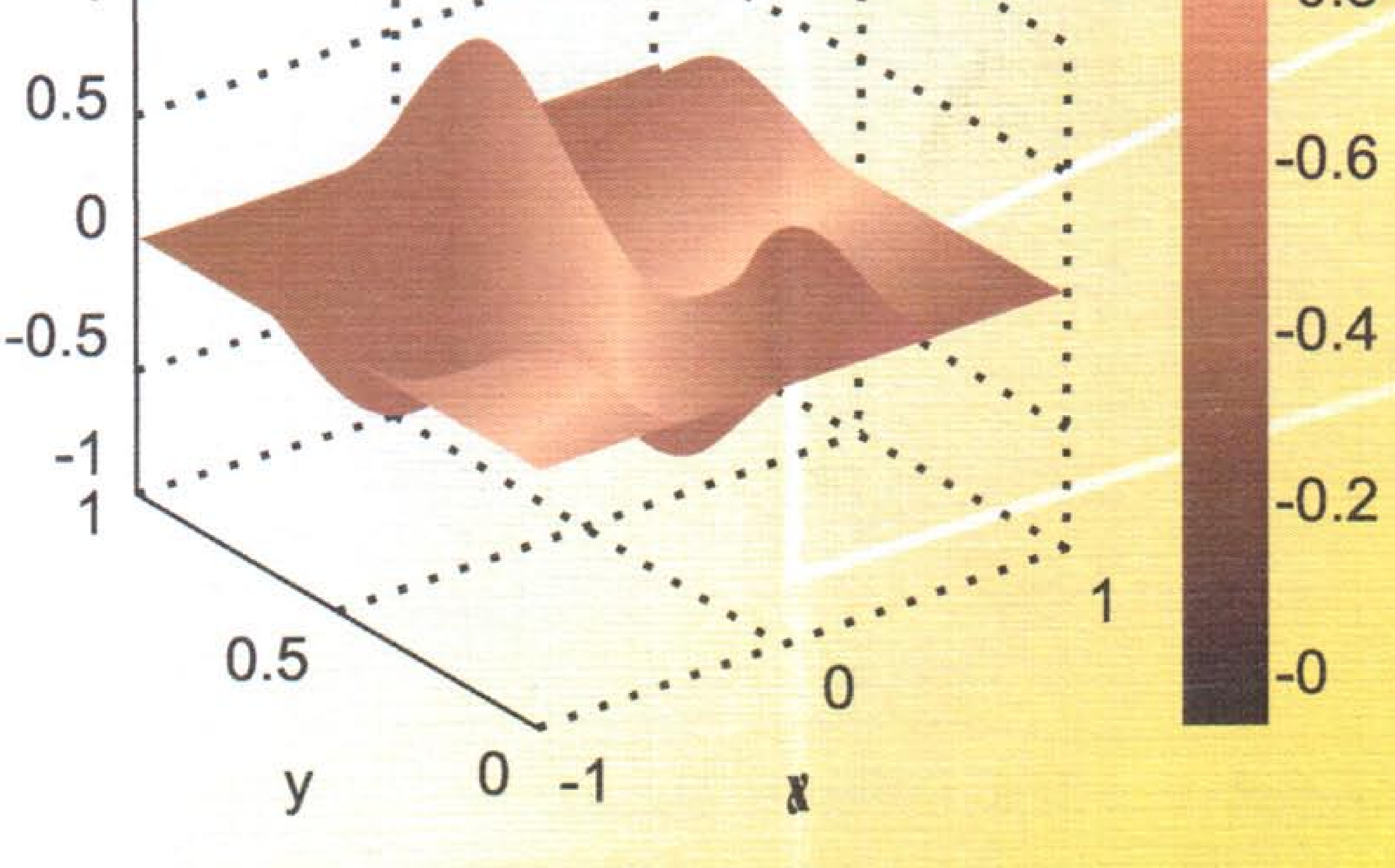
К. С. Дүйсебекова • М. Е. Мансұрова

MATLAB-та ПРОГРАММАЛАУ НЕГІЗДЕРІ

ОҚУ ҚҰРАЛЫ



ӘЛ-ФАРАБИ атындағы ҚАЗАҚ ҰЛТТЫҚ УНИВЕРСИТЕТІ

К. С. Дүйсебекова
М. Е. Мансұрова

MATLAB-та ПРОГРАММАЛАУ НЕГІЗДЕРІ

Оқу құралы

Алматы
«Қазақ университеті»
2011

ББК 32. 97
Д 88

Баспаға әл-Фараби атындағы Қазақ ұлттық университеті химия факультетінің Ғылыми кеңесі; әл-Фараби атындағы Қазақ ұлттық университетінің жанындағы ҚР БжҒМ-нің жоғары және жоғары оқу орнынан кейінгі білім берудің Республикалық оқу-әдістемелік кеңесінің гуманитарлық және жаратылыстану ғылымдары мамандықтары Секция мәжілісі шешімімен және Редакциялық-баспа кеңесі ұсынған (№ 2 хаттама 12 маусым 2009 жыл)

Пікір жазғандар:

физика-математика ғылымдарының докторы ***М.Н. Қалимолдаев;***
физика-математика ғылымдарының докторы, профессор ***А.Е. Дүйсебаев***

Дүйсебекова К.С., Мансұрова М.Е.
Д 88 Matlab-та программалау негіздері: оқу құралы. – Алматы: Қазақ университеті, 2011. – 142 б.
ISBN 9965-29-653-7

Оқу құралында Matlab интерактивті ортасының математикалық есептеулерде және математикалық модельдеуде қолданылуы қарастырылған. Әсіресе осы ортада программалауға ерекше көңіл бөлінген. Бұл оқу құралы жоғары оқу орнында «Информатика», «Ақпараттық жүйелер», «Математикалық және компьютерлік модельдеу» мамандықтары бойынша даярланып жатқан студенттерге, магистранттарға, PhD докторанттарға ұсынылады.

Д $\frac{1602070000}{460(05) - 11}$ **ББК 32. 97**

ISBN 9965-29-653-7

Мазмұны

Кіріспе

Matlab жүйесі (*MATrix LABoratory – МАТрицалық ЛАБоратория*) сөзінен қысқартылып алынған, деректер массивтерімен жұмыс істеуге бағытталған инженерлік және ғылыми есептеулерді орындауға арналған интерактивті жүйе болып табылады. Бұл жүйе математикалық қосымша процессорды колдана отырып, *Fortran, C, C++* тілдерінде жазылған программаларға қатынасуға мүмкіндік береді.

Matlab – жоғарғы өнімділікті техникалық есептеулер тілі. Оған қолданушыға ыңғайлы ортада есептеулер жүргізу, көрсету, программалау кірген. *Matlab*-ты қолдану дегеніміз:

- математикалық есептеулер;
- алгоритмдерді құру;
- үлгілеу;
- деректерді талдау, зерттеу, көрсету;
- ғылыми және инженерлік графика;
- графикалық интерфейспен қоса қосымшаларды құру.

Matlab-та *toolboxes* деп аталатын арнайы программалар тобына ерекше орын берілген. Арнайы тәсілдерді қолдануға және зерттеуге мүмкіндік беретіндіктен, олардың *Matlab* ортасында жұмыс істеушілер үшін рөлі зор.

Toolboxes – *Matlab*-тың есептердің ерекше топтарын шешуге арналған функцияларының (*M*-файлдар) жан-жақты коллекциясы. *Toolboxes* сигналдарды өңдеуде, нейрон желілерінің бақылау жүйесінде, айқын емес логикада, вейвлеттерде, үлгілеуде қолданылады.

Matlab жүйесі негізгі бес бөліктен тұрады.

Matlab – ағымдарды, функцияларды, деректер құрылымдарын, енгізу мен қорытындылауды басқаруға арналған және объектіге бағытталған программалау ерекшеліктері бар жоғарғы деңгейлі массивтер мен матрицалардың тілі. Бұлар *Matlab* ортасында кішігірім программалармен қатар күрделі қосымшаларды құруға мүмкіндік береді.

Matlab ортасы – қолданушыға немесе программалаушыға осы ортада жұмыс істеуге жағдай жасайтын аспаптар мен құралдар тобы. Олар *Matlab*-тың жұмыс кеңістігінде айнымалыларды басқаруға, енгізу-қорытындылау амалдарын орындауға, *M*-файлдарды құруға, бақылауға, орындауға және *Matlab* қосымшаларын құруға арналған құралдардан тұрады.

Басқарылатын графика – бұл *Matlab* ортасының екі немесе үш өлшемді деректерді көрсетуге, кескіндерді өңдеуге, анимациялауға арналған жоғарғы деңгейлі командалардан тұратын графикалық жүйесі. Графиктің сыртқы түрін редакциялауға және *Matlab* қосымшалары үшін Қолданушының Графиктік Интерфейсін (*GUI*) құруға арналған төменгі деңгейлі командалар да осында кірген.

Математикалық функциялардың кітапханасы. Бұл – қосынды, синус, косинус, кешенді арифметика сияқты элементарлық функциялардан бастап, матрицаларды қайтару, өзіндік мәндерді табу, Бессель функциялары және Фурьенің тез түрлендіруі сияқты күрделі функцияларды есептеу алгоритмдерінің жан-жақты коллекциясы.

Программалық интерфейс. *Matlab* ортасымен әрекеттесе алатындай *Fortran* және *C* тілдеріндегі программаларды жазуға мүмкіндік беретін кітапхана болып табылады. *Matlab* ортасын есептеу құралы ретінде пайдалана отырып, ортадан программаларды шақыруға (динамикалық байланыс) және *MAT*-файлдарды оқуға және жазуға арналған жабдықтар да осы кітапхананы құрайды.

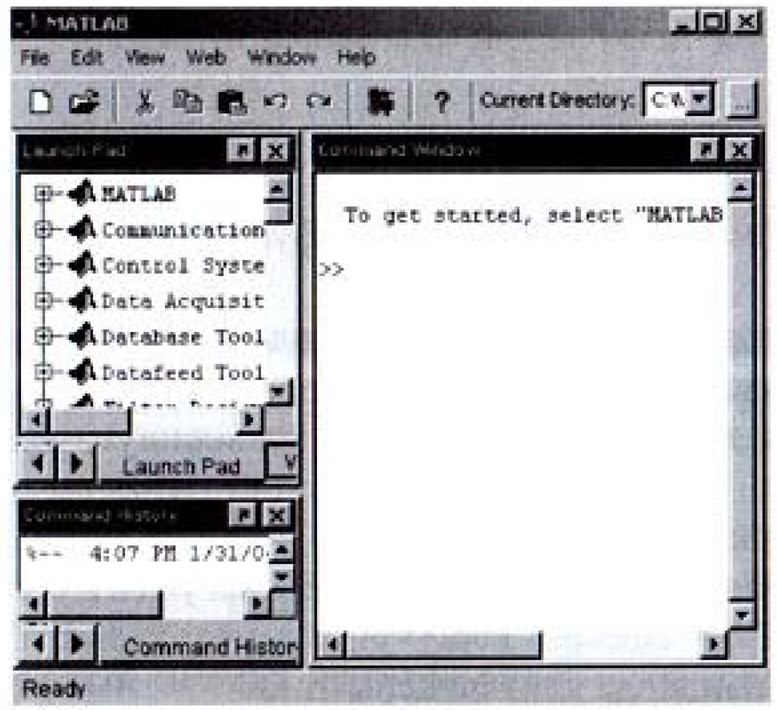
Бірінші сабақ. *Matlab* ортасымен танысу

Сабақтың жоспары

1. *Matlab*-тың жұмыс ортасы.
2. Арифметикалық есептеулер. Есептеу нәтижесін қорытындылау форматтары.
3. Қарапайым функцияларды пайдалану. Ішкі қарапайым функциялар.
4. Айнымалыларды қолдану.
5. Жұмыс ортасын сақтау.
6. Айнымалыларды қарап шығу.
7. Массивтермен жұмыс жасау. Негізгі анықтамалар мен келісімдер.
8. Векторлармен амалдар орындау.
9. Векторлардың элементтерімен амалдар орындау.
10. Векторларды көбейту.

Matlab-тың жұмыс ортасы

Matlab 6.x-ті іске қосу 1.1-суретінде көрсетілген жұмыс ортасының ашылуынан басталады.



1.1-сурет. *Matlab* 6.x-тің жұмыс ортасы

Жұмыс ортасы келесідей элементтерді қамтиды:

- меню;
- батырмалары және ашылатын тізімі бар аспаптар тақтасы;
- *ToolBox*-тың әр түрлі модульдеріне және жұмыс ортасының мазмұнына өте оңай кіруге болатындай *Launch Pad* және *Workspace* ішкі терезелері бар терезе;
- ағымдағы буманы іске қосуға болатын, сонымен қатар алдын-ала

енгізілген командаларды қайта қарауға болатын *Command History* және *Current Directory* ішкі терезелері бар терезе;

- командалық терезе;
- қалып-күй жолы.

Барлық командаларды командалар жолында теру керек. *Matlab* программасы барлық командаларды орындап, өрнектерді есептеу үшін, әр команда соңында <*Enter*> пернесі басылуы керек.

Арифметикалық есептеулер

Matlab-тың құрамдас математиканың функциялары әр түрлі өрнектердің мәндерін табуға мүмкіндік береді. *Matlab*-та нәтижелерді қорытындылау форматын басқару мүмкіндігі бар. Өрнектерді есептеуге арналған командалар барлық жоғарғы деңгейдегі программалау тілдеріне сай, түсінікті түрде келтірілген.

Қарапайым есептеулер

Командалық жолда *3+5*-ті теріп және <*Enter*>-ді басыңыз. Нәтижесінде *Matlab*-тың командалық терезесінде келесідей көрініс болады:

```
>> 3 + 5
ans =
      8
>> I
```

Төменде *Matlab*-тың құрамдас функцияларының мысалдары келтірілген:

- *sin, cos, tan, cot* – синус, косинус, тангенс және котангенс;
- *sec,csc* – секанс, косеканс;
- *asin, acos, atan, acot* – арксинус, арккосинус, арктангенс және арккотангенс;
- *asec, acsc* – арксеканс, арккосеканс.

Тригонометриялық функциялардың аргументтері радиандарда берілуі керек. Сол сияқты кері тригонометриялық функциялар да нәтижелерді радиандарда қайтарулары тиіс.

Кешенді сандармен жұмыс істеуге қажетті функциялар

Оларға *Matlab*-тың келесідей функцияларын жатқызуға болады:

- *abs, angle* – модуль *r* және фаза φ (π-дан π-ға дейінгі радиандарда) кешенді сандар $a + i*b = r*(\cos\varphi + i*\sin\varphi)$;
- *complex* – кешенді санды оның нақты және жорамал бөліктері арқылы құрастырады:

```
>> complex(2.3, 5.8)
```



```
ans =
      2.3000 + 5.8000i;
```

- *conj* – кешенді-сәйкес санды қайтарады;
- *imag, real* – кешенді санның нақты және жорамал бөліктерін қайтарады.

Дөңгелектеу және бөлінгеннен кейінгі қалдық

- Төменде осы функцияларды *Matlab*-та қолдану мысалдары келтірілген:
- *fix* – нөлге қарай ең жақын бүтін санға дейін дөңгелектеу;
- *floor, ceil* – $-\infty$ немесе $+\infty$-ке қарай ең жақын бүтін санға дейін дөңгелектеу;
- *round* – ең жақын бүтін санға дейін дөңгелектеу;
- *mod* – бүтін сандық бөлуден қалған қалдық (таңбасымен қоса);
- *rem* – бүтін сандық бөлуден қалған қалдық;
- *sign* – сан таңбасын қайтарады.

