2. *MATLAB'o pradmenys*

2.1. MATLAB'as inžinerinių skaičiavimų kalba

Programų sistemos MATLAB pavadinimas yra angliškų žodžių "***mat**rix* ***lab**oratory*" (matricų laboratorija) santrumpa. MATLAB'as tai interaktyvi sistema, skirta modeliuoti ir spręsti inžinerinius uždavinius, o taip pat grafiškai pavaizduoti skaičiavimo rezultatus. Kaip matyti iš sistemos pavadinimo, ji skirta skaičiavimams su matricomis. Vartotojas, mokantis programuoti kuria nors aukšto lygio programavimo kalba, kaip FORTRAN, BASIC, PASCAL, C ir pan., turi atkreipti dėmesį į tai, kad MATLAB'e visi duomenys ir rezultatai yra matricos. Pavyzdžiui, skaičiai suprantami kaip (1×1) formato matricos. Todėl, jei anksčiau paminėtose programavimo kalbose, pavyzdžiui, operatorius *a=abs(b)* suprantamas, kaip "*kintamasis a įgauna kintamojo b modulio reikšmę*", tai MATLAB'e, jei *b* yra $(m\times n)$ formato matrica, jis reikštų, kad "*a bus* $(m\times n)$ *formato matrica, sudaryta iš matricos b elementų modulių".*

MATLAB'as vartotojui leidžia:

1) lengvai atlikti skaičiavimus su matricomis bei jų struktūromis;
2) pasinaudoti įsiūtomis funkcijomis, kurių yra apie 300-us, o taip pat įvairiomis vidinėmis procedūromis, kurių yra virš 1000-io;
3) vaizdžiai pavaizduoti skaičiavimo rezultatus, taikant "galingas" dviejų ir trijų dimensijų grafines procedūras;
4) remiantis įvairiais programinių priemonių komplektais (toolbox): *wavelet, pde, spline, signal processing* ir pan., spręsti specifinius inžinerijos uždavinius;
5) naudojant MATLAB'o aukšto lygio programavimo kalbą, pačiam sukurti norimas procedūras, kurios gali būti naudojamos tolesniuose skaičiavimuose.

Programavimas MATLAB'e iš esmės nesiskiria nuo programavimo kitose aukšto lygio programavimo kalbose. Tačiau, norint efektyviai pasinaudoti visomis MATLAB'o teikiamomis galimybėmis, reikia gerai suprasti įsiūtas ir vidines jo procedūras, o taip pat MATLAB'o operatorių semantiką. Pavyzdžiui, norint rasti didžiausią masyvo *a* elementą *x* tarp elementų, didesnių už 2 ir mažesnių už 10, MATLAB'e tereikia vienos komandos:

>> *x =max* (*a*((*a*>2) & (*a*<10))).

Tuo tarpu kitose aukšto lygio programavimo kalbose tam, kad rastume tokį elementą, reikėtų visos eilės komandų.

2.2. Išraiškų skaičiavimas su MATLAB'u

MATLAB'ą paleidus veikti, yra atidaromas komandų langas, per kurį galima bendrauti su MATLAB'u. Reikalui esant, MATLAB'as gali atidaryti ir kitus langus: grafinį, redagavimo, pagalbos ir pan.

Komandų lango kairėje pusėje pasirodo „varnelės" (">>"), jo dešinėje stovi žymeklis („|"), o lango apačioje pasirodo užrašas "*Ready*" (pasiruošęs). Tai reiškia, kad MATLAB'as pasiruošęs priimti vartotojo komandas.

Pastaba. Tolesniame dėstyme, nagrinėjant pavyzdžius, eilutė, kurios kairėje pusėje stovi „varnelės" (">>"), yra suprantama kaip MATLAB'o komanda.

Pradžioje aptarsime kaip MATLAB'u ***apskaičiuoti aritmetines ir algebrines išraiškas***. Tam tikslui, kaip ir bet kurioje aukšto lygio programavimo kalboje, pirmiausia aptarsime ***kintamuosius, operacijas, elementariąsias funkcijas*** ir kitus klausimus, susijusius su išraiškų apskaičiavimu.

MATLAB'o kintamieji. Kaip ir kitos aukšto lygio programavimo kalbos, MATLAB'as turi analogiškas kintamųjų vardų sudarymo taisykles, kurios surašytos 2.1-oje lentelėje.

2.1 *lentelė. MATLAB'o kintamųjų vardų sudarymo taisyklės*

Vardų sudarymo taisyklės	*Komentarai ir pavyzdžiai*
Kintamojo vardas - tai raidžių, skaičių ir pabraukimų (underscores) rinkinys, prasidedantis raide. Kiti skyrybos ženklai: taškai , kableliai, kabliataškiai rinkiniui nepriklauso. MATLAB'e jie turi specialias reikšmes.	*a1, kas_tai, aij, matricos_elementas, b13c, a_b_c,* - tai MATLAB'o kintamųjų pavyzdžiai.
Kintamojo vardą turi sudaryti ne daugiau kaip 31 simbolis.	Vardo pertekliniai simboliai yra ignoruojami.
MATLAB'as skiria vardo mažąsias ir didžiąsias raides	*Cost, cost, COSt ir COST* yra skirtingi vardai.
Sudarant vardus yra kelios specifinės išimtys – tai MATLAB'o raktiniai žodžiai arba rezervuotų žodžių sąrašas	Rezervuotų žodžių sąrašas yra: *for, end, if else, elseif, while, function, return, case, otherwise, switch, continue, try, catch, global, persistent, break.* MATLAB'as perspės, jei kintamojo vardas sutaps su vienu iš tų žodžių. Tačiau, jei šiuose žodžiuose bent vieną raidę pakeisite didžiąja raide, tai šis žodis reikš teisingą kintamojo vardą.

