Determinanto apskaičiavimo algoritmo apimtis yra $O(n^3)$, vadinasi, atvirkštinės matricos apskaičiavimo pagal (4.6) formulę algoritmo apimtis bus $O(n^5)$.

Išnagrinėkime kitus atvirkštinės matricos apskaičiavimo metodus, kurių apimtis $O(n^3)$. Metodo idėją aptarsime imdami antrosios eilės matricą A.

Tarkime, kad matricos $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ atvirkštinė matrica yra $X = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix}$. Tada $AX = E$.

Išskleista ši lygybė atrodys taip:

$$\begin{pmatrix} a_{11}x_{11} + a_{12}x_{21} & a_{11}x_{12} + a_{12}x_{22} \\ a_{21}x_{11} + a_{22}x_{21} & a_{21}x_{12} + a_{22}x_{22} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Matricos yra lygios, kai jų atitinkami elementai yra lygūs.
Vadinasi,

$$\begin{cases} a_{11}x_{11} + a_{12}x_{21} = 1, \\ a_{21}x_{11} + a_{22}x_{21} = 0 \end{cases}$$

ir

$$\begin{cases} a_{11}x_{12} + a_{12}x_{22} = 0, \\ a_{21}x_{12} + a_{22}x_{22} = 1. \end{cases}$$

Taigi, norint rasti n-tosios eilės kvadratinės matricos A atvirkštinę matricą, reikia išspręsti n tiesinių lygčių sistemų, kurių koeficientų prie nežinomųjų matricos lygios matricai A, o laisvieji nariai yra vienetinės matricos stulpeliai. Tokias lygčių sistemas Gauso, QR, LU ir kt. metodai įgalina spręsti vienu metu. Jei lygčių sistemos sprendžiamos Gauso metodu, tai tas atvirkštinės matricos skaičiavimo metodas vadinamas Gauso metodu.

Atvirkštinės matricos apskaičiavimas Gauso metodu

Duota lygčių sistema

$AX = E$; (4.7)

čia A ir X — n-tosios eilės kvadratinės matricos, o E — n-tosios eilės vienetinė matrica. Šią lygčių sistemą spręsime Gauso metodu.

(4.7) lygčių sistema skiriasi nuo anksčiau nagrinėtos tiesinių lygčių sistemos tuo, kad turi ne vieną, o n laisvųjų narių stulpelių. Vadinasi, Gauso metodo tiesioginiame etape elementarius pertvarkius reikės atlikti ne su vienu laisvųjų narių stulpeliu, o su visais stulpeliais. Tuo pačiu, atvirkštiniame etape, su kiekvienu laisvųjų narių stulpeliu reikės išspręsti sistema, kurios sprendinys ir bus atitinkamas atvirkštinės matricos stulpelis.

Pavyzdys. *Gauso metodu apskaičiuokime matricos* $A = \begin{pmatrix} 1 & 0 & 1 \\ 5 & -1 & 4 \\ 4 & 1 & 4 \end{pmatrix}$ *atvirkštinę matricą.*

Tiesioginis etapas

$$\begin{pmatrix} 1 & 0 & 1 \\ 5 & -1 & 4 \\ 4 & 1 & 4 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}; \begin{pmatrix} 1 & 0 & 1 \\ 0 & -1 & -1 \\ 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ -5 & 1 & 0 \\ -4 & 0 & 1 \end{pmatrix}; \begin{pmatrix} 1 & 0 & 1 \\ 0 & -1 & -1 \\ 0 & 0 & -1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ -5 & 1 & 0 \\ -9 & 1 & 1 \end{pmatrix}.$$

Atvirkštinis etapas

Pastaba. *Raidė „l“ žymi laisvųjų narių stulpelio numerį.*

Kai $l = 1$, sprendžiame sistemą $\begin{pmatrix} 1 & 0 & 1 \\ 0 & -1 & -1 \\ 0 & 0 & -1 \end{pmatrix}\begin{pmatrix} 1 \\ -5 \\ -9 \end{pmatrix}$ ir turime pirmąjį A^{-1} stulpelį $\begin{pmatrix} -8 \\ -4 \\ 9 \end{pmatrix}$.

Kai $l = 2$, sprendžiame sistemą $\begin{pmatrix} 1 & 0 & 1 \\ 0 & -1 & -1 \\ 0 & 0 & -1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$ ir turime antrąjį A^{-1} stulpelį $\begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$.

Kai $l = 3$, išsprendę sistemą $\begin{pmatrix} 1 & 0 & 1 \\ 0 & -1 & -1 \\ 0 & 0 & -1 \end{pmatrix}\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$, turime trečiąjį A^{-1} stulpelį $\begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix}$.

Taigi matricos A atvirkštinė matrica yra tokia: $A^{-1} = \begin{pmatrix} -8 & 1 & 1 \\ -4 & 0 & 1 \\ 9 & -1 & -1 \end{pmatrix}$.

Žemiau pateikta funkcija ***atvmat***, apskaičiuojanti atvirkštinę matricą Gauso metodu.

```
function [x,p] = atvmat(a)
% ATVMAT funkcija apskaičiuoja atvirkštinę matricą Gauso metodu.
% Įėjimo parametrai
% a - kvadratine matrica
% Išėjimo parametrai
% x - matricos a atvirkštinė matrica
% p - požymis: jei p=1, tai matrica a - neišsigimusi,
%              jei p=0, tai matrica a - išsigimusi,

n,t]=size(a); x=zeros(n); b=eye(n);
% Tiesioginis etapas
p=1; k=1;
while (p ==1 ) & (k <= n-1)
  [d,l]=max(abs(a(k:n,k)));
```

```
l=l+k-1;
if d <= eps
           p=0;
           disp('matrica išsigimusi')
           return
       else
           if l ~= k
                z=a(k,:); a(k,:)=a(l,:); a(l,:)=z;
                z=b(k,:); b(k,:)=b(l,:); b(l,:)=z;
           end;
                s=a(k+1:n,k)/a(k,k);
                a(k+1:n,k)=0;
                a(k+1:n,k+1:n)=a(k+1:n,k+1:n)-s*a(k,k+1:n);
                b(k+1:n,:)=b(k+1:n,:)-s*b(k,:);
           end;
 k=k+1;
end;
%  Atvirkštinis etapas
       if abs(a(n,n)) <= eps
           p=0;
           disp('matrica išsigimusi')
           return
        else
           for i = n:-1:1
                x(i,:)=(b(i,:)-a(i,i+1:n)*x(i+1:n,:))/a(i,i);
           end;
        end;
```

MATLAB‘as turi vidinę atvirkštinės matricos apskaičiavimo funkciją ***inv***.

Pavyzdys. Pasinaudodami funkcija ***inv***, apskaičiuokime ankstesnio pavyzdžio atvirkštinę matricą.

```
>> a=[1 0 1;5 -1 4;4 1 4]
a =
     1     0     1
     5    -1     4
     4     1     4
>> x=inv(a)
x =
   -8.0000    1.0000    1.0000
   -4.0000         0    1.0000
    9.0000   -1.0000   -1.0000
```

Pastaba. Nors funkcija ***inv*** gali būti naudojama sprendžiant tiesinių lygčių sistemą $Ax = b$, surenkant komandą $x = inv(A)b$, tačiau sistema bus sprendžiama tiksliau ir greičiau taikant funkciją **„\“** , t.y. surinkus komandą $x = A \setminus b$.