Айнымалыларды қолдану

Matlab-та айнымалылармен жұмыс істеу мүмкіндігі қарастырылған. Сонымен қатар, енгізілетін айнымалыларды тегін көрсетудің қажеті жоқ.

Мысал келтірейік:

```
>> a = 3.67
a =
   3.67
```

Matlab-та командаларды нүктелі-үтірмен аяқтауға болады. Бұл жағдайда амалдар орындалғанмен, оның нәтижесі экранға шығарылмайды.

Жұмыс ортасын сақтау

Барлық айнымалылардың мәнін сақтаудың бір тәсілі – ол *File* менюінен *Save Workspace As* пунктін таңдау. Алдын-ала келісім бойынша *Matlab*-тың негізгі каталогының *work* ішкі каталогында файлды сақтау мүмкіндігі бар. *Matlab* жұмыс нәтижесін файлда **.mat* кеңейтілуімен сақтайды. Енді *Matlab*-ты келесі тәсілдердің бірін қолданып жабуға болады:

- *File* менюінде *Exit Matlab* пунктін таңдау арқылы;
- <*Ctrl*>+<*Q*> пернелерін басу арқылы;
- командалық жолда *Exit* командасын таңдап <*Enter*>-ді басу;
- *Matlab* программасының терезесінің жоғарғы оң жақ бөлігінде орналасқан жабу батырмасын басу арқылы.

Айнымалылардың мәндерін қалпына келтіру үшін құрылған файлды *File* менюінің *Open* ішкі пунктін таңдау арқылы жүзеге асыруға болады. Осыдан кейін барлық айнымалылар іске қосылады және оларды келесі сеанста пайдалануға болады.

Жұмыс ортасының айнымалыларын сақтау және қалпына келтіруді командалық жолдан іске асыруға болады. Ол үшін *save* және *load* командалары қолданылады. *Matlab*-пен жұмыс сеансының соңында

>>save session_1

командасын орындау керек.

Ал келесі сеансты бастаған кезде айнымалыларды оқу үшін

>>load session_1

командасын орындау керек.

Save және *load* командалары туралы ақпараттарды командалық жолда *help save* және *help load*-ты терген кезде алуға болады. **.mat* кеңейтілуімен берілген файлдарда айнымалылар екілік түрінде сақталады.

Matlab-та орындалатын командаларды және оның нәтижелерін мәтіндік файлға жазып және оны кез-келген мәтіндік редакторда оңай оқып, басып шығару мүмкіндігі бар. Журналдың алғашқы басталуы үшін *diary* командасы қолданылады. *Diary* командасының аргументі ретінде жұмыс журналы кай файлда сақталатын болса, сол файлдың аты берілуі керек:

>>diary session_1.txt.

Жұмыс сеансын жазуды тоқтатқан кезде:

>>diary off-ты теру керек.

Жұмыс кеңістігі

Жұмыс кеңістігі бұл – *Matlab*-тың командалық жолынан хабарласуға болатын жады аймағы. *Who* және *whos* командалары жұмыс кеңістігінің ағымдағы жағдайын көрсетеді. *Who* командасы қысқа тізімді береді, ал *whos* командасы өлшемді және қолданылатын жады көлемін көрсетеді.

Вектор-бағаналар және вектор-жолдар. Векторларды енгізу, қосу және олардың айырмасын табу

Векторлардың қосындысын есептеу керек делік $a=\begin{pmatrix}1.3\\5.4\\6.9\end{pmatrix}, b=\begin{pmatrix}7.1\\3.5\\8.2\end{pmatrix}$.

Векторларды сақтау үшін *a* және *b* массивтерін колданыңыз. Командалық жолда вектор бағанын жазу үшін тік жақшаны пайдаланып, вектор элементтерін нүктелі-үтірмен ажырата отырып, *a* массивін енгізіңіз.


```
>> a = [1.3; 5.4; 6.9]
a =
    1.3000
    5.4000
    6.9000
>> b = [7.1; 3.5; 8.2];
```

Векторлар қосындысы

```
>> c = a + b
c =
    8.4000
    8.9000
    15.1000
```

тең *c* массивінің өлшемін *ndims* және *size* функциялары арқылы анықтауға болады:

```
>> ndims (a)
ans =
      2
>> size(a)
ans =
      3   1
```

Бірақ көбейтілетін векторлардың өлшемдері бірдей болулары керек.

Вектор-жолды қалыптастырған кезде тік жақшалар қолданылады, сонымен қатар, вектор элементтері бір-бірінен бос орын арқылы ажыратылады.

Matlab тілінің көмегімен элементтері арифметикалық прогрессияны құрайтын векторларды қысқартып енгізуге болады. Егер *d0* осы прогрессияның бастапқы мәні, *dn* – соңғы мәні, *h* – прогрессия қадамы болса, онда векторды [*d0:h:dn*]: жазбасы арқылы енгізуге болады.

>> v1 = [0: 0.1:2];

Векторлармен амалдар орындау

Matlab-та векторларға жасалатын әрекеттер екі топқа бөлінеді: математикадағы векторлық есептеулер, вектор элементтерін түрлендіретін әрекеттер.

Векторды санға көбейту, қосу, азайту, жолдар мен бағандарын ауыстырып түрлендіру, векторларды өзара көбейту арифметикалық амалдардың таңбалары арқылы жүзеге асырылады.

Мысалы:

```
>>x = [1 2 3]; y = [5;7;8];
>>v = x*y
v = 43
```

Мәліметтерді өңдеу функцияларын векторларға қолдану

Төменде векторларға қолданылатын кейбір функциялар көрсетіледі:
prod(z) – *z* векторының элементтерін көбейту;
length(z) – *z* векторының ұзындығын анықтау;
sum(z) – *z* векторының элементтерінің қосындысын анықтау;
sort(z) – векторды *z* бойынша өсетіндей реттеу;
min(z) – *z* векторының элементтерінің ішінен минимумды табу;
max(z) – *z* векторының элементтерінің ішінен максимумды табу.

Екі аргументті қайтаратын *min* функциясын шақыру *m* айнымалысына *z* массивінің минималды элементінің мәнін меншіктейді, ал минималды элементтің нөмірін *k* айнымалысына кіргізеді.

```
>>[m, k] = min(z)
```

Векторлардың элементтерімен амалдар орындау

Matlab-та бар вектордың барлық элементтерінің мәні есептеледі.

Мысалы:

```
>> d = sin(c)
d =
   0.8546
   0.5010
   0.5712
```

Matlab-та векторлармен элементтер бойынша жұмыс істеу қарастырылады және ол кейіннен графиктерді құруға және функция мәндерін есептеуге мүмкіндік береді.

Екі вектор-жолды енгізіңіз:

```
>> v1 = [2  -3   4    1];
>> v2 = [7   5   -6   9];
```

.* (нүкте мен жұлдызша арасында бос орын қалдырмаңыз) операциясы бірдей ұзындықтағы векторларды элементтері бойынша көбейтуге мүмкіндік береді. Нәтижесінде элементтері берілген вектор элементтерінің қосындысына тең келетін вектор шығады:

```
>> u = v1.*v2
u =
   14  -15   -24   9
```

.^ амалының көмегімен векторларды элементтері бойынша дәрежеге шығару орындалады:

```
>> p = v1.^2
p =
   4    9    16    1
```


Дәреже көрсеткіші ретінде ұзындығы дәрежеге шығарылатын вектордың ұзындығына тең вектор алынады. Бұл кезде, бірінші вектордың әрбір элементі, оған сәйкес екінші вектордің элементіне тең дәрежеге шығарылады:

```
>> p = v1.^v2
p =
   128.0000      -243.0000         0.0002            1.0000
```

Ұзындықтары бірдей екі вектордың сәйкес элементтерін бөлу үшін ./ амалы қолданылады.

```
>> d = v1./v2
d =
   0.2857 -0.6000 -0.6667 0.1111
```

Элементтері бойынша кері бөлу (екінші вектор элементтерін сәйкесінше бірінші вектор элементтеріне бөлу) .\ амалының көмегімен жүзеге асады.

```
>> dinv = v1.\v2
dinv =
      3.5000  -1.6667  -1.5000   9.0000
```

Matlab-та нүкте ондық бөлшектерді енгізу үшін ғана емес, сонымен қатар, бірдей өлшемді массивтерді көбейту немесе бөлу олардың элементтері бойынша орындалуы керектігін білдіреді.

/ таңбасының көмегімен векторды санға бөлуге болады:

```
>>  p = v/2
p =
   2     3     4     5
```

Егер санды вектордың әрбір элементіне бөліп, оның нәтижесін жаңа векторға жазу керек болса, онда ./ амалы қолданылады:

```
>> w =  [4   2   6];
>> d =  12./ w
d =
   3     6     2
```

Функция мәндерінің кестелерін құру

Функцияның берілген нүктелердегі мәнін есептеу керек дейік. Есеп екі этаппен шығарылады.

$$y(x) = \frac{sin^2 x}{1 + cosx} + e^{-x} \cdot lnx$$

Берілген нүкте координаторларынан тұратын *x* вектор-жол құрылады.

x векторының әр элементі үшін *y(x)* функциясының мәнін есептеу және шыққан нәтижелерді *y* вектор-жолына жазу керек. Әрбір *x* вектор-жолының элементеріне байланысты функция мәндерін табу керек, сол себептен барлық амалдар элементтер бойынша орындалуы қажет.

```
>> x = [0.2:0.1:1.2];
>> y = sin(x).^2./(1 + cos(x)) + exp(-x).*log(x)
y =
Columns 1 through 11
-1.2978 -0.8473 -0.5353 -0.2980 -0.1057 0.0580 0.2030 0.3356 0.4597
0.5781 0.6926
```

Тапсырмалар

1. *x* айнымалысын енгізіп, *cos(x) + 2sin(5x)* функциясының мәнін есептеңіз. Ағымдағы жұмысты мәтіндік файлда, ал айнымалыларды *mat*-файлда сақтаңыз.
2. Векторларды енгізіп, олармен қосу және шегеріп тастау амалдарын орындаңыз. *ndims* және *size* функциялары арқылы массивтің өлшемі мен ұзындығын анықтаңыз.
3. Мәліметтердің өңдеу функцияларын векторларға қолдану.
4. Векторлармен элементтері бойынша амалдарды орындау:
   v1 = [2 –3 4 1];
   v2 = [7 5 –6 9];
5. Үш вектор берілген *a(1, 2), b(-5,- 1), c(-1, 3).*

2a + 3b – c, 16a + 5b – 9c векторларының координаталарын табыңыз.

6. Төрт вектор берілген *a(3, 0, -2), b(1, 2, -5), c(-1, 1, 1), d(8, 4, 1).*

–5a + b – 6c + d, 3a – b – c – d векторларының координаталарын табыңыз.

7. *a* және *b* векторларының скалярлық көбейтіндісін табыңыз, егер:
   $|a| = 3,\ |b| = 1,\ (a, b) = 45°;$
   $|a| = 6,\ |b| = 7,\ (a, b) = 120°;$
   $|a| = 4,\ |b| = 2,\ (a, b) = 90°;$
   $|a| = 2,\ |b| = 1,\ (a, b) = 30°;$
8. $|a| = 3,\ |b| = 2,\ (a, b) = 150°$. Болғандағы $|a|2 - 3(a,b) + 5|b|2$ өрнегінің мәнін есептеңіз.
9. Өзінің координаторларымен берілген *a* және *b* векторларының скалярлық көбейтіндісін табыңыз.
   a(4,-1), b(-1,-7);
   a(2,1), b(1,-3);
   a(1,2), b(-4,2).
10. Өзінің координаторларымен берілген *a* және *b* векторларының арасындағы бұрышты анықтаңыз.
    a(1,2), b(2,4);

    a(1,2), b(4,2);
    a(1,2), b(-2,1);
    a(1.-1), b(-4,2);
    a(2,-1), b(-4,2).
11. Өзінің координаторларымен берілген *a* және *b* нүктелері арасындағы қашықтықты табыңыз.
    A(-1,2), B(5,10);
    A(3,-2), B(3,3);
    A(1,2), B(1,2).
12. Өзінің координаторларымен берілген *a* және *b* векторларының скалярлық көбейтіндісін табыңыз.
    a(3,2,-5), b(10,1,2);
    a(1,0,3), b(-4,15, 1);
    a(2,1,5), b(7,-9,-1).

Екінші сабақ. Матрицалар

Сабақтың жоспары

1. Матрицаларды енгізу. Матрицаларды әртүрлі жолдармен көрсету.
2. Матрицаларды біріктіру. Жолдар мен бағандарды жою.
3. Матрицалармен орындалатын қарапайым амалдар.
4. Матрицалық амалдар.
5. Деректерді өңдеу функцияларын матрицаларға қолдану.
6. Матрицалардың элементтері бойынша амалдар орындау

Матрицаларды енгізу

Матрицаны енгізу кезінде келесі негізгі шарттарды орындау керек:

- жол элементтерін бос орын белгісімен немесе үтірмен ажырата жазу;
- нүкте үтірді әр жолдың аяқталғанын көрсету үшін қолдану, ал барлық элемент тізімін аяқтауды тік жақшамен *[ ]* белгілеу;
- массив элементтеріне қатынасуда жай жақшаны пайдалану *( )*.

Matlab-та матрицаларды бірнеше әдістермен енгізуге болады:

- элементтер тізімін толық енгізу;
- сыртқы файлдан матрицаны жүктеу;
- ішкі функцияларды қолдана отырып, матрицаны кездейсоқ құру;
- матрицаны *М*-файлдағы өзіңіздің функцияңыз арқылы құру.

Матрицаны нақты анықтау

>>A = [3 1 1; 4 5 6]

A =

3 1 1

4 5 6

Жұмыс ортасының айнымалыларын *who* және *whos* командалары арқылы көруге болады. Бұл жағдайда төмендегідей хабарлама беріледі

A 2×3 48 double array

Яғни, *A 48* байттан тұратын *2×3* өлшемді екіөлшемді матрицаны көрсетеді.

Сандарды экспоненциалды түрде (мысалы, *2.34e-9*) ендіруде бос орын белгісі қолданылмайды. Үлкен матрицаларды ендіруді *М*-файлы арқылы орындау ыңғайлы, өйткені онда қателерді тез табуға және жоюға мүмкіндік бар. *М*-файлдарын құру алтыншы сабақта қарастырылады.

Матрицаның жеке элементтері мен векторларға сілтеме жай жақшадағы индекс арқылы орындалады. *A* матрицасының *i* жолындағы және *j* бағанындағы элементін *A(i,j)* түрінде көрсетеді. Мысалы, *A(2,3)=6*

A матрицасының элементтеріне жалғыз *к* индексін пайдаланып та сілтеме жасауға болады *A(k)*. Бұл жағдайда осы матрица бастапқы матрицаның бағаналарынан құрылған бір вектор – бағана түрінде қарастырылады. Келтірілген матрицада 6-шы элемент *A(2,3)*, немесе *A(6)* сияқты шақырылуы мүмкін. Қайсысы болса да дұрыс.

Матрицаның кездейсоқ құрылуы

Matlab негізгі матрицаларды құрайтын арнайы функциялардан тұрады. Олар:
zeros – матрицаның барлық элементтері – нөлдер;
eye – бірлік матрица;
ones – матрицаның барлық элементтері – бірлер;
rand – кездейсоқ элементтерді бірыңғай, біркелкі орналастыру;
randn – кездейсоқ элементтерді кәдімгідей орналастыру.