MATLAB'as turi eilę rezervuotų vidinių kintamųjų, skirtų įvairioms konstantoms žymėti. Šie kintamieji surašyti 2.2 lentelėje. Šių kintamųjų rezervavimas nėra griežtas: juos galima naudoti suteikiant jiems kitas reikšmes, tačiau tada prarandama MATLAB'o suteikta reikšmė. Pavyzdžiui, kintamasis *pi* MATLAB'e reiškia skaičių π = 3.14159265....- apskritimo ilgio ir jo skersmens santykį. Tačiau, jei *pi* priskirsime reikšmę *2*, t.y. parašysime komandą *pi* = 2*,* tai tolesniuose skaičiavimuose *pi* jau reikš 2.

Komandos who, whos ir clear. Programuojant kartais pasimiršta atmintyje (darbo lauke - *workspace)* saugomų kintamųjų vardai.

Kintamųjų vardus ekrane pateikia komanda ***who***.

Jei domina ir kintamųjų dydis naudojama komanda ***whos***.

Numatytasis MATLAB'o rėžimas,- visų kintamųjų reikšmės saugomos iki darbo pabaigos.

Kintamasis iš atminties (darbo lauko) pašalinamas komanda:

***clear kint_vardas*.**

Nenurodžius argumento, komanda *clear* iš atminties (darbo lauko) pašalina visus kintamuosius.

2.2 *lentelė. Specialūs MATLAB'o kintamieji*

Kintamieji	*Paaiškinimai*
ans	Vardas rezultato, kai vartotojas jo nenurodo. Pvz,. parašius komandiniame lange komandą `>> 3+5` ir paspaudus klavišą "*Enter*", lange pasirodo atsakymas: `ans = 8`
pi	Skaičius π
eps	Mažiausias skaičius tenkinantis nelygybę 1+*eps*>1, *eps*=$2.220446049250313 \cdot 10^{-16}$.
inf	Žymi begalybę, t.y. rezultatą 1/0.
NaN arba *nan*	Žymi neapibrėžtumą 0/0. Kintamojo vardas kilęs nuo angliškų žodžių "*Not-a-Number*" (nėra skaičiaus).
i arba *j*	*i* arba *j* žymi menamą vienetą — $\sqrt{-1}$
nargin	Funkcijos įėjimo argumentų skaičius.
nargout	Funkcijos išėjimo argumentų skaičius.
realmin	Mažiausias realusis skaičius, kurį MATLAB'as skiria nuo nulio; *realmin*=$2.225073858507201 \cdot 10^{-308}$
realmax	Didžiausias realusis skaičius, kurį MATLAB'as gali įsiminti; *realmax*=$1.797693134862316 \cdot 10^{308}$.
bitmax	Didžiausias galimas teigiamas sveikasis skaičius; *bitmax*=2^{53}-1.
varargin	Funkcijos kintamų įėjimo argumentų skaičius.
varargout	Funkcijos kintamų išėjimo argumentų skaičius.

Aritmetinės operacijos. Užrašant algebrines išraiškas, be konstantų ir kintamųjų naudojamos aritmetinės operacijos, kurios išvardytos 2.3-ioje lentelėje.

2.3 *lentelė. Aritmetinės operacijos*

Operacija	*Simbolis*	*Pavyzdys*
Sudėtis	+	3+15
Atimtis	-	3.14-10.3
Daugyba	*	8*13.4
Dalyba	/ arba\	5/2 arba 2\5=5/2
Kėlimas laipsniu	^	2^3

Išraiškos apskaičiuojamos iš kairės į dešinę laikantis tokios veiksmų atlikimo tvarkos: pirmiausia atliekamas laipsnio kėlimo veiksmas, turintis aukščiausią *prioritetą, po to - daugybos ir dalybos veiksmai, turintys tą patį prioritetą ir, galiausiai, sudėties ir atimties veiksmai, turintys tą patį žemiausią prioritetą.*

Skliaustai gali pakeisti veiksmų atlikimo tvarką – pirmiausia veiksmai atliekami skliaustuose. Tačiau kiekvienos skliaustų poros viduje veiksmai atliekami anksčiau nurodyta tvarka.

MATLAB'as operuoja su dvigubo tikslumo (double precision) skaičiais. Tai dažniausiai kompiuteriuose naudojamas formatas atliekant inžinerinius skaičiavimus. Šis formatas įgalina atlikti skaičiavimus su skaičiais, kurie vaizduojami 16-ka dešimtainių skilčių.

Kompiuterio aritmetiniame įrenginyje operacijos atliekamos su 32-iem dešimtainėmis skiltimis, tačiau rezultatas pateikiamas su 16-ka skilčių. ***Vadinasi, norint gauti didesnį skaičiavimo tikslumą, rašant išraiškas, nereikėtų naudoti tarpinių kintamųjų.***

Skaičių vaizdavimo formatai. MATLAB'o ekrane (komandų lange) skaičiai vaizduojami laikantis tokių taisyklių:

- jei skaičius yra sveikasis, tai jis ir vaizduojamas kaip sveikasis skaičius;
- jei skaičius yra realusis, tai galimi įvairūs jo vaizdavimo formatai, kurie surašyti 2.4-oje lentelėje, parodant, kaip šiais formatais vaizduojamas skaičius π.

2.4 *lentelė. Realiųjų skaičių vaizdavimo formatai*

MATLAB komanda	*pi*	*Komentarai*
format short	3.1416	5 skaitmenys.
format long	3.14159265358979	16 skaitmenų.
format short e	3.1416 e+000	5 skaitmenys ir dešimties laipsnis.
format long e	3.14159265358979e+000	16 skaitmenų ir dešimties laipsnis.
format short g	3.1416	Geriausias tarp formatų: "*format short*" ir "*format short e*".
format long g	3.14159265358979	Geriausias tarp formatų: "*format long*" ir "*format long e*".
format bank	3.14	2 dešimtainiai skaitmenys.
format +	+	Jei skaičius teigimas, tai "+"; jei neigiamas, tai "-": jei lygus 0, tai "0".
format rat	355/113	Skaičiaus racionalioji aproksimacija.

MATLAB'o funkcijų kategorijos. Kaip buvo minėta anksčiau, MATLAB'as turi labai daug funkcijų. Tam, kad būtų lengviau surasti norimą funkciją, jos yra suskirstytos į kategorijas. Kai kurios funkcijos yra realizuotos („įsiūtos") pačiame MATLAB'o interpretatoriuje, tačiau dauguma jų yra *m*-failų (žr. 2.7 paragrafą) pavidale. Pagrindinės funkcijų kategorijos yra šios:

- ***color*** – spalvų ir apšvietimo valdymo funkcijos,
- ***datafun*** – duomenų analizės ir Furje transformacijos funkcijos,
- ***demos*** – demonstravimo funkcijos,
- ***elfun*** – elementariosios funkcijos ,
- ***elmat*** – matricos ir manipuliavimo su jomis funkcijos,
- ***funfun*** – skaitinių metodų funkcijos,
- ***general*** – bendros paskirties funkcijos (komandos),
- ***graphics*** – grafinės funkcijos,
- ***iofun*** – įvedimo ir išvedimo funkcijos,
- ***lang*** – MATLAB'o programavimo kalbos konstrukcijų funkcijos,
- ***matfun*** – tiesinės algebros skaitinių metodų funkcijos,
- ***ops*** – MATLAB'o operatorių funkcijos,
- ***plotxy*** – dviejų dimensijų grafinės funkcijos,
- ***plotxyz*** – trijų dimensijų grafinės funkcijos,
- ***polyfun*** – polinomų ir interpoliavimo funkcijos,
- ***sounds*** – garso funkcijos,
- ***sparfun*** – retųjų matricų funkcijos,
- ***specmat*** – specialiųjų matricų funkcijos,
- ***specfun*** – specialiosios matematinės funkcijos,
- ***strfun*** – simbolių eilučių funkcijos.

Yra funkcijų, kurios pagal paskirtį gali būti priskirtos kelioms kategorijoms, todėl kai kurias funkcijas galima rasti ir vienoje ir kitoje funkcijų kategorijoje.

Funkcijų kategorijos yra puiki priemonė norint rasti norimos paskirties MATLAB'o funkciją. Tam tikslui komandų lange reikia surinkti pagalbos komandą (apie pagalbą žr. žemiau):

help *funkcijų kategorijos vardas.*

Komandų lange bus atspausdintos nurodytos kategorijos funkcijos su trumpais komentarais, apibūdinančiais jų paskirtį.

Elementariosios funkcijos. Į algebrines išraiškas dažnai įeina įvairios elementariosios funkcijos. MATLAB'as turi didelį rinkinį vidinių funkcijų, skirtų apskaičiuoti įvairias elementariąsias funkcijas. Dalinis jų sąrašas pateiktas 2.5-oje lentelėje.

2.5 *lentelė. MATLAB'o elementariosios funkcijos*

Funkcijos	*Aprašymas*
Trigonometrinės funkcijos	
sin(x), asin(x), sinh(x), asinh(x)	Sinusas, atvirkštinis sinusas, hiperbolinis sinusas, atvirkštinis hiperbolinis sinusas.
cos(x), acos(x), cosh(x), acosh(x)	Kosinusas, atvirkštinis kosinusas, hiperbolinis kosinusas, atvirkštinis hiperbolonis kosinusas
tan(x), atan(x), tanh(x), atanh(x)	Tangentas, atvirkštinis tangentas, hiperbolinis tangentas, atvirkštinis hiperbolinis tangentas.
cot(x), acot(x), coth(x), acoth(x)	Kotangentas, atvirkštinis kotangentas, hiperbolinis kotangentas, atvirkštinis hiperbolinis kotangentas.

sec(x), asec(x), sech(x), asech(x)	Sekantas (1/*cos*(x)), atvirkštinis sekantas, hiperbolinis sekantas, atvirkštinis hiperbolinis sekantas.
cse(x), acse(x), cseh(x), acseh(x)	Kosekantas (1/*sin*(x)), atvirkštinis kosekantas, hiperbolinis kosekantas, atvirkštinis hiperbolinis kosekantas.
Eksponentinės funkcijos	
^ (pvz., x^2)	Laipsnio kėlimo funkcija, (pvz., x^2).
exp(x)	Eksponentinė funkcija, e^x.
log(x)	Natūralusis logaritmas, *ln* (x).
log2(x)	Logaritmas pagrindu 2, $log_2(x)$.
log10(x)	Dešimtainis logaritmas, *lg*(x).
sqrt(x)	Kvadratinė šaknis, $\sqrt{x}$.
pow2(n)	Dvejeto laipsnis, *pow*2(5)=2^5.
nextpow2(n)	*p=nextpow2(n)*, tai mažiausias natūralusis skaičius, tenkinantis nelygybę $2^p \geq abs(n)$.
Apvalinimo ir liekanos funkcijos	
floor(x)	Skaičiaus x sveikoji dalis link minus begalybės, t.y., *floor*(x) =$\lfloor x \rfloor$, pvz., *floor* (2.7) =2, *floor* (-1,2) =-2.
ceil(x)	Skaičiaus sveikoji dalis link plius begalybės, t.y. *ceil*(x) =$\lceil x \rceil$. pvz., *ceil* (2.1) =3 *ceil* (-1.3) =-1.
fix(x)	Skaičiaus x sveikoji dalis link nulio $fix(x) = \begin{cases} \lfloor x \rfloor, jei\ x \geq 0, \\ \lceil x \rceil, jei\ x < 0. \end{cases}$
round(x)	Skaičiaus x apvalinimas pagal įprastas apvalinimo taisykles; pvz., round (1.235) =1.
mod(x,y)	Liekanos funkcija *mod* (*x,y*) =*x-n.*y*, čia *n=floor* (*x./y*), jei $y \neq 0$; *mod* (*x,y*) =*x*, jei y =0. ***Pastaba****. Simbolių pora „.*"reiškia paelementę daugybą, o pora "./" - paelementę dalybą. Šios operacijos bus paaiškintos vėliau 2.3.3 paragrafe.*
rem(x,y)	Kita liekanos funkcija: *rem* (*x,y*)=*x-n.*y,* čia *n=fix(x./y)*, jei $y \neq 0$; *rem* (*x,y*) =*nan*, jei y =0; *mod* (*x,y*) turi tą patį ženklą kaip ir *y*; tuo tarpu