Мысалға, *rand(n)* командасы *mxn* квадраттық матрицасын құрайды. Оның әрбір элементі – *[0 1]* диапазонында біркелкі орналасатын кездейсоқ сан, ал *rand(m,n)* командасы *mxn* өлшемді тура осындай матрицаны құрайды. Сонымен қатар матрицалар *for* (сегізінші сабақты қараңыз) циклінің көмегімен құрылуы мүмкін.

Кейбір мысалдар:

```
>> Z = zeros(2,3)
Z =
    0 0 0
    0 0 0
>> Q = eye(3,4)
Q =
    1 0 0 0
    0 1 0 0
    0 0 1 0
>> W = 5*ones(3,3)
W =
     5 5 5
     5 5 5
     5 5 5
>> N = fix(10*rand(1,9))
N =
     9 2 6 4 8 7 4 0 8
>> R = randn(4,4)
R =
    -0.4326   -1.1465   0.3273   -0.5883
    -1.6656    1.1909   0.1746    2.1832
```

0.1253 1.1892 -0.1867 -0.1364
0.2877 -0.0376 0.7258 0.1139

Көп жағдайда диагоналды матрица құру қажеттілігі туындайды, яғни барлық диагоналдан тыс элементтер нөлге тең болатын матрицалар. *Diag* функциясы вектордан немесе вектор – жолдан тұратын диагоналды матрицаны құрайды және оның элементтерін матрица диагоналы бойынша орналастырады:

>> d = [1; 2; 3; 4];
>> D = diag(d)
D =
1 0 0 0
0 2 0 0
0 0 3 0
0 0 0 4

Басты емес, ал қосымша диагоналды толтыру үшін, екі аргументті *diag* функциясын шақыру мүмкіндігі алдын-ала қарастырылған. Бұл жағдайда, екінші аргумент қосымша диагоналдың бастыдан қаншалықты артта қалғанын білдіреді, ал оның таңбасы бағытты білдіреді, қосу – жоғарыға, алу – төменге қарай:

>> d = [1; 2];
>> D = diag(d, 2)
D =
0 0 1 0
0 0 0 2
0 0 0 0
0 0 0 0
>> D = diag(d, -2)
D =
0 0 0 0
0 0 0 0
1 0 0 0
0 2 0 0

diag функциясы матрица диагоналінің векторға бейнеленуі үшін де қызмет етеді, мысалы

>> A = [10 1 2; 1 20 3; 2 3 30]
A =
10 1 2
1 20 3
2 3 30
>> d = diag(A)

d =
10
20
30

Matlab-тың кез келген көлемді сиқырлы квадрат құрайтын функциясы бар (сиқырлы квадратта жол элементінің қосындысы баған элементерінің қосындысына тең және ол басты және қосымша диагоналдар элементтерінің қосындысына тең). Сондықтан, бұл функция *magic* деп аталады.

>> B = magic(4)
B =
16 2 3 13
5 11 10 8
9 7 6 12
4 14 15 1

Матрицаларды жүктеу

Load командасы құрамында *Matlab*-та алдында құрылған матрицалар бар екілік файлдарды немесе құрамында сандық мәліметтер бар мәтіндік файлдарды есептейді. Мәтіндік файлдар арасы бос орын белгісімен бөлінген, әр жолдағы элементтер саны бірдей тікбұрышты сандар кестесі түрінде құрастырылуы керек. *Matlab*-тан тыс төрт жолдан тұратын мәтіндік файл құрылды дейік:

16 2 3 13
5 11 10 8
9 7 6 12
4 14 15 1

Ол *Matlab*-тың *work* ағымдағы каталогында *magic.txt* деген атпен сақталған. Онда:
>>load magik.txt
бұл файлды оқиды және құрамында осы матрица бар *magic* айнымалысын құрайды. Берілген матрицаның мәнін кейбір айнымалыға беруге болады:
>>D = load('magik.txt');
>>d = D;
мәтіндік файлда жұмыс облысының мәліметтерін сақтау үшін, *save* командасы қолданылады:
>> save 'magcopy' d –ascii
осы команда бойынша *d* айнымалысының мәні *'magcopy'* файлында сақталады. *Ascii* параметрі мәтіндік форматтағы жазуды білдіреді.

Нәтижені екінші дәрежелі дәлдікті файлға жазу үшін келесі команданы қолданады

>> save 'magcopy' d –ascii –double

Біріктіру

Біріктіру дегеніміз – кіші матрицаларды үлкен матрица құру үшін біріктіру. Қос тік жақша – біріктіру операторы. *D* матрицасы арқылы (сиқырлы квадрат *4×4*) жаңа матрицаны құруға болады.

>> B = [D; D + 32; D + 48; D + 16]

Нәтижесінде *8×8* матрицасы төрт матрицалардың бірігуінен алынады:

16	2	3	13	48	34	35	45
5	11	10	8	37	43	42	40
9	7	6	12	41	39	38	44
4	14	15	1	36	46	47	33
64	50	51	61	32	18	19	29
53	59	58	56	21	27	26	24
57	55	54	60	25	23	22	28
52	62	63	49	20	30	31	17

Жолдар мен бағандарды жою

Матрицалардың жолдары мен бағандарын қос бос тік жақша арқылы жоюға болады. Матрица келесі түрде болсын:

>> X = D;

және *X* матрицасының екінші бағанын жою керек. Ол үшін төмендегідей іс әрекет жасау керек:

>> X(:,2) = [ ]

Бұл амал *X* матрицасын төмендегідей түрге келтіріледі:

X =

```
    16    3   13
     5   10    8
     9    6   12
     4   15    1
```

Қосу, азайту, көбейту, транспонирлеу және дәрежеге шығару

Матрицалық амалдарды орындаған кезде, қосу не алуда матрицалар өлшемі бірдей, ал көбейтуде бірінші матрицаның баған саны екінші матрицаның жол санына тең болуы керек. Матрицаны қосу және алу, векторлар мен сандар сияқты қосу және алу таңбалары арқылы орындалады.

>>A = [1 2 3;4 5 6];
>>B = [5 6 2;8 9 0];
>>S = A + B

S =

```
    6    8    5
   12   14    6
```

>>R = A – B

R =

```
   -4   -4    1
   -4   -4    6
```

болсын.

Бұл жерде өлшемдердің сай келуін қадағалау керек, ондай болмаған жағдайда қателік туғандығы туралы хабарлама беріледі:

>> S = A + B
??? Error using ==> + MATrix dimensions must agree.

Матрицаларды көбейту үшін жұлдызша белгісі қолданылады:

>> C = [1 2; 3 4; 1 2];
*>> P = A*C*

P =

```
   10   16
   25   40
```

Матрицаны санға көбейту үшін де жұлдызша қолданылады, сонымен қатар не оң жақтан немесе сол жақтан көбейте беруге болады:

*>> P = A*3*

P =

```
    3    6    9
   12   15   18
```

*>> P = 3*A*

P =

```
    3    6    9
   12   15   18
```

Матрицаның вектор сияқты жолдары мен бағандарының орнын ауыстырып түрлендіру .′ таңбасы арқылы жүзеге асады, ′ таңбасы кешенді сәйкестендіруді білдіреді. Нақты матрицалар үшін бұл амалдар бірдей нәтижелер береді:

>> B′

ans =

```
   58
   69
   20
```

>> B.′

ans =

```
   58
   69
   20
```

Қоснүкте ":" – бұл *Matlab* амалдарының ішіндегі ең маңыздысы болып саналады. *A(1:k,j)* – бұл *A* матрицасының *j*-ші бағанасының алғашқы *k* элементтері. *sum(A(1:2,3))* функция үшінші бағанның алғашқы екі жолының элементтерінің қосындысын есептейді. Қоснүкте матрица бағанының және жолының барлық элементтеріне катынасуға мімкіндік берсе, ал *end* сөзі соңғы бағанға немесе жолға катынасуға мүмкіндік береді. Сонымен *sum(A(:,end)) A* матрицасының соңғы бағанындағы элементтер қосындысын есептейді.

Кешенді сандары бар матрицалардың жолдары мен бағандарын ауыстырып түрлендіру және сәйкестендіру арқылы әртүрлі матрицалар құрылады:

```
>> K = [1 - i,   2 + 3i;   3 - 5i,   1 - 9i]
K =
     1.0000 - 1.0000i        2.0000 + 3.0000i
     3.0000 - 5.0000i        1.0000 - 9.0000i
>> K'
ans =
     1.0000 + 1.0000i     3.0000 + 5.0000i
     2.0000 - 3.0000i     1.0000 + 9.0000i
>> K.'
ans =
     1.0000 - 1.0000i    3.0000 - 5.0000i
     2.0000 + 3.0000i    1.0000 - 9.0000i
```

D квадрат матрицасын бүтін дәрежеге шығару ^ операторын қолдану арқылы жүзеге асырылады:

```
>> D2 = D^2
```

Алынған нәтижені тексеру үшін, матрицаны бір-біріне көбейтіп көріңіз. Матрицалармен амалдар орындаған кезде міндетті түрде амалдар үстемділіктерін ескеру керек: алғашқыда матрицасының жолдары мен бағандарын ауыстырып түрлендіруді орындау керек, содан кейін оны дәрежеге шығару; ары карай көбейту, косу және азайту ең соңғы кезекте орындалады.

Матрица мен векторды көбейту

Мысалы:

$$[1\ 3\ {-2}]\begin{pmatrix} 2 & 0 & 1 \\ -4 & 8 & -1 \\ 0 & 9 & 2 \end{pmatrix}\begin{bmatrix} -8 \\ 3 \\ 4 \end{bmatrix}$$

оны келесідей түрде жүзеге асыруға болады:

```
>> a = [1  3 -2];
>> B = [2  0  1;  -4  8 -1;  0  9  2];
```



```
>> c = [-8;3;4];
>> a*B*c
ans = 74
```

Сызықты теңдеулер жүйесін шешу

Үш белгісізі бар үш теңдеулер жүйесін шешу керек болсын:

$1.2x_1 + 0.3x_2 - 0.2x_3 = 1.3;$

$0.5x_1 + 2.1x_2 + 1.3x_3 = 3.9;$

$-0.9x_1 + 0.7x_2 + 5.6x_3 = 5.4$

Жүйе матрицасын *A* массивіне енгізіңіз, оң жақ бөліктегі вектор үшін *b* массивін колданыңыз. Жүйені \ символын колданып шығарыңыз.

```
>> x = A\b
1. 0 000
1. 0000
1. 0000
```

A-ны *x*-ке көбейту арқылы жауаптың дұрыстығын тексеріңіз.

Деректерді өңдеу функцияларын матрицаларға қолдану

Матрицаға қатысты қолданылған *sum* функциясын *Matlab*-та ұзындығы матрица бағандарының санына тең, ал әрбір элементі матрицаның сәйкес бағанының элементтерінің қосындысына тең болатындай вектор-жолды есептеуге болады. Мысалы:

```
>> M = [1  -2  -4
          3 -6    4
          2  -2   0];
>> s = sum(M)
   6    -10    0
```

Басқаша анықталмаса, *sum* функция бағандар бойынша қосындыны есептейді және ол 2-ші индексті тұрақты қалдырып, массивтің бірінші индексін өзгертеді. Жолдар бойынша қосындыны табу үшін, *sum* функциясын екі аргументпен шақыру керек және бұл жерде қосындысы табылып отырған индексті көрсету керек.

```
>> s2 =  sum(M, 2)
s2 =
      -5
       1
       0
```

Prod функциясы да дәл осылайша жұмыс істейді:

```
>> p = prod(M)
p =
     6   -24    0
```

>> p2 = prod(M, 2)
p2 =
8
72
0

Soft функциясы матрицаның әр баған элементтерін өсу ретімен орналастырады. *Soft* функциясын 2-ші аргументі екіге тең болатындай етіп шақыру, жолдар элементтерін реттеуге мүмкіндік береді:

>> MC = sort(M)
MC =

1	*-6*	*-4*
2	*-2*	*0*
3	*-2*	*4*

>> MR = sort(M, 2)
MR =

-4	*-2*	*1*
-6	*3*	*4*
-2	*0*	*2*

max және *min* функциялары матрицаның бағандарының сәйкесінше максималды және минималды элементтерінен тұратын вектор-жолдарды есептейді.

>> mx = max(M)
mx =
3 -2 4
>> mn = min (M)
mn =
1 -6 -4

Элементтердің максималды немесе минималды мәндері ғана емес, сонымен қатар олардың бағандарындағы нөмірлерін білу үшін *max* және *min* функцияларын екі шығарылатын параметрмен шақыру керек:

>> [mx, k] = max (M)
mx =
3 -2 4
k =
2 1 2
>> [mn, n] = min(M)
mn =
1 -6 -4
n =
1 2 1

Егер *min* және *max* функцияларының аргументтері сан болса, онда осы сандардың максималды немесе минималды мәні қайтарылады.

Егер максимумды немесе минимумды матрицаның бағандары бойынша емес, жолдары бойынша табу керек болса, онда ол екінші аргументі бос-массив болып шақырылады:

>> max = max(M, [ ], 2)
max =
1
4
2

Матрицалық мәліметтердің өңделуі туралы көбірек мәліметтерді алу үшін, *help datafun* командасын колданып, одан кейін керекті функцияға байланысты ақпаратты қарауға болады, мысалы *help max*.

Матрицаның анықтауышы *det* функциясында, рангісі – *rank* функциясында орналасады.

Матрицалардың элементтері бойынша амалдар орындау.
Екі матрица берілген:

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

Бір матрица элементтерін 2-ші матрица элементтеріне көбейту үшін.* операторы қолданады.

*>> C = A.*B*
C =

-2	*10*	*-8*
21	*-12*	*-45*

Бірінші матрица элементтерін сәйкесінше 2-ші матрица элементтеріне бөлу үшін ./, ал 2-ші матрица элементтерін 1-ші матрица элементтеріне бөлу үшін .\ қолданады.

>> R1 = A./B
R1 =
-2.0000 2.5000 -0.1250
0.4286 -1.3333 -1.8000
>> R2 = A.\B
R2 =
-0.5000 0.4000 -8.0000
2.3333 -0.7500 -0.5556

Элементтерді дәрежеге шығару ^ операторының көмегімен жүзеге асырылады.

>> P = A.^2
P =
4 25 1
9 16 81

Тапсырмалар

1. A матрицасын енгізіңіз.
2. A'-ті табыңыз.
3. Қайтаруды *inv(A)* орындаңыз, теңдікті тексеріңіз.
4. Диагональді матрицаны құрастырыңыз, оның диагоналін бөліп көрсетіңіз.
5. Матрицаны файлдарға толтырыңыз және жазыңыз.