	rem (*x,y*) turi tą patį ženklą kaip ir *x*. *mod (x,y) = rem (x,y*), jei *x* ir *y* turi tą patį ženklą; *mod* (*x,y*) ≠*rem* (*x,y*) jei *x* ir *y* ženklai yra skirtingi.
sign(x)	Ženklo funkcija; $sign\ (x) = \begin{cases} 1, jei\ \ x > 0, \\ 0, jei\ \ x = 0, \\ -1, jei\ \ x < 0. \end{cases}$
Kompleksinės funkcijos (z = x+i y)	
abs(z)	Kompleksinio skaičiaus modulis $abs(z) = \sqrt{x^2 + y^2}$.
angle(z)	Kompleksinio skaičiaus argumentas; *angle*(*z*) reikšmė yra tarp -π ir π ir apskaičiuojama pagal formulę $angle\ (z) = \begin{cases} arctg\ \ y/x,\ x \geq 0, \\ \pi + arctg\ \ y/x,\ x \leq 0, y > 0 \\ -\pi + arctg\ \ y/x,\ x < 0, y < 0 \end{cases}$
conj(z)	c*onj*(*z*)=*x–iy*, jungtinis kompleksinis skaičius
imag(z)	Kompleksinio skaičiaus menamoji dalis
real(z)	Kompleksinio skaičiaus realioji dalis
isreal(z)	"*true*" (tiesa), jei z – realusis skaičius
complex(a,b)	*complex(a,b) =a+i b*
sign(z)	Ši funkcija kompleksinio argumento atveju duos rezultatą: *sign*(*z*) = *z./abs*(*z*)

Komentarai. MATLAB'as, kaip ir kitos aukšto lygio programavimo kalbos leidžia naudoti komentarus, kurie įgalina paaiškinti komandą, reiškinį, kintamuosius, duomenis ir pan. arba pačią komandą interpretuoti kaip komentarą. Komentarams žymėti naudojamas procento simbolis, - "**%**". Komandos vykdymo metu eilutės tekstas už procento simbolio yra ignoruojamas – suprantamas kaip komentaras. Pavyzdžiui,

>> *n* = 3; % kintamasis *n* žymi lygčių skaičių

>> % *a* = 2;

Pirmuoju atveju komentaras paaiškina kintamojo *n* prasmę, o antruoju atveju komanda "*kintamajam n suteikti reikšmę* 2" suprantama kaip komentaras, t.y. ji neatliekama.

Rezultatų vaizdavimas. MATLAB'as dirba interpretavimo režimu, t.y. paeiliui interpretuoja komandas ir kiekvienos komandos rezultatą parodo ekrane - komandų lange. Jei norime, kad komandos rezultatas nebūtų išvedamas į ekraną, reikia šios komandos gale parašyti kabliataškį –"**;**". Ši galimybė yra labai patogi, nes bereikalingi tarpiniai skaičiavimo rezultatai nebus rodomi. Pavyzdžiui,

```
>> a=3;
>> b=6;
>> c=sqrt(a+b)
c =
     3
```

Atlikus šias komandas, tik galutinis skaičiavimo rezultatas *c* parodomas ekrane.

Komandų rašymo tvarka. Paprastai MATLAB'o komandos rašomos atskirose eilutėse. Norėdami kelias komandas parašyti vienoje eilutėje, jas turime skirti kableliais arba kabliataškiais. Po kablelių arba kabliataškių gali būti ir tarpo simbolių. Pavyzdžiui, prieš tai buvusio pavyzdžio komandas galime parašyti taip

```
>> a=3;,b=6; c=sqrt(a+b)
c = 3
```

Kartais MATLAB'o komandos yra labai ilgos ir jas patogiau rašyti keliose eilutėse. MATLAB'as įgalina jas taip rašyti panaudojant daugtaškio simbolį, -"**...**": eilutės gale esantis daugtaškis reiškia, kad kitos eilutės komanda yra jos tęsinys. Pavyzdžiui, komandos

```
    >> c=-12;
>> x1=-1/2+...
sqrt(1/4-c)
x1 =
     3
```

reiškia, kad *c* įgaus reikšmę "-12", $x1$ reikšmė bus apskaičiuota pagal formulę $x1=1/2+\sqrt{\frac{1}{4}-c}$ ir gautas rezultatas $x1$ bus parodytas ekrane. Reikia pabrėžti, kad komentarai arba kintamųjų vardai negali būti tęsiami per kelias eilutes

Skaičiavimo nutraukimas. Komandų interpretacija gali būti nutraukta bet kuriuo metu kartu paspaudus klavišus "***Ctrl***" ir "***C***".

Pagalba. MATLAB'e yra numatyta galimybė gauti informaciją apie MATLAB'o funkcijas, procedūras, jose realizuotus skaitinius metodus ir pan. Yra dvi galimybės gauti šią informaciją: 1) kreiptis į pagalbos žinyną ("*help*") arba 2) greitesnį- tiesiogiai, komandų lange, užrašant komandą:

help *funkcijos vardas*

arba

helpwin (*'funkcijos vardas'*).