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 2 \\ 0 & 1 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 1 & 0 & 0 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 1 & 0 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 & 1 & 2 & 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 & 3 & -4 & 0 & 0 & 0 & 0 \\ 3 & 3 & 3 & 3 & 3 & 0 & -4 & 0 & 0 & 0 \\ 3 & 3 & 3 & 3 & 3 & 0 & 0 & -4 & 0 & 0 \\ 3 & 3 & 3 & 3 & 3 & 0 & 0 & 0 & -4 & 0 \\ 3 & 3 & 3 & 3 & 3 & 0 & 0 & 0 & 0 & -4 \end{bmatrix},$$

$$G = \begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 \end{bmatrix}.$$

6. Сиқырлы квадрат құрыңыз. Қасиеттерді тексеріңіз: кез-келген жолдың, бағанның, диоганаль элементтерінің қосындысы тең келуі керек.
7. Кездейсоқ түрде таңдалған квадраттық матрицаны сандармен толтырыңыз, онан 5-тен құралған матрицаны алып тастаңыз. Матрица нормаларын (№11) есептеңіз.
8. Матрицаның анықтауышын, рангісін табыңыз.
9. Матрицаның минималды элементін табыңыз.
10. $Ax = b$ сызықты теңдеулер жүйесі берілген. **.txt* файылынан A және b векторын енгізіңіз. Нәтижелерді жаңа файлда сақтаңыз.
11. a_{ik} элементтері бар n өлшемді A квадраттық матрицасы берілген, төменде келтірілген шамаларды (матрица нормаларын) есептеңіз:

$$p = \max_{1\le i\le n} \sum_{k=1}^{n} |a_{ik}|, \quad q = \max_{1\le k\le n} \sum_{i=1}^{n} |a_{ik}|, \quad r = \left(\sum_{i,k=1}^{n} a_{ik}^{\;2} \right)^{1/2}, \quad s = \sum_{i,k=1}^{n} |a_{ik}|.$$

12. 0-ден 10-ға дейін кездейсоқ бүтін сандардан құрылған 4×4 өлшемді квадратты матрицаны құрыңыз, одан 5-тен құралған матрицаны шегеріңіз және алынған матрицаның нормасын есептеңіз.

Үшінші сабақ. Функциялардың графиктерін тұрғызу

Сабақтың жоспары

1. Бір айнымалыдан тәуелді функцияның графигін тұрғызу.
2. Графиктерді жеке терезелерге шығару.
3. Бірнеше графиктерді бір өс бойына шығару.
4. Бірнеше графиктерді бір графикалық терезеде тұрғызу.
5. *Fplot* функциясы.
6. Функциялардың графиктерін полярлық координаталар жүйесінде тұрғызу.

Matlab жазықтықтағы графиктерден басқа үш өлшемді торлы қабатты, сонымен қатар қозғалмалы графиктерді және анимацияны құрай алады.

Matlab графиктерді құру үшін қолданылатын жоғары деңгейдегі командалар жинағын ұсынады. Олар *plot*, *title*, *axis*, *text,hist*, *contour* және бірқатар басқа командалар. Жоғары деңгейлі графиктер командасы графикалық объектілердің қасиеттерін автоматты түрде анықтайды және қажетті координаттар, түстер палитрасы және тағы басқа жүйесінде графиктердің өңделуін қамтамасыз етеді.

Мысалы, бір нақты айнымалыдан тәуелді функцияның графигін тұрғызу керек дейік. Келесідей өрнектер:

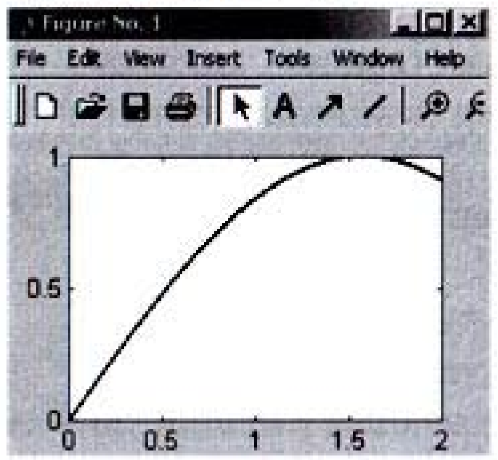
>> x = 0:0.01:2;

>> y = sin(x);

берілген аргументтер жинағы үшін *sin* функциясының мәндерінен тұратын *y* массивін есептейді. Осыдан кейін

>> plot(x, y)

функциясын шақырту арқылы оның графигін тұрғызуға болады (3.1 суретін қараңыз).



3.1-сурет. *y = sin(x)* функциясының графигі

Matlab тақырыбында *figure* (фигура) деген сөзі бар арнайы графикалық терезелерде графикалық объектілерді көрсетеді.

Дисплейдің экранынан бірінші графикалық терезені алып тастамай, клавиатурадан келесідей формуланы енгізіңіз

>> x = 0:0.01:2;

>> z = cos(x);

>> plot(x, z)

содан кейін, дәл осы графиктік терезеде функцияның жаңа графигі шығады.

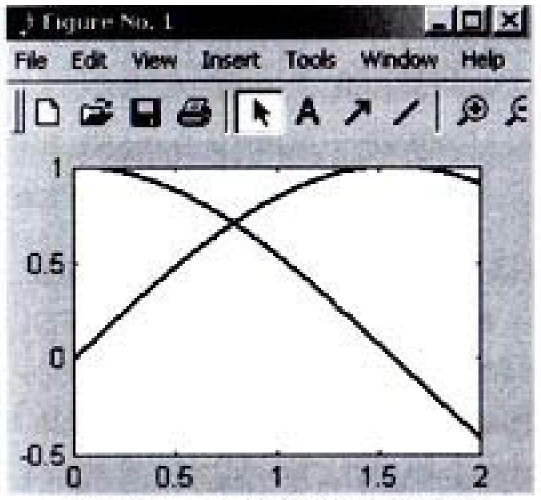
Бір суретте бірнеше графиктерді көрсетудің екі әдісі бар. Бірінші әдіске *hold on* командасын қолдану жатады. Ол ағымдағы графикті

тұрақтандырады, сонда келесі қисықтар осы графикке орналастырылады. *hold off* командасы *plot* командасының әрбір шақырылған кезінде жаңа суреттің осы бетте шығуына, яғни алдыңғы графиктің өшірілуіне алып келеді.

>> hold on

>> plot(x, z)

Нәтижесінде келесідей сурет шығады (3.2 суретін қараңыз).



3.2-сурет. *sin(x)* және *cos(x)* функцияларының графиктері

Тура осы нәтижені *plot* функциясын аргументтердің келесідей *x* айнымалысы, бірінші функция, *x* айнымалысы, екінші функция және тағы басқа ретімен пайдаланып алуға болады:

>> x = 0:0.01:2;

>> y = sin(x);

>> z = cos(x);

>> plot(x, y, x, z)

Бұл әдістің тағы да бір артықшылығы әр түрлі графиктерді автоматты түрде әр түрлі түспен тұрғызады.

Егер сонымен бірге әр графикті бөлек шығара отырып, нәтижелерді көрсету керек болса, онда оны тағы да екі әдіспен жасауға болады: бірінші әдіс – оларды әр түрлі графиктік терезеде тұрғызу. Жаңа графиктік терезені құру үшін *figure* командасы қолданылады. Мысалы, *exp(x)* функциясының графигін жаңа терезеде көрсету керек дейік. Ол үшін келесідей командаларды жазу жеткілікті:

>> w = exp(x);

>> figure;

>> plot(x, w)

Координаттар осінің диапазондарының шиеленіссіз бірнеше графиктерді көрсетудің екінші әдісі *subplot* функциясын қолдану болып табылады. Бұл функция графиктік ақпаратты шығару облысын бірнеше кішігірім облыстарға бөлуге мүмкіндік береді. Олардың әрқайсысына әр түрлі функциялардың графиктерін шығаруға болады.

Мысалы, *sin*, *cos* және *exp* функциялары үшін бірінші екі функцияның графигін бірінші кішігірім аймақта, ал үшінші функцияның графигін екінші кішігірім аймақта, бір графикалық терезеде тұрғызу керек дейік (3.3-суреті). Ол үшін келесідей командаларды жазу жеткілікті:

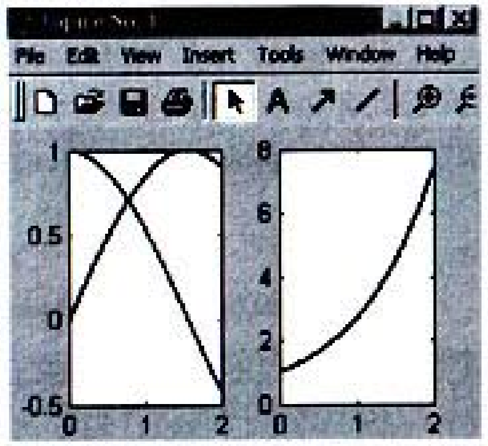
>> subplot (1, 2, 1);

>> plot (x, y, x, z);

>> subplot (1, 2, 2);

>> plot (x, w)

Осы кішігірім аймақтардың координаттар осіндегі айнымалылардың өзгеру диапазондары бір-біріне тәуелсіз. *subplot* командасында матрица түрінде кішігірім графиктер бар және ол үш параметрмен қолданылады: *subplot (i, j, n)*, бұл жерде *i* және *j* – вертикаль және горизонталь бойынша кішігірім графиктердің саны, ал *n* – ағымдағы графикке айналдырылатын кішігірім графиктің нөмірі.



3.3-сурет. *subplot* командасын қолдану мысалы

Нөмір сол жақ үстіңгі бұрыштан бастап жолдар бойынша саналады.

Мысалы, *subplot (3, 2, 4)* командасы алты кішігірім графиктердің бар екендігін білдіреді және төртіншіні ағымдағы график етіп белгілейді. Егер жалғыз график үшін бір немесе екі координаттар осі бойымен айнымалылардың өзгеруінің диапазондары өте үлкен болса, онда логарифмдік масштабтардағы графиктерді тұрғызу функциясын қолдануға болады. Ол үшін *semilogx*, *semilogy* және *loglog* функциялары пайдаланылады. Бұл функцияларды қолдану жөніндегі толық мәліметті әр кезде *Matlab* жүйесінің командалық терезесінде орындалатын *help <функция аты>* командасының көмегімен алуға болады.

fplot функциясы

fplot функциясы *y*-ті *x*-ке қатысты есептеуге қарағанда және одан кейінгі осы қисықтың *plot* функциясы арқылы бейнеленуімен салыстыра отырып альтернативті бейнелеу мүмкіндігін ұсынады. Бұл функцияға қажет функцияны *f(x)* түріндегі суреттейтін жолды жіберіп отыру керек. *f(x)*-ті суреттейтін жол *Matlab*-та қолданылатын кез келген амал және/немесе функция болуы мүмкін.

Мысалы, *x*-тің 0-ден 5π-ге дейінгі диапазонында $y = sin(x)cos(2x)$ қисығын салу үшін

```
>> fplot ('sin(x) .*cos(2*x)', [0 5*pi])
```

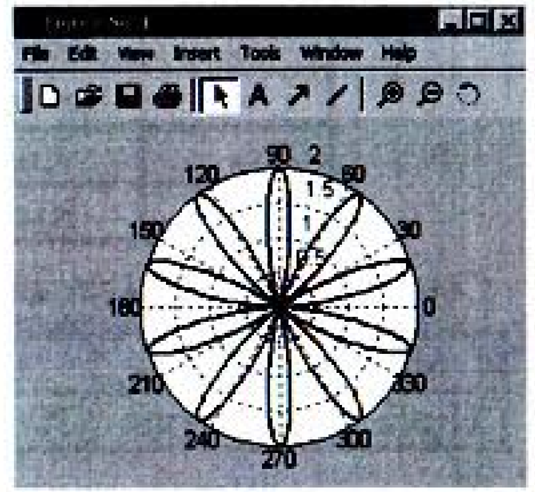
функциясын шақыру керек.

fplot функциясының тағы да екі қосымша аргументі бар. Олардың бірі – сызықтың типі мен түсін көрсететін (төменде қараңыз) жол, ал екіншісі – дәлдікті көрсетеді. Басқаша көрсетілмесе, дәлдік 2×10^{-3}-не тең және ол сызықтық интерполяциядан қателік шегі осы берілген дәлдіктен аспайтындай интервалды бөлу нүктелерінің санын анықтайды.

Функциялардың графиктерін полярлық координаталар жүйесінде тұрғызу

$r = sin(3\varphi)$ функциясының графигін полярлық координаталар жүйесінде тұрғызу керек дейік. Бұл тапсырманы орындау үшін *polar* командасы қолданылады. Ол екі аргументтен радианда берілген *phi* бұрышынан және *r* радиусынан тәуелді. Графикті тұрғызу үшін (3.4-сурет), келесі командаларды жазу керек:

```
>> phi = 0:0.01:2*pi;
>> r = sin(3*phi);
>> polar(phi, r)
```



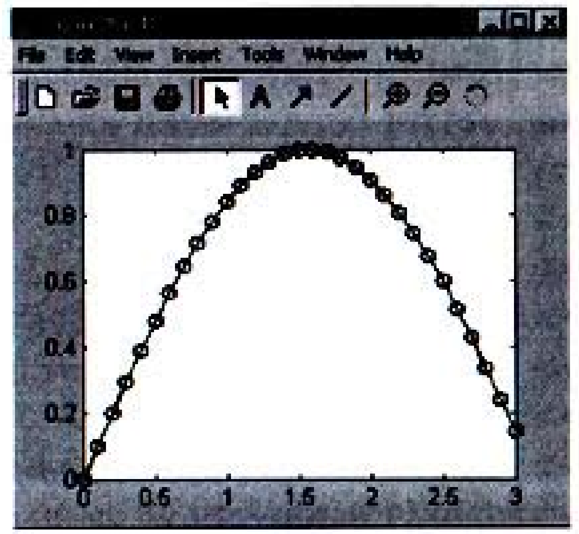
3.4-сурет. $r = sin\ (3\varphi)$ функциясының графигін тұрғызу

Графиктерді және графиктік терезелерді безендіру

Графиктердің сыртқы түрін басқарумен байланысты сызықтардың түсі мен стилін белгілеу, сонымен қатар графиктік терезе ішінде әртүрлі жазуларды орналастыру ортаның қосымша мүмкіндіктері болып табылады (3.5-сурет).

Мысалға:

```
>> x = 0:0.1:3;
>> y = sin(x);
>> plot(x, y, 'r-' , x, y, 'ko')
```



3.5-сурет. *sin(t)* функциясының графигі

командалары график сызығын қызыл түспен бейнелеп, дискретті есептелетін нүктелерде қара жұлдызшалар суреттелуіне мүмкіндік береді.