Vienintelis skirtumas tarp šių komandų yra tas, kad pirmuoju atveju informacija apie funkciją išvedama į komandų langą, o antruoju atveju - atidaromas naujas peržiūros langas ("*help browser*") ir informacija pateikiama tame lange. Pavyzdžiui, jūs pamiršote, ar MATLAB'o funkcija *log* yra natūralusis ar dešimtainis logaritmas. Surinkę komandą

>> help log

gausite atsakymą:

LOG Natural logarithm (natūralusis logaritmas)

LOG (x) is the natural logarithm of the elements of x (yra elemento x natūralusis logaritmas)

Complex results are produced if x is not positive (rezultatas yra kompleksinis skaičius, jei x neteigiamas)

See also LOG10, EXP, LOGM

(žiūrėk taip pat funkcijas log10, exp, logm)

Šis sakinys sako, kad išvardytos funkcijos yra giminingos nagrinėjamai funkcijai.

Jei mes nežinome tikslaus funkcijos vardo, tai sužinoti apie ją galime komandos *lookfor simbolinė eilutė* pagalba. Pavyzdžiui, mes nežinome ar MATLAB'as turi hiperbolinio kosinuso apskaičiavimo procedūrą. Rašome komandą *>>lookfor cosine*, čia “*cosine*” yra simbolinė eilutė. Jei ši simbolinė eilutė yra funkcijos vardo dalis arba vidinės funkcijos pirmosios komentarų eilutės dalis, tai šių funkcijų vardai su paaiškinimais atspausdinami komandų lange. Tuo būdu į pateiktą komandą ekrane gausime atsakymą

ACOS inverse cosine (atvirkštinis kosinusas)
ACOSH inverse hyperbolic cosine (atvirkštinis hiperbolinis kosinusas)
COS cosine (kosinusas)
COSH hyperbolic cosine (hiperbolinis kosinusas)

Norėdami gauti smulkesnės informacijos apie hiperbolinį kosinusą, dabar galime pasinaudoti komanda

>> help cosh

Norėdami susipažinti su MATLAB‘o funkcijos tekstu, komandų lange reikia parašyti komandą : ***type*** *komandos_vardas.*

Šios komandos rezultatas yra komandų lange užrašytas nurodytos funkcijos tekstas.

Aritmetinių ir algebrinių išraiškų apskaičiavimo komandos.

Susipažinome su MATLAB’o kintamaisiais, operacijomis, komandomis ir kitais klausimais. Dabar galime aptarti aritmetinių ir algebrinių išraiškų apskaičiavimo komandas.

Aritmetinės išraiškos apskaičiavimo komanda yra: *“rezultato_vardas=aritmetinė išraiška*” arba “ *aritmetinė išraiška*”.

Pirmuoju atveju kintamasis “*rezultato_vardas*”, o antruoju atveju MATLAB’o kintamasis *“ans*” įgaus aritmetinės išraiškos rezultato reikšmę. Pavyzdžiui, komanda *x*=3(*sqrt*(5)-1)/2 reiškia, kad kintamasis *x* įgaus reikšmę $\frac{3(\sqrt{5}-1)}{2}$, t.y. $x = 1.8541$.

Algebrinės išraiškos apskaičiavimo komanda yra analogiška, tik žodžiai “aritmetinė išraiška” turi būti pakeisti žodžiais “algebrinė išraiška”. Be to, visų kintamųjų, įeinančių į algebrinę išraišką, reikšmės turi būti apibrėžtos prieš kreipiantis į algebrinės išraiškos apskaičiavimo komandą. Pavyzdžiui, komandos

>>*a*=1*;* *b*=5*;* *c*=-6*;*

>>*x*=-*b*+*sqrt(b*^2-4**a***c*)/(2**a*), reiškia, kad kintamasis *a* įgaus reikšmę 1, kintamasis *b* – reikšmę 5, kintamasis *c* – reikšmę – 6, o kintamasis *x* įgaus reikšmę $\frac{-5+\sqrt{25-4\cdot 1\cdot 6}}{2\cdot 1}$, t.y. $x = -2$.

2.3. Masyvai ir operacijos su masyvais

MATLAB’as, kaip ir kitos aukšto lygio programavimo kalbos, leidžia operuoti su duomenų struktūromis, kurios vadinamos masyvais. Masyvai gali būti ***vienmačiai***, pvz., vektoriai, aibės, ***dvimačiai*** – matricos ir ***daugiamačiai***. Pavyzdžiui, trimatį masyvą galime įsivaizduoti kaip knygą, kurios kiekvienas lapas yra dvimatis masyvas – matrica.

MATLAB’as skirtingai nuo kitų aukšto lygio programavimo kalbų turi labai patogią ir lanksčią komandų ir vidinių funkcijų sistemą, leidžiančią masyvus formuoti, performatuoti (keisti masyvų formatus), o taip pat atlikti veiksmus tiek su atskirais masyvo elementais arba jų poaibiais, tiek ir su pačiais masyvais.

Tolesniame dėstyme apsiribosime tik vienmačių ir dvimačių masyvų nagrinėjimu. Daugiamačių masyvų formavimas ir veiksmai su jais yra analogiški vienmačių ir dvimačių masyvų formavimui ir jų veiksmams. Be to, daugiamačiai masyvai skaitiniuose metoduose naudojami rečiau.

2.3.1. Masyvų konstravimas

Yra įvairių masyvų konstravimo būdų. Dažniausiai naudojami masyvų sudarymo būdai yra:

1) išvardinant masyvo elementus,
2) naudojant specialias komandas arba funkcijas, kurios įgalina manipuliuoti turimais masyvais arba keisti jų formatus, t.y. performatuojant masyvus,
3) išrenkant poaibius masyvų elementų, tenkinančių nurodytas sąlygas.