Бұл жерде *plot* функциясы екі стильде бір функцияның графигін екі рет тұрғызады. Берілген қосымша мүмкіндіктерді *fplot* қатысты қолдануға болады. Жалпы жағдайда, *plot(x1, y1, s1, x2, y2, s2,...)* функциясы бір графиктік терезеде *s1, s2,...* стилдері арқылы *y1 (x1), y2 (x2),...*, функцияларының бірнеше графиктерін біріктіруге мүмкіндік береді.

s1, s2,... стилдері өзіндік тырнақшаға алынған үш символдық маркерді теру түрінде беріледі. Осы маркерлердің біреуі сызық типін анықтайды:

Маркер	-	- -	:	-.
Сызық типі	Үздіксіз	Үзік сызық	Қос нүктелі	Сызық нүкте

Басқа маркер түсті анықтайды:

Маркер	Сызық түсі	Маркер	Сызық түсі
c	Көгілдір	*g*	Жасыл
m	Қанық көк	*b*	Көк
y	Сары	*w*	Ак
r	Қызыл	*k*	Қара

Соңғы маркер қойылатын "нүкте" типін анықтайды:

Маркер	·	+	*	○	×
Нүкте типі	Нүкте	Плюс	Жұлдызша	Дөңгелек	Крестик

Графиктерді безендіру элементтеріне өстерге, координаталық торға, бас тақырыпқа және түсініктемеге қолтаңба қою жатады. Торды түсіру *grid on*, өстерге қолтаңба қою *xlabel*, *ylabel*, ал бас тақырып *title* командасының көмегімен атқарылады. Бір остің бойында бірнеше графиктің болуы сызықтар туралы мәліметтері бар *legend* командасымен түсініктемені енгізуді қажет етеді. Келесі командалар қажетті мәліметтің барлығымен қамтамасыз етілген 3.6-суретінде көрсетілген тәуліктік температураның өзгеру графигін шығарады.

```
>> time = [0 4 7 9 10 11 12 13 13.5 14 14.5 15 16 17 18 20 22];
>> temp1 = [14 15 14 16 18 17 20 22 24 28 25 20 16 13 13 14 13];
>> temp2 = [12 13 13 14 16 18 20 20 23 25 25 20 16 12 12 11 10];
>> plot(time, temp1, 'ro-', time, temp2, 'go-')
>> grid on
>> title('Суточные температуры')
>> xlabel('Время (час.)')
>> ylabel('Температура (С)'), legend('10 мая', '11 мая')
```

Түсініктемелерді енгізгенде *legend* командасының аргументтер тәртібі мен саны графиктегі сызықтарға сәйкес келуі керек. *Legend*-тің соңғы қосымша аргументі графиктік терезедегі түсініктеменің орналасу жағдайы болуы мүмкін:

-1 – графиктен тыс графиктік терезенің оң жақ үстіңгі бұрышында;

0 – графиктердің өзін неғұрлым аз жабатындай графиктің ішіндегі ең тиімді орын таңдалады;

1 – графиктің оң жақ үстіңгі бұрышында (үнсіздік бойынша пайдаланылады);

2 – графиктің сол жақ үстіңгі бұрышында;

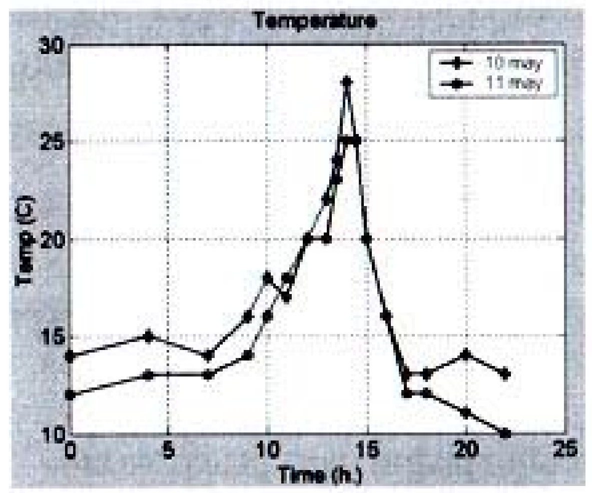
3 – графиктің оң жақ астыңғы бұрышында;

4 – графиктің сол жақ астыңғы бұрышында.

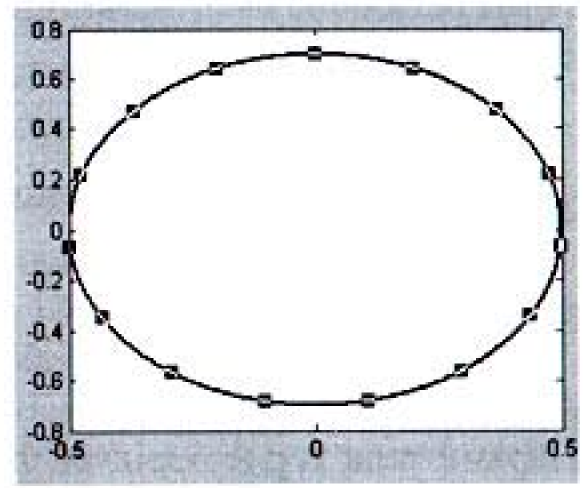
Параметрлік түрде берілген функцияны тұрғызу үшін ең алдымен аргумент мәнінің векторын кездейсоқ көрсету қажет. Содан кейін

функцияның мәнін есептеп, оны векторға жазу керек, осы вектор *plot* функцияның аргументтері ретінде пайдаланылады. 3.7-суретінде келтірілген $t \in [0, 2\pi]$ үшін $x(t) = 0.6sint$, $y(t) = 0.8cost$ (эллипс) функциясының графигі, келесі командалардың көмегімен жасалады:

```
>> t = [0:0.01:2*pi];
>> x = 0.5*sin(t);
>> y = 0.7*cos(t);
>> plot(x, y)
```



3.6-сурет. Тәуліктік температура өзгерісінің графигі



3.7-сурет. Параметрлік түрде берілген эллипстің графигі

Тапсырмалар

1. Параметрлік түрде берілген функцияның графигін тұрғызыңыз

$$\begin{cases} x = a_1 \cdot cos(\omega_1 t) \\ y = a_2 \cdot cos(\omega_2 t) \end{cases}.$$

   Егер $\omega 1/\omega 2$ – рационалды сан болса, онда бұндай қисық Лиссажу фигурасы деп аталады. Операторларды қосымша терусіз (↑ батырмасы), олардың қайталап орындалуын пайдаланып, $\omega 1/\omega 2$ әртүрлі қатынасы үшін және әртүрлі *a1*, *a2* амплитуда мәндері үшін Лиссажу фигурасының есептелуін және тұрғызылуын орындаңыз.

2. Косинус аргументтеріне $\omega 1$, $\omega 2$, бастапқы фазаларды қосу, яғни $cos(\omega_1 t)$-ті $cos(\omega_1 t + \varphi_1)$-мен, ал $cos(\omega_2 t)$-ті $cos(\omega_2 t + \varphi_2)$-мен ауыстырып, графиктерді тұрғызыңыз.

3. Параметрлік түрде берілген функция графигін тұрғызыңыз

$$\begin{cases} x = 4e^{-0.05t} sint \\ y = 0.2e^{-0.1t} sin2t \end{cases}.$$

4. Коэффициенттердің әр түрлі мәндерін қоя отырып, тербелмелі қозғалыс графигін тұрғызу $y(t) = a_1 cos(\omega_1 t) + a_2 cos(\omega_2 t)$.

5. $\omega 1 << \omega 2$ кезінде соғудың басқа түрін бақылауға болады. Бұл

жағдайда тербелістің накты мәнін көру үшін, соғудың есептелу уақытын 10-20 есе көбейткен жөн.

6. Екінші және үшінші гармониканың тербелістің формасына әсерін зерттеу. Ол үшін $y(t)$: $y(t) = a_1cos(\omega_1 t) + a_2cos(\omega_2 t + \varphi_2) + a_3cos(\omega_3 t + \varphi_3)$. функциясын таңдау керек және *a2*, *a3* амплитудаларын және ϕ_2, ϕ_3 фазаларын бақылау керек.
7. Әртүрлі тригонометриялык функциялар үшін радиус-векторларды есептеу және графиктер ретін кұру. Реттеу үшін кез-келген батырманы басқанша ағымдағы терезені ұстап тұратын *pause* командасын колдануға болады. Бұрыш –π-ден π-ге дейінгі диапазонда өзгереді.

$$r1 = 2sin(5r)^2, r2 = sin(10r)^3$$

$$r3 = 2cos(10r)^2, r4 = 5cos(r)^3.$$

8. *[0.2, 3]* диапазонында $y(x) = \dfrac{sin^2 x}{1 + cos(x)} + e^{-x} ln(x)$ функциясының графигін тұрғызыңыз.
9. Параметрлік түрде берілген функцияның графигін тұрғызыңыз

$$\begin{cases} x = 4e^{-0.05t} sint \\ y = 0.2e^{-0.1t} sin2t \end{cases} \quad t \in [0, 50].$$

10. Әртүрлі тригонометриялык функциялардың графиктерін бір графиктік терезедегі кішігірім облыстарға шығарыңыз.

Төртінші сабақ. Екі айнымалыдан тәуелді функциялардың графиктерін тұрғызу

Сабақтың жоспары

1. Функциялардың үш өлшемді графиктері.
2. Графиктерді безендіру.
3. Параметрлік түрде берілген жазықтықтар мен сызықтарды тұрғызу.
4. Жарықтандырылған жазықтықты тұрғызу.

Matlab екі айнымалыдан тәуелді функцияларды көрсетудің әртүрлі әдістерін үш өлшемді графиктерді, деңгейлер сызықтарын және параметрлік түрде берілген беттер мен сызықтарды тұрғызуды ұсынады.

Функциялардың үш өлшемді графиктері

Екі айнымалыдан тәуелді функцияны көрсету үшін:

1. Функцияның тікбұрышты анықталу облысындағы тор түйіндерінің координаттарынан матрицаны кездейсок кұру қажет.
2. Тор түйіндеріндегі функцияны есептеп, алынған мәндерді матрицаға жазу керек.
3. *Matlab* графиктік функцияларының біреуін пайдалану керек.
4. Графикке косымша мәліметтерді енгізу, соның ішінде түстердің функция мәніне сәйкес келуін көрсету керек.

Тор екі аргументпен шақырылатын *meshgrid* командасының көмегімен кездейсок құрылады. Функцияның аргументтері ретінде тікбұрышты облыстағы ол салынатын торға сәйкес келетін вектор алынады. Егер функцияны тұрғызу облысы – квадрат болса бір аргументті пайдалануға болады. Функцияны есептеу үшін элементтер бойынша орындалатын амалдарды колдану керек.

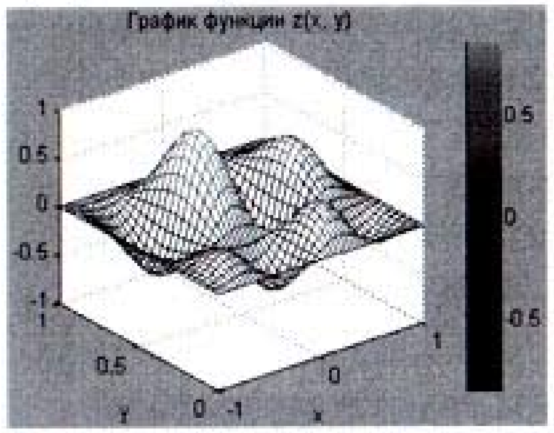
Matlab-тың екі айнымалыдан тәуелді функцияларды көрсету үшін берілетін негізгі мүмкіндіктерін $x \in [-1,1]$, $y \in [0,1]$ тікбұрышты анықтау облысында графигін тұрғызу мысалымен $z(x,y) = 4sin(2\pi x)cos(1.5\pi y) \times \times (1 - x^2) y(1 - y)$ қарастыруға болады. Алдымен тор түйіндерінің координаттарынан құрылған матрицаны және функция мәндерін дайындау керек:

```
>> [X, Y] = meshgrid(-1:0.05:1, 0:0.05:1);
>> Z =  4*sin(2*pi*X).*cos(1.5*pi*Y).*(1-X.^2).*Y.*(1-Y);
```

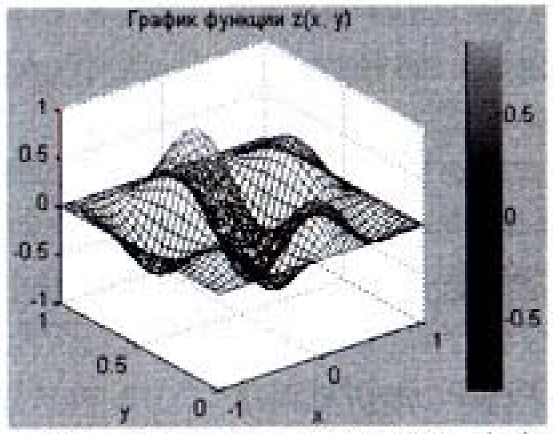
Каркасты бетті тұрғызу үшін, үш аргументпен шақырылатын *mesh* функциясы колданылады (4.1-суреті):

```
>> mesh(X,Y,Z)
```

Беттер сызықтарының түсі функция мәндеріне сәйкес келеді. *Matlab* беттердің тек көрінетін бөлігін салады. *Hidden off* командасының (4.2-сурет) көмегімен жасырын бөлігін қосып, каркасты бетті "мөлдір" етуге болады. *hidden on* командасы жасырын бөлікті жойып, графикті бұрынғы қалпына келтіреді.



4.1-сурет. Каркастың беті (*mesh*)

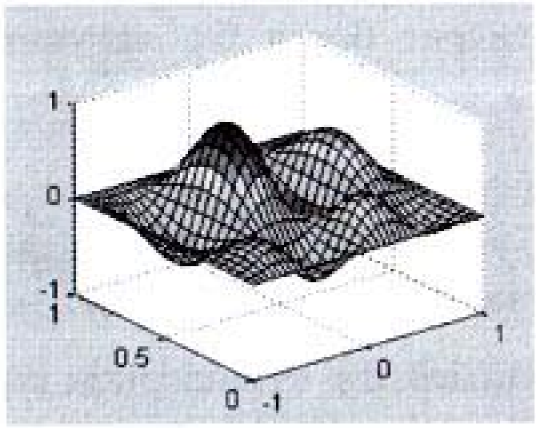


4.2-сурет. Мөлдір каркастың беті (*mesh, hidden of*)

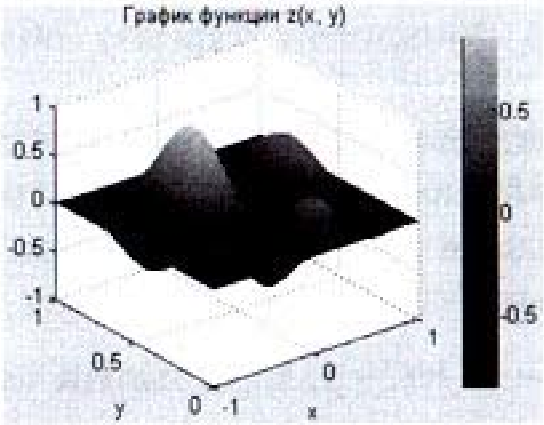
Surf функция графигіне каркастық бет құрастырып, беттің әрбір торын белгілі түспен бояйды, ол тордың бұрыштарына сәйкес нүктелердегі функцияның мәніне байланысты болады.
>> *surf(X,Y,Z)*
командасы 4.3-суреттегі көрсетілгендей нәтижеге әкеледі.

Әрбір тордағы түс тұрақты. *Shading flat* командасы каркастық сызықтарды алып тастауға мүмкіндік береді. *Shading interp* командасы функцияның мәніне байланысты түске боялған бетті алу үшін қолданылады (4.4-сурет).

Shading faceted 4.3-суреттегі көрсетілгендей бетке келуге мүмкіндік береді.



4.3-сурет. Түспен боялған каркастық бет (*surf*)



4.4-сурет. Түспен боялған бет (*surf, shading interp*)

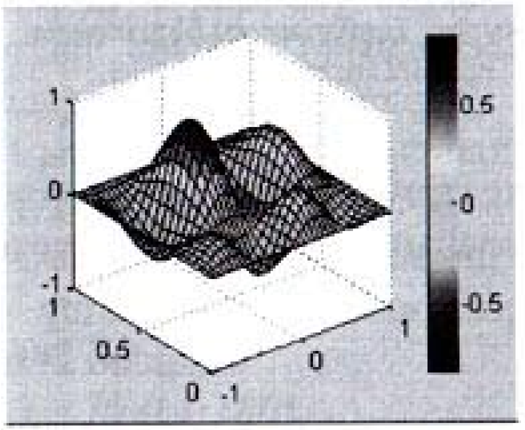
4.1-4.4-суреттерінде көрсетілген үш өлшемді графиктер беттің формасы туралы мәлімет алуға ыңғайлы болғанымен, олар арқылы функцияның мәнін талдау қиын.

Matlab-та *colorbar* командасы анықталған. Ол график жанында функцияның түсі мен мәнінің сәйкестігін көрсететін бағана шығарады. *Surf* арқылы беттің графигін құрыңыз және оны түсі туралы ақпаратпен толықтырыңыз.

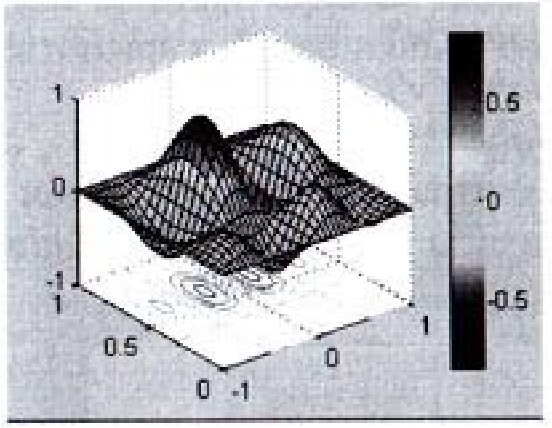
>> *surf(X,Y, Z)*
>> *colorbar*

4.5-суретте алынған нәтиже көрсетілген. *Colorbar* командасын үшөлшемді объектіні құрайтын барлық функциялармен сәйкес қолдануға болады.

Meshc және *surfc* командалары функция туралы нақтырақ ақпарат алуға мүмкіндік береді. Бұл командалар каркастық бетті немесе түспен боялған каркастық бетті (4.6-сурет) құрады және *x, y* жазықтығына функция деңгейінің сызығын орналастырады (функция мәндерінің тұрақтылық сызықтары):
>> *surfc(X,Y,Z)*
>> *colorbar*



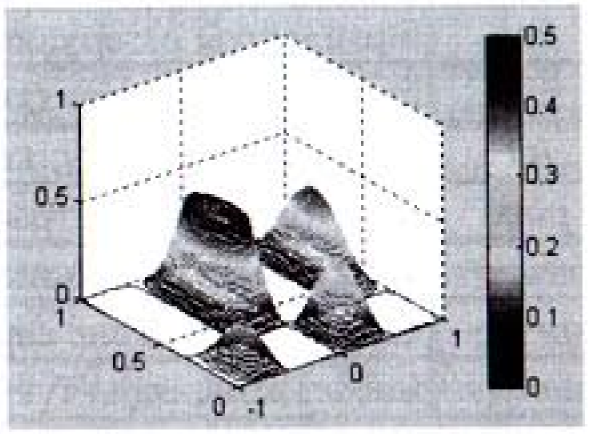
4.5-сурет. Функцияның түсі мен мәндерінің сәйкестігі (*colorbar*)



4.6-сурет. *x, y* жазықтығындағы деңгей сызықтары бар (*surf*) беттің графигі

Matlab contour3 функциясы арқылы деңгей сызықтарынан тұратын бетті құруға мүмкіндік береді. Бұл функцияны, сонымен бірге, жоғарыда келтірілген *mesh, surf, meshc* және *surfc* функциялары сияқты үш аргументпен шақыруға болады. Бұл жағдайда деңгей сызықтарының саны автоматты түрде таңдалады. *Contour3* функциясының төртінші аргументі ретінде деңгей сызықтарының санын немесе элементтері деңгей сызықтары түрінде берілген функцияның мәні болатын векторды беруге болады.

Функцияны оның кейбір мәндерінің аймағында зерттеу керек болса, векторды көрсеткен ыңғайлы. Функцияның 0-ден 0.5-ке дейінгі мәндері үшін 0.01 қадамымен деңгей сызығынан тұратын бетті тұрғызған кезде, келесі нәтижені көреміз (4.7-сурет):
>> *levels = [0:0.01:0.5];*
>> *contour3(X,Y,Z,levels)*
>> *colorbar*



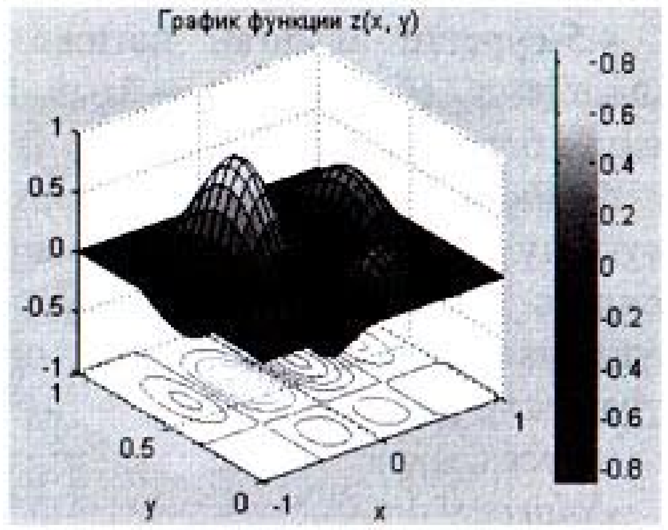
4.7-сурет. Деңгей сызығынан тұратын функция қиығының графигі (*contour3*)

Графикті безендіру

Графикті түспен безендірудің қарапайым, әрі тиімді әдісі *colormap* функциясы арқылы түс палитрасын орнату болып табылады.

Келесі мысалда *autumn* палитрасын қолдану нәтижесі 4.8-суретте көрсетілген.

```
>> surfc(X, Y, Z)
>> colorbar
>> colormap(autumn)
>> title('График функции z(x,y)')
>> xlabel('x')
>> ylabel('y')
>> zlabel('z')
```



4.8-сурет. Таңбалармен белгіленген график (палитра *autumn*).

Палитраның алғашқы мәнін қалпына келтіруде *colormap ('default')* командасы қолданылады. *Matlab*-та пайдаланылатын палитра түстері 4.1-кестесінде көрсетілген.

4.1-кесте

Түс палитрасы

Палитра	Түстің өзгеруі
autumn	бірқалыпты өзгеруі: қызыл-қызғылт сары-сары
bone	палитра *gray* палитрасын ұқсайды, бірақ көк түсті
colorcub	барлық түс коюдан ашыққа өзгереді
cool	көгілдір және қанық түсті
copper	сарғылт түсті
flag	циклдық өзгеру: қызыл-ақ-көк-қара
gray	сұр түсті
hot	бірқалыпты өзгеру: қара-қызыл-қызғыт сары-сары-ақ
hsv	бірқалыпты өзгеру (кемпірқосақтың түсіндей)
jet	бірқалыпты өзгеру: көк-көгілдір-жасыл-сары-қызыл
pink	палитра *gray* палитрасына ұқсас, қоңыр түсті белгілері бар
prism	циклдық өзгеру: қызыл-қызғылт сары-сары-жасыл-көк-күлгін
spring	қара қошқыл және сары түсті
summer	жасыл және сары түсті
vga	*Windows* палитрасы он алты түсті
white	бір ақ түс
winter	көк және жасыл түсті

Параметрлік түрде берілген жазықтықтар мен сызықтарды тұрғызу

Matlab келесі формулалармен берілген үшөлшемді сызықтарды құруға мүмкіндік береді:

$x = x(t),\ y = y(t),\ z = z(t),\ t \in [a,b]$

және беттер

$x = x(u, v),\ y = y(u, v),\ z = z(u, v),\ u \in [a, b],\ v \in [c, d]$

тәуелділігімен берілген.

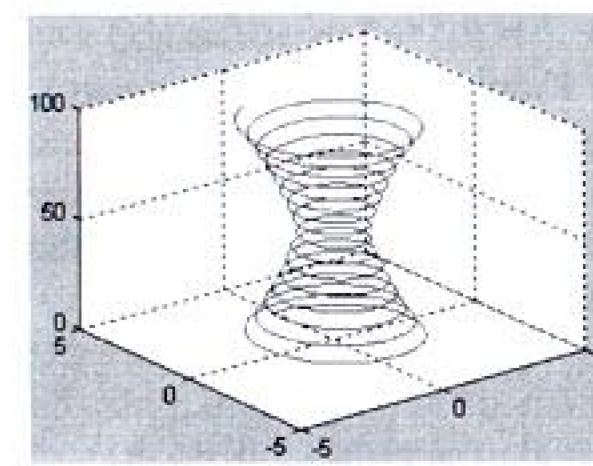
Plot3 функциясы параметрлік түрде берілген сызықтарды көрсетеді, оның аргументі ретінде $x(t)$, $y(t)$ және $z(t)$ функцияларының t нүктесіндегі мәндерінен құрылған векторлар алынған. Ең алдымен векторды құрып алу керек. Ол қос нүкте арқылы тұрақты қадаммен толтыру арқылы орындалады және векторларға функцияның сәйкес мәндері жазылып отырады.

Сызық келесі түрде сипатталсын: $x = e^{-|t-50|/50} \sin t,\ y = e^{-|t-50|/50} \cos t,$ $z = t,\ t \in [0,100]$.

Функция мәнін есептеп, графикті алу үшін келесі командаларды қолдануға болады:

```
>> t = [0:0.1:100];
>> x = exp(abs(t-50)/50).*sin(t);
>> y = exp(abs(t-50)/50).*cos(t);
>> z = t;
>> plot3 (x, y, z)
>> grid on
```

Нәтижесінде 4.9-суреттегі график шығады.



4.9-сурет. Параметрлік түрде берілген сызық *(plot3)*

Параметрлік түрде берілген бетті үш өлшемді графикті бейнелеуге арналған кез-келген функция арқылы құруға болады. Тек аргументтерді дұрыс дайындаудың маңызды зор. Өйткені $x(u,v)$ және $y(u,v)$ функциялары көп мәнді болуы мүмкін. Оны тор түйіндерінің құру аймағында орналасуы туралы ақпаратты сақтайтын матрица мен $z(u,v)$ функциясының осы нүктелердегі мәндерінен тұратын матрицаларды құруда есте ұстаған жөн.

Келесі тәуелділіктер бойынша берілген бетті (конус) тұрғызу есебі қойылсын:

$x(u,v) = 0.3 \cdot u \cdot \cos v,\ y(u,v) = 0.3 \cdot u \cdot \sin v,\ z(u,v) = 0.6 \cdot u,\ u,\ v\ ([-2\pi, 2\pi]$.

Ол үшін берілген интервалда қос нүкте арқылы параметрлер мәндерін сақтаушы вектор-баған және вектор-жол құру керек (u – вектор-баған, ал v – вектор-жол екені маңызды!):

>> *u = [-2*pi:0.1*pi:2*pi]';*
>> *v = [-2*pi:0.1*pi:2*pi];*

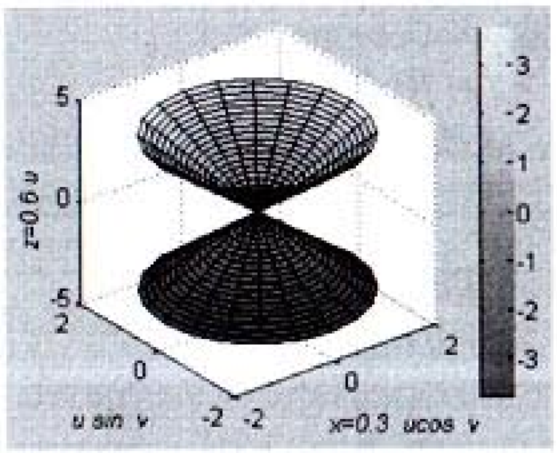
Ары қарай вектордың сыртқы көбейтіндісі арқылы табылған параметрлердің сәйкес мәндеріне тең нүктелердегі $x(u,v)$, $y(u,v)$ функцияларының мәндерінен X, Y матрицаларын құру керек (нүктесіз жұлдызша амалы):

>> *X = 0.3*u*cos(v);*
>> *Y = 0.3*u*sin(v);*

Z матрицасының өлшемі X, Y матрицаларының өлшемдерімен бірдей болу керек. Сонымен бірге ол параметрлердің сәйкес мәндерінен құрылуы керек. Егер де $z(u,v)$ функциясының құрамына u және v көбейтінділері кіргенде, онда Z матрицасын X пен Y матрицалары сияқты сыртқы көбейтіндіні қолдану арқылы толтыруға болар еді. Екінші жағынан, $z(u,v)$ функциясын $z(u,v)= 0.6 \cdot u \cdot g(v)$, мұндағы $g(v)=1$ түрінде көрсетуге болады. Сондықтан, Z-ті есептеу үшін тағы да u-дің бірлерден құралған v -дің өлшеміне тең өлшемді вектор-жолға сыртқы көбейтіндісін орындау қажет:

>> *Z = 0.6*u*ones(size(v));*

Қажетті матрицалардың барлығы құрылды. Үш өлшемді графиктерге жоғарыда келтірілген функциялардың қайсыбірін қолданып, берілген графикті құруға болады.



4.10-сурет. Параметрлік түрде берілген беттің графигі

4.10-суретте бейнеленген графиктер келесі командалардың тізбегімен орындалады:

>> *surf(X, Y, Z)*
>> *colorbar*
>> *xlabel('\itx=0.3 \itucos \itv')*
>> *ylabel('\ity=0.3 \itu sin \itv')*
>> *zlabel('\itz=0.6 \itu ')*

Мұнда *tex* командасының форматы қолданылған, ол графикке формулаларды қосуға мүмкіндік береді.

Жарық түсірілген бетті құру

Жарық түсірілген бетті құруда *surfl* функциясы қолданылады.

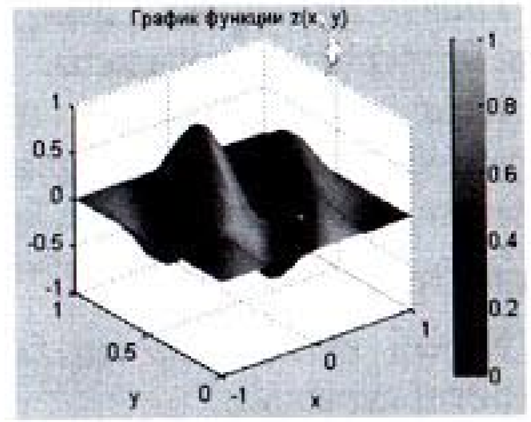
$x \in [-1,1]$, $y \in [0,1]$) тікбұрышты аймағында

$$z(x,y) = 4\sin(2\pi x)\cos(1.5\pi y)(1-x^2)y(1-y)$$

формуласы арқылы берілген жарық түсірілген бетті құру керек болсын. *surfl* функциясын қолданғанда, жарық интенсивтілігі сызықты өзгеретін *copper, bone, gray, pink* түсті палитраларымен шақыру тиімді.

Төменде келтірілген командалар арқылы 4.11-суретте көрсетілген жарық түсірілген бетті құруға болады:

>>*[X,Y] = meshgrid(-1:0.05:1, 0:0.05:1);*
>>*Z = 4*sin(2*pi*X).*cos(1.5*pi*Y).*(1-X.^2).*Y.*(1-Y);*
>>*surfl(X, Y, Z)*
>>*colormap('copper')*
>>*shading interp*
>>*xlabel('x')*
>>*ylabel('y')*
>>*zlabel('z')*



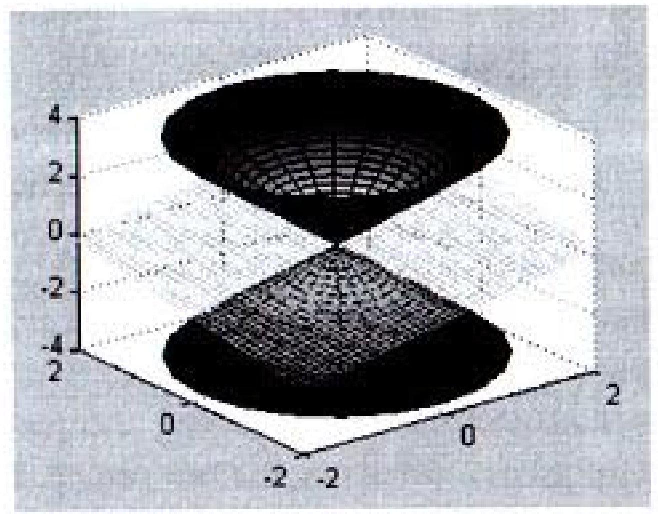
4.11-сурет. Жарық түсірілген беттің графигі *(surfl)*

Жарық көзі бақылаушыға қарағанда жоғарылау бұрышының мәні тең және 45°-қа артық азимутқа ие. *surfl*-тің қосымша төртінші аргументі екі элементтен тұратын вектор-жол – азимут және жарық көзінің жоғарылау бұрышының мәні. Мысалы, азимуттың көзін бақылаушыға қарағанда -90°, ал жоғарылау бұрышын нөлге өзгертіңіз. Мұны келесі командалар арқылы орындауға болады:

>> *[Az, El] = view;*
>> *surfl(X, Y, Z, [Az-90, 0])*
>> *shading interp*

Графиктерді біріктіру

Hold on командасы графиктерді біріктіруге арналған. Оны келесі графикті құрудың алдында шақыру керек. Төменде келтірілген мысалда параметрлік түрде берілген жазық пен конустың қиылысуы шығады. Нәтижесі 4.12-суретте көрсетілген.