Masyvų formavimas išvardinant elementus. Masyvai, išvardinant jų elementus, sudaromi laikantis taisyklės: "*tarp laužtinių skliaustų rašomi masyvo elementai (skaičiai, kintamieji), kurie vienas nuo kito skiriami arba tarpais, arba kableliais, arba kabliataškiais.*

Jei elementai vienas nuo kito skiriami tarpais arba kableliais, tai kiekvienas elementas talpinamas į tos pačios eilutės skirtingus stulpelius. ***Suformuojama matrica eilutė.***

Jei elementai vienas nuo kito skiriami kabliataškiais, tai jie talpinami į to paties stulpelio skirtingas eilutes. ***Suformuojama matrica stulpelis (vektorius).*** Naudojant transponavimo operaciją, kuri žymima viršutiniu štrichu " **'** ", iš eilutės galima gauti stulpelį ir atvirkščiai, iš stulpelio galima gauti eilutę.

Pavyzdžiai.

```
>> x=[1 2 3 7 10]
x =
     1     2     3     7    10
>> a=[2; 5; -3; 8]
a =
     2
     5
    -3
     8
>> c=a'
c =
     2     5    -3     8
>> d=[1 2 3;4 5 6;7 8 9]
d =
     1     2     3
     4     5     6
     7     8     9
>> d=d'
d =
     1     4     7
     2     5     8
     3     6     9
>> y=[x c]
y =
     1     2     3     7    10     2     5    -3     8
```

Masyvų formavimas naudojant specialias komandas. Komandos, formuojančios vienmačius masyvus, surašytos 2.6 lentelėje.

2.6 lentelė. Vienmačių masyvų formavimo specialiosios komandos

Masyvų formavimo komandos	*Paaiškinimai*
x=first:h:last;	Ši komanda sukuria vienmatį masyvą (matricą eilutę) *x,* kurio pirmasis elementas yra skaičius "*first*", *k*-asis elementas gaunamas prie (*k*-1) elemento pridėjus žingsnį *h*, o paskutinysis elementas *b* tenkina sąlygą *last-b*<*h*. Pvz., `>> x=1:0.5:4.2` `x =` `    1.0000 1.5000 2.0000 2.5000 3.0000 3.5000` `4.0000`
y=first:last	Ši komanda sukuria vienmatį masyvą analogiškai kaip ir komanda *x=first:h:last,* tik šiuo atveju *h* =1. Pvz., `>> y=1:4.3` `y =` `     1 2 3 4`
x=linspace(first,last,n)	Funkcija *linspace* sukuria vienmatį masyvą, turintį *n* elementų ir, kurio pirmasis elementas yra "*first*", o paskutinysis – "*last*". Jei *n* reikšmė nenurodyta, tai laikoma, kad *n*=100. Pvz., `>> x=linspace(1,5,9)` `x =` `  Columns 1 through 7` `    1.0000 1.5000 2.0000 2.5000 3.0000 3.5000` `4.0000` `  Columns 8 through 9` `    4.5000 5.0000`
x=logspace(first,last,n)	Funkcija *logspace* sukuria vienmatį masyvą, turintį *n* elementų ir, kurio pirmasis elementas yra 10^{first}, o paskutinysis – 10^{last}. Pvz., `>> x=logspace(1,5,9)` `x =` `  1.0e+005 *` `  Columns 1 through 7` `    0.0001 0.0003 0.0010 0.0032 0.0100 0.0316` `0.1000` `  Columns 8 through 9` `    0.3162 1.0000`
x=randperm(n)	Ši komanda suformuoja masyvą *x,* kuris yra atsitiktinai sugeneruotas *n*-elementinis kėlinys.

Dvimačiai masyvai gali būti formuojami išvardinant elementus, kaip kad buvo paaiškinta anksčiau, o taip pat naudojant MATLAB'o vidines funkcijas. Šios funkcijos formuoja standartines matricas, pasižyminčias vienokiomis ar kitokiomis savybėmis, leidžia manipuliuoti masyvais arba performatuoti jau turimus masyvus. Dažniausiai naudojamų tokių funkcijų sąrašas pateikiamas 2.7 lentelėje.

2.7 lentelė. Dvimačių masyvų formavimo funkcijos

Funkcijos	*Paaiškinimai*

Standartinės matricos	
ones(n)	Suformuojama *n*-osios eilės kvadratinė matrica, kurios visi elementai lygūs 1. Pvz., `>> ones(3)` `ans =` `     1     1     1` `     1     1     1` `     1     1     1`
ones(m,n)	Taip pat, kaip ir *ones(n)* tik formuojama *m*×*n* formato matrica. Pvz., `>> ones(2,3)` `ans =` `     1     1     1` `     1     1     1`
zeros(n)	Suformuojama *n*-osios eilės kvadratinė matrica, kurios visi elementai lygūs 0. Pvz., `>> zeros(2)` `ans =` `     0     0` `     0     0`
zeros(m,n)	Taip pat, kaip ir *zeros(n*), tik formuojama *m*×*n* formato matrica. Pvz., `>> zeros(2,3)` `ans =` `     0     0     0` `     0     0     0`
eye(n)	Suformuojama *n*-osios eilės kvadratinė vienetinė matrica. Pvz., `>> eye(3)` `ans =` `     1     0     0` `     0     1     0` `     0     0     1`
eye(m,n)	Suformuojama *m*×*n* formato matrica, kurios viršutinė kairioji *k*-os eilės submatrica yra vienetinė matrica, o visi likę suformuotos matricos elementai lygūs 0, čia *k=min(m,n)*.Pvz., `>> eye(2,4)` `ans =` `     1     0     0     0` `     0     1     0     0`
rand(n)	Suformuojama *n*-osios eilės kvadratinė matrica, kurios elementai yra skaičiai tolygiai pasiskirstę intervale [0,1]. Pvz., `>> rand(2)` `ans =` `    0.9501    0.6068` `    0.2311    0.4860`

rand(m,n)	Taip pat, kaip ir *rand(n),* tik matricos formatas yra $m\times n$.
randn(n)	Funkcija *randn(n)* formuoja matricas, kaip ir *rand(n),* tik šiuo atveju matricos elementai yra atsitiktiniai skaičiai, pasiskirstę pagal normalųjį dėsnį, kurio vidurkis lygus nuliui, o dispersija – vienetui. Pvz., `>> randn(2)` `ans =` `   -0.4326    0.1253` `   -1.6656    0.2877`
randn(m,n)	Taip pat, kaip ir *rand(n),* tik matricos formatas yra $m\times n$.
randint(m,n,range)	Funkcija generuoja $m\times n$ formato matricą, sudarytą iš sveikųjų skaičių, tolygiai pasiskirsčiusių intervale [*range*(1), *range*(2)], čia *range*(1), *range*(2) yra sveikieji skaičiai.
Specialiosios matricos	
h=hadamard(n)	Suformuojama *n*-os eilės Adamaro matrica *h*. Ji sudaryta iš vienetukų ir minus vienetukų ir jos stulpeliai tarpusavyje yra ortogonalūs t.y. *h'*h=n*I,* čia *I* – vienetinė *n*-os eilės matrica. Adamaro matrica taikoma kombinatorikoje, signalų analizėje ir kt. Pvz., `>> hadamard(4)` `ans =` `     1     1     1     1` `     1    -1     1    -1` `     1     1    -1    -1` `     1    -1    -1     1`
h=hankel(c) *h=hankel(c,r)*	Funkcija *hankel (c)*, čia *c* yra *n*-elementis vienmatis masyvas, suformuoja *n*-os eilės kvadratinę Hankelio matricą, kurios pirmasis stulpelis yra *c*, o visi elementai, esantys žemiau pagrindinės antiįstrižainės yra lygūs nuliui. Funkcija *hankel(c,r),* čia *c* - *n*-elementis, *r* – *m*-elementis vienmačiai masyvai, suformuoja ($n\times m$) Hankelio matricą, kurios pirmasis stulpelis yra *c*, o paskutinioji eilutė *r*. Hankelio matricos elementai, priklausantys tai pačiai antįstrižainei, yra lygūs ir formuojami pagal formulę : $h_{ij} = p_{i+j-1}, \quad i=\overline{1,n}, \quad j=\overline{1,m},$ kur *p* = [*c r*[2:*end*]]. Pvz., `>> c=1:3;` `>> r=7:10;` `>> h=hankel(c,r)` `h =` `     1     2     3     8` `     2     3     8     9` `     3     8     9    10` *Pastaba:* matrica *h* buvo formuojama remiantis masyvu *p*=[1, 2, 3, 8, 9, 10]
h=hilb(n)	Funkcija *hilb(n)* suformuoja *n*-osios eilės kvadratinę Hilberto

	matricą. Hilberto matricos elementai apskaičiuojami pagal formulę: $h_{ij}=1/(i+j-1),\quad i=\overline{1,n},\quad j=\overline{1,n}.$ Hilberto matrica, tai blogai sąlygotos matricos pavyzdys.
p=pascal(n)	Funkcija *pascal(n)* formuoja *n*-os eilės kvadratinę simetrinę teigiamai apibrėžtą matricą, sudarytą pagal Paskalio trikampį. Paskalio matricos pirmos eilutės ir pirmo stulpelio elementai yra lygūs 1. Paskalio matricos ir jos atvirkštinės matricos elementai yra sveikieji skaičiai, t.y. Paskalio matricos determinantas yra lygus 1. Pvz., `>> p=pascal(4)` `p =` `     1     1     1     1` `     1     2     3     4` `     1     3     6    10` `     1     4    10    20`
v=vander(x)	Funkcija *vander(x),* čia *x* yra *n*-elementis vienmatis masyvas formuoja *n*-os eilės kvadratinę Vandermondo matricą. Vandermondo matrica naudojam sprendžiant, pavyzdžiui, interpoliavimo uždavinį. Vandermondo matricos elementai apskaičiuojami pagal formulę: $v_{ij}=x_i^{n-j},\quad i=\overline{1,n},\quad j=\overline{1,n}.$ Pvz., `>> v=vander(1:0.5:3)` `v =` `    1.0000    1.0000    1.0000    1.0000    1.0000` `    5.0625    3.3750    2.2500    1.5000    1.0000` `   16.0000    8.0000    4.0000    2.0000    1.0000` `   39.0625   15.6250    6.2500    2.5000    1.0000` `   81.0000   27.0000    9.0000    3.0000    1.0000`
gallery	Funkcija *gallery* gali formuoti vieną iš 54 specialios paskirties matricų. Plačiau apie šią funkciją galima sužinoti iš MATLAB'o pagalbos žinyno.
Manipuliavimas masyvais	
diag(x)	Jei *x* yra vienmatis *n*-elementis masyvas (eilutė arba stulpelis), tai funkcija *diag*(*x*) sukuria *n*-os eilės kvadratinę įstrižaininę matricą, kurios pagrindinė įstrižainė yra masyvas *x*. Pvz., `>> x=[1 -2 4];` `>> d=diag(x)` `d =` `     1     0     0` `     0    -2     0` `     0     0     4` Jei *x* yra matrica, tai *diag(x)* sukuria vienmatį masyvą (stulpelį), kurio elementai yra matricos *x* pagrindinės įstrižainės elementai. Pvz., `>> a=[1 2 3;4 5 6;7 8 9]` `a =`