4.12-сурет. Жазық пен конустың қиылысуы

>> *u = [-2*pi:0.1*pi:2*pi]';*
>> *v = [-2*pi:0.1*pi:2*pi];*
>> *X = 0.3*u*cos (v);*
>> *Y = 0.3*u*sin(v);*
>> *Z = 0.6*u*ones(size(v));*
>> *surf(X, Y, Z)*
>> *[X,Y] = meshgrid(-2:0.1:2);*
>> *Z = 0.5*X + 0.4*Y;*
>> *hold on*
>> *mesh(X, Y, Z)*
>> *hidden off*

Hidden off командасы жазықтың астында орналасқан конус бөлігін көрсету үшін қолданылады.

Hold on командасы графиктердің ағымдағы терезеге келесі қорытындылануының барлығына әсер етеді. Графиктерді жаңа терезеге орналастыру үшін *hold off* командасын қолдану керек.

Тапсырмалар

1. Келесі қатынастармен берілген эллипсоидтың мөлдір каркастық бетін құрыңыз:

$x(u,v) = cosu{\cdot}cosv,\ y(u,v) = 0.7cosu{\cdot}sinv,\ z(u,v) = 0.8{\cdot}sinu,\ u,\ v \in [-2\pi,\ 2\pi].$

2. Функциялардың графиктерін құрыңыз:

- $z = e^{-x^2-y^2}(x-1)^2 sin2\pi y,\ x,\ y \in [-1,\ 1];$
- $z = \dfrac{sinx}{x^2+y^2+0.3},\ x,\ y \in [-3,\ 3];$
- $z = x^2+y^2\ ,\ x,\ y \in [-3,\ 3]\ ;$
- $z = sin(x)cos(y)e^{\left|\frac{xy}{10}\right|},\ x,\ y \in [-3,\ 3]\ ;$
- $z = sin(x)(y+5)(4-y),\ x \in [-3\pi,\ 3\pi],\ y \in [-5,\ 4].$

3. Параметрлік түрде берілген қисықты құрыңыз:

$$\begin{cases} x = 4e^{-0.05t}sint \\ y = 0.2e^{-0.1t}sin2t \end{cases}.$$

4. $z = 0.7x + 0.5y$ жазықтығының берілген $x^2 + 2y^2 - 3z - 1 = 0,\ x,\ y \in [-3,\ 3]$ бетімен қиылысуын көрсетіңіз.
5. $x = -1,\ x = 0,\ x = 1$ жазықтықтарының берілген $2x^2 - y^2 = 3z,\ x,\ y \in [-3,\ 3]$ бетімен қиылысуын көрсетіңіз.
6. *Subplot* командасын қолданып және жоғарыда көрсетілген функцияларды пайдалана отырып үшөлшемді графиктерді құрудың әр түрлі әдістерін көрсетіңіз.

Бесінші сабақ. Анимацияланған графиктер

Сабақтың жоспары

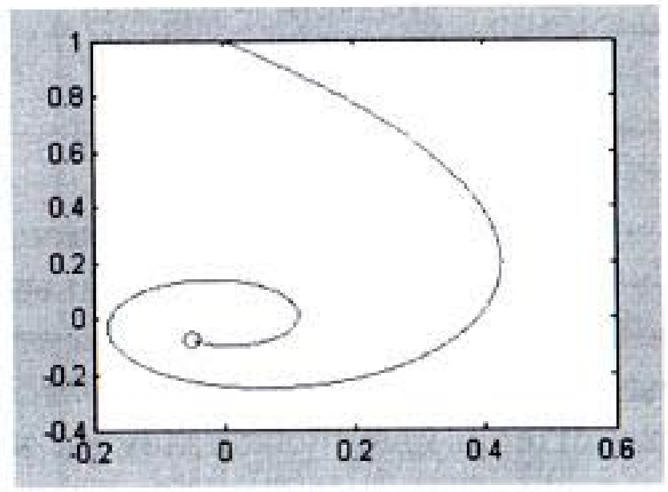
1. Анимацияланған графиктер.
2. Графиктік объектілердің қасиеттері.
3. *set* және *get* функциялары, ағымдағы объектілер.
4. Өстердің қасиеттері.
5. Сызықтар мен жазықтықтардың қасиеттері.
6. Объектілерге көрсеткіштер.

Matlab анимацияланған графикті алуға мүмкіндік береді, ондағы нүктені белгілейтін дөңгелек жазықтықта немесе кеңістікте қозғала отырып, артынан қозғалыс траекториясына сәйкес ізін қалдырады. Анимацияланған графиктер құру үшін *comet* және *comet3* функциялары қолданылады. Координаталары келесі заңмен өзгеретін қозғалыс нүктесінің 10 секунд ішіндегі траекториясын құру үшін

$x(t) = \dfrac{sint}{t+1}\quad y(t) = \dfrac{cost}{t+1},$

мына командалар қолданылады:

```
>> t = [0:0.001:10];
>> x = sin(t)./(t+1);
>> y = cos(t)./(t+1);
>> comet (x, y)
```



5.1-сурет. Траектория қозғалысының соңғы түрі (*comet*)

Соңғы команданы орындаған кезде, терезе графикпен бірге басқа терезелерден жоғары болуын қадағалау керек. Траекторияның соңғы түрі 5.1-суретте берілген.

Дөңгелек қозғалысының жылдамдығын уақытқа байланысты векторды автоматты түрде толтырғанда әр түрлі қадамдарды бере отырып басқаруға болады. *comet*-ті бір аргументпен (вектормен) шақыру нөмір элементтерінің графигін олардың нөміріне байланысты динамикалық түрде салуға мүмкіндік береді. *comet* функциясын кометаның құйрығының ұзындығына тең болатын үшінші қосымша сандық параметр арқылы да шақыруға болады. Алдын-ала оның мәні 0.1-ге тең.

Кеңістекте қозғалатын нүктенің траекториясын құру үшін *comet3* функциясы қолданылады. Нүктелердің координаталары 100 секунд ішінде келесі заңдармен өзгерсін:

$x = e^{-|t-50|/50} sint,\ y = e^{-|t-50|/50} cost,\ z = t.$

Нүктенің қозғалысының траекториясын көрсету үшін келесі командалар қолданылады:

>> t = [0:0.1:100];
*>> x = exp(abs(t-50)/50).*sin(t);*
*>> y = exp(abs(t-50)/50).*cos(t);*
>> z = t;
>> comet3 (x, y, z)

comet3 функциясын төртінші сандық аргумент арқылы да шақыруға болады. Онда *comet*-тегідей комета құйрығының ұзындығы беріледі.

Графикалық объектілердің қасиеттері

Matlab-тың ішкі функциялары қарастырылып отырған кез-келген объектінің қасиетінің мәнін өз қалауыңызша алуға және құруға мүмкіндік береді. Берілген функцияларды қолдану объектіні таңдаудан және қасиетінің мәнін құрудан тұрады.

set және get функциялары, ағымдағы объектілер

Графиктік объектілердің қасиеттерін зерттеуді косинусоидалар мысалынан көруге болады:
>> x = [0:0.1:10];
>> y = cos(x);
>> plot(x, y)

Matlab-тың графикалық элементтері объектілер болып табылады. Бұл жағдайда *plot* функциясының шақырылу нәтижесі үш гарфиктік объект: *Figure №.1* графикалық терезе, конустың графигі – осьтері мен сызықтарының құрылуы болып табылады. *Matlab* объектілерінің қасиеттерін манипуляциялау *get* және *set* функциялары арқылы жүзеге асырылады. *get* функциясы қасиеттің мәнін анықтауға, ал *set* функциясы жаңа мәндерді орнатуға арналған. Сонымен қатар *get* және *set* функцияларына берілген объектілердің қайсысымен жұмыс істеп отырғаныңызды көрсетуіңіз керек. *get* және *set*-тің кіру аргументі ретінде қолданылатын 3 стандартты функция бар:

gcf – ағымдағы графиктік терезе;
gca – ағымдағы өстер;
gco – ағымдағы графиктік объект.

gcf, gca, gco – ағымдағы терезе, өс және объектінің қасиеттеріне кіруге мүмкіндік береді. Қарастырылып отырған жағдайда ағымдық болып саналатын бір ғана графиктік терезе ашық тұр. Ағымды графиктік терезедегі жалғыз өсте ағымдық болып есептеледі.

Өстердің қасиеттері

Функция графиктерінен тұратын өстердің қасиеттерін алу үшін келесі команданы орындау керек:
>>get (gca)

Командалық терезеге кесте қасиеті мен оның мәні шығады. 5.1 және 5.2-кестелерде әдетте қосымшаларды құруда қолданылатын өстің қарапайым қасиеттері берілген. *get* функциясын екі аргументпен шақыруға болады. Екінші аргумент мәні қорытындылануға тиіс қасиеттің аты.

5.1-кесте

Өстің ортақ түріне жауап беретін қасиеттер

Қасиет аты	Сипаттамасы	Мәні
Box	Өстерді тік бұрышты аймаққа орналастыру	*on* немесе *off*
Color	Өс түсінің фоны	Үш элементті вектор *RGB* форматында түсті шақыру, мысалы *[1 1 1]*, немесе белгілі бір түсті: *r*, *g* және т.б. (1-қосымшаны қараңыз)
FontAngle	Остерді бөліп көру кезіндегі шрифтің иілуі	*normal* немесе *italic*
FontName	Шрифт аты	Шрифт атымен баған, мысалы *Courier*
FontSize	Шрифт өлшемі	Бүтін сан
FontWeigh	Шрифт қалыңдығы	*normal, bold, light,* немесе *demi*
GridLineStyle	Сетка сызығының стилі	-, —, :, -. немесе *none*
LineWidth	Сызық ені	Пункттердегі мән (1 пункт = 1/72 дюйм)
Visible	Өстердің бөліп көрінуі	*on* (Өстер алғашында көрінеді), *off*

5.2-кесте

Әр өстің қасиеті (*X* өсі мысалында)

Қасиет аты	Сипаттамасы	Мәні
XColor	Өс түсі	үш элементті вектор *RGB* форматында түсті шақыру, мысалы *[1 1 1]*, немесе белгілі бір түсті: *r*, *g* және т.б. (1-қосымшаны қараңыз)
XDir	Өс бағыты	*normal* немесе *reverse* (кері)
XGrid	Өске перпендикулярды тор	*on off*
XAxisLocation	Өстің орналасуы	*top* немесе *bottom* (*right* немесе *left Y* өсі үшін)

XLim	Айнымалылардың өзгеру шегі	Айнымалылардың өзгеру шегіне тең екі компонентті вектор, мысалы *[-1.5 2.3]*
XScale	Өс масштабы	*linear* немесе *log*
XTick	Өс бөліп көрсетілуінің координатасы	Бөліп көрсетудің координаталары векторы, *[0 1 3 5]*
XTickLabel	Өс көрсетілуі	Көрсетілу атымен берілген вектор ұяшығы (ұяшық саны көрсетілудің координаталарының векторының ұзындығына тең), мысалы *('zero'; 'one'; 'three'; 'five')*

Мысалы:

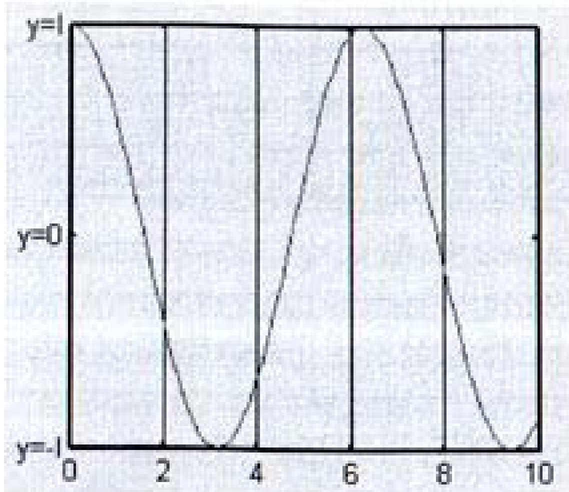
>> fn = get(gca, 'FontName')

командасы ағымдағы өсте, өзгермелі *fn* жолында қолданылатын шрифт атын жазады және экранға оның мәнін көрсетеді:

fn =

Helvetica

set командасы ағымдағы өстің әрбір қасиетіне мән беруге мүмкіндік жасайды (5.1-сурет). Бірінші аргумент *gca*, ал екінші аргумент ретінде өсті шектеуге қажетті командалар қолданылады, мысалы, ось сызығының қалыңдығын беру, координаталар және бөлуді көрсету, тор сызықтарын және түсті орнату командаларын қолдану (өс фонының түсі графиктік терезенің түсімен сәйкес):



5.1-сурет. Берілген қасиеттерімен косинус функциясының графигі

>> set (gca, 'Box', 'on')
>> set (gca, ' LineWidth', [2])
>> set (gca, 'YTick', [-1 0 1])
>> set (gca, 'YTickLabel', {'y = -1';'y = 0';'y = 1'})
>> set (gca, 'XGrid', 'on')
>> set (gca, 'GridLines', '-')
>> set (gca, 'Color', [0.8 0.8 0.8])

set командасы өзінің файл-программасынан немесе файл-функциясынан қасиеттерге ену мүмкіндігіне ие болып, графикті өз қалауымен өзгерте алады. Қасиет атының бөлігі *Mode* сөзімен аяқталады, мысалы *YTickMode*. Мұндай қасиеттер екі мәнге ғана ие бола алады – *'auto'* (алдын-ала құрылған) немесе *'manual'*, *'auto'* қасиетке тән мәнді

автоматты түрде таңдауға, бұл жағдайда *YTick*, мүмкіндік береді. *YTick*-та вектордың берілуі *YTickMode* мәнінің *'auto'*-дан *'manual'*-ға өзгеруіне әкеледі. Әркашан *YTick* қасиетінде орындалған өзгертулерді *YtickMode*-ті *'auto'*-ға қою арқылы жоюға болады.

Сызық пен беттің қасиеттері

Графиктегі ағымдағы сызыққа немесе бетке қатысуға арналған *Matlab*-та арнайы құрылған функция жоқ. Сызықты ағымдағы объектіге айналдыру үшін тышқанмен график терезесіне шерту, сосын кесте қасиетін және оның мәнін *gco* қолданып командалық терезеге апару керек:

>>get (gco)

5.3-кестесі неғұрлым жиі қолданылатын сызық қасиетінен тұрады.

Келесі командалар графикті 5.2-суретте берілген косинус түріне алып келеді.