	`` 1 2 3 4 5 6 7 8 9 >> diag(a) ans = 1 5 9 >> b=[1 2 3;4 5 6] b = 1 2 3 4 5 6 >> diag(b) ans = 1 5 ```
diag(x,k)	Jei *x* – vienmatis *n*-elmentis masyvas, tai funkcija *diag(x,k)* sukuria (*n+abs(k)*)-os eilės kvadratinę matricą, kurios *k*-oji šalutinė įstrižainė yra masyvas *x*, o visi likę elementai lygūs nuliui. Jei *k*>0, tai *k*-oji šalutinė įstrižainė bus talpinama virš pagrindinės įstrižainės, o, jei *k*<0, tai – žemiau pagrindinės įstrižainės. Pvz., ``` >> x=[-1 3 4] x = -1 3 4 >> diag(x,1) ans = 0 -1 0 0 0 0 3 0 0 0 0 4 0 0 0 0 >> diag(x,-2) ans = 0 0 0 0 0 0 0 0 0 0 -1 0 0 0 0 0 3 0 0 0 0 0 4 0 0 ``` Jei *x* yra matrica, tai funkcija *diag*(*x,k*) sukuria vienmatį masyvą –stulpelį, kuris yra matricos *x k*-oji šalutinė įstrižainė. Pvz., `>> a=[1 2 3;4 5 6;7 8 9]` ``` a = 1 2 3 4 5 6 7 8 9 >> diag(a,1) ans = 2 6 >> diag(a,-1) ans = 4 8 ```
rot90(a)	Ši funkcija masyvą *a* pasuka 90-čia laipsnių kampu prieš laikrodžio rodyklę. Pvz., ``` a = 1 2 3 4 5 6 7 8 9 ```

<table>
<tr><td></td><td><pre>>> rot90(a)
ans =
     3   6   9
     2   5   8
     1   4   7
>> x
x =
    -1   3   4
>> rot90(x)
ans =

     4
     3
    -1</pre></td></tr>
<tr><td>rot90(a,k)</td><td>Ši funkcija masyvą a pasuka 90*k laipsnių kampu prieš laikrodžio rodyklę, jei k>0, ir - pagal laikrodžio rodyklę, jei k<0.
Pvz.,<pre>a =
     1   2   3
     4   5   6
     7   8   9
>> rot90(a,-1)
ans =
     7   4   1
     8   5   2
     9   6   3
>> x=[1 4 8]
x =
     1   4   8
>> rot90(x,2)
ans =
     8   4   1</pre></td></tr>
<tr><td>fliplr(a)</td><td>Ši funkcija masyvą a perrašo iš kairės į dešinę.
Pvz., <pre>a = 1   2   3
         4   5   6
         7   8   9
>> fliplr(a)
ans = 3   2   1
       6   5   4
       9   8   7</pre></td></tr>
<tr><td>Flipud(a)</td><td>Ši funkcija masyvą a perrašo iš apačios į viršų. Pvz.,<pre>a =
     1   2   3
     4   5   6
     7   8   9
>> flipud(a)
ans =
     7   8   9
     4   5   6
     1   2   3
>> x=[1;2;3]
x =
     1
     2
     3</pre></td></tr>
</table>

<table>
<tr><td></td><td><pre>>> flipud(x)
ans =
     3
     2
     1</pre></td></tr>
<tr><td>tril(a)</td><td>Ši funkcija iš matricos a suformuoja apatinę trikampę matricą visus matricos a elementus, esančius virš pagrindinės įstrižainės, pakeisdama nuliais.</td></tr>
<tr><td>triu(a)</td><td>Taip pat kaip ir tril(a), tik suformuojama viršutinė trikampė matrica. Pvz.,<pre>a =
     1   2   3
     4   5   6
     7   8   9
>> tril(a)
ans =
     1   0   0
     4   5   0
     7   8   9
>> triu(a)
ans =
     1   2   3
     0   5   6
     0   0   9</pre></td></tr>
<tr><td>x=a(:)</td><td>Ši komanda konvertuoja matricą a į stulpelį x surašydama vieną po kito matricos a stulpelius. Pvz.,<pre>>> a=[2 5 7;3 8 1]
a = 2   5   7
     3   8   1
>> a(:)
ans =
     2
     3
     5
     8
     7
     1</pre></td></tr>
<tr><td>Masyvų performatavimas</td><td></td></tr>
<tr><td>reshape(a,m,n)</td><td>Ši funkcija transformuoja masyvą a, turintį m*n elementų, į m×n formato matricą. Pvz.,<pre>>> a=linspace(1,9,9)
a = 1 2 3 4 5 6 7 8 9
>> b=reshape(a,3,3)
b =
     1   4   7
     2   5   8
     3   6   9
>> c=[1 2 3 4;5 6 7 8;9 10 11 12]
c =
     1   2   3   4
     5   6   7   8
     9   10   11   12</pre></td></tr>
</table>

	`>> d=reshape(c,4,3)` `d =` `     1     6    11` `     5    10     4` `     9     3     8` `     2     7    12` Performatuojant matricą *c*, ją galima įsivaizduoti kaip 12-kos elementų vektorių, sudarytą iš nuosekliai vienas po kito išdėstytų matricos *c* stulpelių. Iš šio vektoriaus formuojama 4×3 matrica: iš eilės imami 4-elemenčiai poaibiai, kurie sudaro formuojamos matricos *d* stulpelius.
repmat(a,m,n)	Ši funkcija formuoja *m*×*n* *f*ormato matricą, pakartodama masyvą *a*, t.y. masyvas *a* yra naujai formuojamos matricos konstrukcinis elementas, kartojamas per *m* eilučių ir *n* stulpelių.
repmat(a,n)	Taip pat kaip ir *repmat*(*a,m,n*), tik šiuo atveju eilučių ir stulpelių skaičius yra *n*. Pvz., `>> a=[1 2 3]` `a =     1     2     3` `>> b=repmat(a,2,3)` `b =` `     1  2  3  1  2  3  1  2  3` `     1  2  3  1  2  3  1  2  3` `>> c=[1 2;3 4]` `c =` `     1     2` `     3     4` `>> d=repmat(c,2)` `d =` `     1     2     1     2` `     3     4     3     4` `     1     2     1     2` `     3     4     3     4`

Trečiasis masyvų formavimo būdas, kai nauji masyvai formuojami iš turimų masyvų išrenkant elementus, tenkinančius nurodytas sąlygas, glaudžiai siejasi su masyvų elementų adresavimu ir loginėmis operacijomis, kurios yra nagrinėjamos tolesniuose paragrafuose.