>> set (gco,'Color', 'k')
>> set (gco,'LineWidth', 1)
>> set (gco, 'Marker', '')*
>> set (gco, 'MarkerFaceColor', 'w')
>> set (gco, 'MarkerSize', 10)

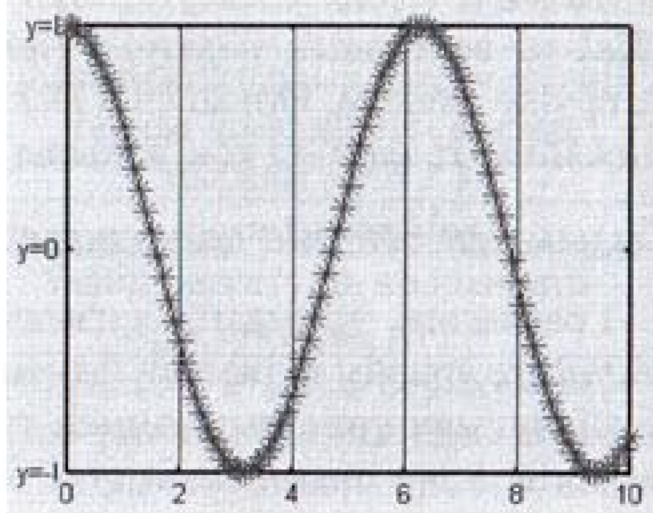
5.3-кесте

Сызық қасиеті

Қасиет аты	Сипаттамасы	Мәні
Color	Түс	Үш элементті вектор *RGB* форматында түсті шақыру, мысалы *[1 1 1]*, немесе белгілі бір түсті: *r*, *g* және т.б.
LineStyle	График сызығының стилі	-, — , :, -. немесе *none*
LineWidth	Сызық қалыңдығы	Пунктегі мәні
Marker	Маркер типі	Стандартты мәннің бірі, мысалы *o, s*
MarkerEdgeColor	Маркер шекарасының түсі	*Color* дағыдай
MarkerFaceColor	Маркер түсі	*Color* дағыдай
MarkerSize	Маркер өлшемі	Пунктер мәні

gco функциясы қолданушының тышқанды шерту арқылы таңдаған ағымдағы объектісін көрсетеді. Объект дәл осы уақытта жасалса, немесе қолданушы оны тышқанды шерту арқылы таңдаса, ағымдағы болып есептеледі.

Дегенмен, программаны орындау барысында қажетті объектінің кейбір қасиеттерін ендіру керек болады. Бұл көрсеткіштерді пайдалану арқылы тез шешіледі.



5.2-сурет. Сызық қасиетінің өзгеруі

Объектілерге көрсеткіштер

Matlab-та қандайда бір объектіні құрғанда, оған сандық көрсеткіштің пайда болуымен қатар жүреді. Осылайша, әрбір объект *Matlab* ортасында идентификацияланады. *gcf*, *gca* және *gco* функциялары ағымдағы терезе, өс және объектіге көрсеткішті орнатады. Графикті объектілерді құрғанда, олардың көрсеткіштерін айнымалыға жазып отырған дұрыс, нәтижесінде қажетті объектіге қатысу тез және ыңғайлы түрде орындалады. *Figure*, *axes*, *plot*, *mesh* және т.б. функцияларын шығу аргументтерімен шақыру оларға сәйкес графиктік терезеге, өске, графиктік сызыққа немесе бетке көрсеткіштің мәнінің берілуіне алып келеді. Сонымен бірге, егер *plot* бірнеше сызықты құрса (бірнеше пар вектор аргументі мен функция мәндері берілген), онда шығушы аргумент элементтері график сызығына көрсеткіштерден тұратын вектор болып табылады.

Мысалы, косинусты құрғанда графикке көрсеткішті келесі түрде шақыруға болады:

```
>>p = plot(x,y)
```

ары қарай *p* объектісін көрсете отырып, қасиеттерді шақыру:

```
>> set (p,'Color','r')
>> set(p,'Marker','*')
>> set(p,'MarkerFaceColor', 'b')
```

Беттің қасиеттері де дәл осындай түрде өзгереді.

Тапсырмалар

1. Шеңбердің бойында белгіленген нүктенің түзу бойынша сырғанауының (циклоида) қозғалыс траекториясын алыңыз. Циклоида келесі параметрлік тәуелділікпен берілген.

$x(t) = t - \sin t,\ y(t) = 1 - \cos t.$

Графикті басқару командаларын қолдана отырып объектінің қасиетін өзгертіңіз.

2. Функцияның анимацияланған графигін алыңыз.

$f(x) = sin(x)cos(y + p/2).$

3. Анимацияланған графикті параметрлік түрде берілген қисықпен алыңыз.

$x = 4e^{-0.05t} sint$

$y = 0.2e^{-0.1t} sin2t$

4. $z = exp(-x^2 - y^2)$ функциясының графигін $x \in [-1,\ 1],\ y \in [0,\ 1]$ тік бұрышты аймақта барлық белгілі әдістермен оларды жеке ішкі графиктерде орналастыра отырып құрыңыз.
5. Қандай да бір екі айнымалыдан тәуелді беттің графигін құрыңыз, тышқан тетігін шерту арқылы бетті ағымдағы жасап және қасиеттері мен оның мәндерінен құрылған кестені командалық терезеге шығарыңыз. Өз бетінше, мүмкін қасиеттерді зерттеңіз.

Алтыншы сабақ. *М*-файлдар

Сабақтың жоспары

1. *М*-файлдар тектері.
2. *М*-файлдардың құрылымы.
3. Айнымалылар тектері.
4. *М*-файлдарды құру.
5. *М*-сценарийлер.
6. *М*-функциялар.
7. *М*-функциялардың құрылымы.
8. Функциялар мен бұйрықтардың екіжақтылығы.

Matlab тілінің кодтарынан тұратын файлдарды *М*-файлдар деп атаймыз. *М*-файлды құру процедурасы екі амалдан тұрады:

- *М*-файлды мәтіндік редакторы арқылы құру:

Листинг 6.1. *myfile* программасы

function c = myfile(a, b)
c = sqrt((a.^2)+(b.^2))

- *М*-файлды командалық жолдан шақыру немесе басқа *М*-файлдан шақыру:

>> a = 7.5
>> b = 3.342
>> c = myfile(a, b)
c = 8.2109

Нәтижесінде, шығу айнымалысының мәнін аламыз.

М-файлдардың тектері

М-файлдарды 2 тегі бар: *М*-сценарийлер мен *М*-функциялар, олардың мінездемесі:

М-сценарийлер	*М*-функциялар
Кіріс және шығыс аргументтеріне ие	Кіріс және шығыс аргументтері мүмкін
Жұмыс облысындағы мәліметтермен жұмыс істейді	Ескертусіз ішкі айнымалылар функцияларға қатысты жергілікті болып табылады
Көп рет қайталанатын амалдарды автоматизациялау үшін колданылады.	*Matlab* тілінің мүмкіндіктерін кеңейту үшін қолдынылады (кітапхана функциясы, қолданбалы программалар пакеті)

М-файлдардың құрылымы

Функция ретінде құрылған *М*-файл келесі компоненттерден тұрады:

- Функцияларды анықтау жолдары;
- Түсініктемелердің бірінші жолдары;
- Түсініктемелер;
- Функция денесі.

function y = func(x)
x нүктесінде полиномның мәнін табу
*y = x.^3-x.^2+2*x*
%- компоненттерге амалдар қолдану

Осы функцияның құрылымы жалпы *Matlab* жүйесінің барлық функцияларына қатысты компоненттерден тұрады:

- Функцияны анықтау жолы – кіріс және шығыс аргументтерінің аты, саны мен жүру реттерін көрсетеді;
- Түсініктемелердің бірінші жолдары – функция қызметін анықтайды. Ол экранға *lookfor* немесе *help* <*каталог аты*> командасы арқылы шығарылады;
- Түсініктемелер – *help* < *функция аты*> командасы арқылы бірінші жолдармен бірге шығады;
- Функция денесі – бұл шығу аргументеріне мән беретін және есептеуді іске асыратын программалық код.

Айнымалылар тектері. Жергілікті және ауқымды айнымалылар

Айнымалыларды *М*-файлдарда колдану оларды командалық жолда қолданудан ешқандай айырмашылығы жоқ:

- айнымалылар хабарлауды талап етпейді, айнымалыларға мән бермес бұрын, ең алдымен оң жақтағы барлық айнымалыларға мән берілгені жөнінде білу керек;
- қандай болмасын меншіктеу амалы айнымалыларды құрады, егер қажет болса, алдыңғы айнымалылардың мәнін өзгертеді;
- кез-келген айнымалылар атауы әріптен, цифрлардан, төменгі сызықтардан басталады.

Жалпы *М*-файл ретінде берілетін әр *М*-функция өзінің жеке жергілікті айнымалыларынан тұрады, ол айнымалылар жұмыс облысының және басқа функциялардың айнымалыларынан ерекше. Егер бірнеше функциялар мен жұмыс облысы қандай да бір айнымалыны ауқымды ретінде хабарласа, онда олардың барлығы ол айнымалының бір ғана көшірмесін колданады. Бұл айнымалының кез-келген меншіктелуі ол денесінде ауқымды түрінде хабарланған барлық функцияларға тарайды.

Лотка-Вольтерра теңдеулерімен бейнеленген "жыртқыш-құрбан" моделі үшін *a* және *b* коэффициенттерінің әсерін табу мақсатында *lotka.m* атты *M*-файлын құру қажет.

Листинг 6.2. *lotka* программасы

```
function yp = lotka(t, y)
%LOTKA "жыртқыш-құрбан" моделі үшін Лотка-Вольтерра теңдеуі
global ALPHA BETA
yp = [y(D - ALPHA*y(1)*y(2); -y(2) + BETA*y(1)*y(2)];
```

Одан кейін командалық жол арқылы операторлар енгізіледі:

```
>> global ALPHA BETA
>> ALPHA = 0.01;
>> BETA =0.02;
>> [t, y] = ode23('lotka2', [0 10], [1; 1]);
>> plot(t, y)
```

global командасы *ALPHA* және *BETA* айнымалыларын ауқымды деп хабарлайды және *lotka.m* функциясында қол жетерліктей жасайды. Сол сияқты олар командалық жолдан өзгеруі мүмкін, ал жаңа шешімдер *lotka.m M*-файлын редактірлемей-ақ алынады.

Жұмыс облысының айнымалылары ауқымды болу үшін, оны командалық жолдан ауқымды екендігін көрсету керек; айнымалы шықпай тұрып әр функцияда *global* командасымен *M*-файлдар басында көрсетіледі.

M-файлдарды құру

M-файлдар кәдімгі мәтіндік файлдар болып табылады, олар мётіндік редактордың көмегімен орындалады. Дербес компьютердің операциялық ортасы үшін *Matlab* жүйесі арнайы енгізілген редакторды қолданады, кодтары бар басқа да мәтіндік редакторды қолдануға болады.

Редакторды екі түрлі тәсілмен ашуға болады:

- *File* менюінен *New* операциясын, одан кейін *M-File*-ды таңдаңыз;
- *edit <файлдың аты>*редактірлеу командасын қолдану арқылы.

Мысалы:

edit lotka командасы редакторды іске қосады және *lotka.m* файлын ашады. Егер файл аты болмаса, онда редактор іске қосылады және аты жоқ файл ашылады. Енді *func* функциясында мәтін жолдарын енгізіп және оларды ағымдағы каталог ішінде *func.m* атымен файлды сақтауға болады.

Осындай файл құрылғаннан кейін келесідей командаларды жүзеге асыруға болады:

- *what* командаларының көмегімен ағымдағы каталогтағы файл аттарын экран бетіне шығару;

- *type func* командасының көмегімен *func.m M*-файлдың мәтінін экран бетіне шығару;
- берілген параметрлері бойынша *func* функциясын шақыру:

```
>> func (5)
ans = 24
```

M-сценарийлер

M-файлдың ішіндегі сценарийлер ең қарапайым тектері болып табылады. Оларда кіріс және шығыс аргументтері болмайды. Олар негізінен командалық жолдан бірнеше рет енгізілетін *Matlab* командаларының есептеулерін автоматты түрдегі бірінен кейін бірі орындалуға тиіс амалдарды орындау үшін қажет.

Сценарийлер жұмыс облысынан мәліметтермен жұмыс істейді және де осы файлда әрмен қарай өнделетін мәліметтерді де құрастыра алады. Сценарийлерде қолданылатын мәліметтер жұмыс аймағында сақталады және оларды әрі қарай есептеулер жүргізу үшін қолдануға болады.

Келесі операторлар *theta* бұрышынан тәуелді әртүрлі тригонометриялық функциялар үшін радиус-векторды *rho* есептейді және полярлық координаталарда графиктерді құрастырады.

Листинг 6.3 Жапырақ графигін құрастыру

```
% M-file petals - Жапырақ графигін құрастырушы сценарий
theta = -pi:0.01:pi;
rho(1, :) = 2*sin(5*theta).^2;
rho(2, :) = cos(10*theta).^3;
rho(3, :) ; = sin(theta).^2;
rho(4, :) = 5*cos(3.5*theta).^3;
for i = 1:4
polar (theta, rho (i, :))
pause
end
```

Matlab жүйесінің командалық жолында *petals.m* командасын енгізу осы сценарийді орындатады.

Сценарий бірінші графикті көрсеткеннен кейін келесі графикке өту үшін *Enter*-ді басыңыз. Программаны орындап болғаннан кейін (*i, theta, rho*) айнымалылары жұмыс аймағында қалады. Олардың тізімін көру үшін *whos* командасын енгізіңіз.

M-функциялары

Функциялар кіріс және шығыс аргументтері бар *M*-файлдар болып саналады. Олар *Matlab* жүйесінің ортасынан бөлек өзіндік жұмыс аймағында ғана айнымалылармен жұмыс істейді.

average функциясы – вектордың элементтерінің орташа мәнін есептейтін *M*-файл.

Листинг 6.4. *average* программасы

```
function y = average (x)
% AVERAGE векторды ң элементтерінің орташа мәні
% AVERAGE(X), X – вектор, орташа мәнін есептейтін
% векторды ң элементтерінің
% егер енгізілетін аргумент вектор болмаса кездейсоқ беріледі
% қате
[m,n] = size (x);
if (~((m == 1) | ( n== 1)) | (m == 1 & n == 1))
error (' Input must be a vector') end
y =sum(x)/length(x);
% өзіндік есептеу
```

M-функциялардың құрылымы

M-функциялар келесідей компоненттерден тұрады:

- Функцияларды анықтау жолдары;
- Түсініктемелердің бірінші жолдары;
- Түсініктемелер;
- Функция денесі;
- жолдық түсініктемелер.

Функцияны анықтау жолы. Функцияны анықтау жолы *Matlab* жүйесіне файл *M*-функциялар болатындығын және кіріс аргументтерінің тізімін көрсетеді.

Мысалы:

average функциясының анықтау жолы келесідей түрде болады:

function y = average(x)

Мұндағы:

function – *M*-функцияны анықтайтын негізгі;

y – шығыс аргументі;

average – функция аты;

x – кіріс аргументі;

Егер функцияда бірден көп шығыс аргументтері болса, онда шығыс аргументтерінің тізімі тік жақшада жазылады. Егер кіріс аргументтері болатын болса, онда олар жай жақшаға жазылады. Кіріс және шығыс аргументтерін бір-бірінен ажырату үшін үтір қолданылады:

function [x, y, z] = sphere(theta, phi, rho)

Түсініктемелердің бірінші жолдары

Бұл қолданушы *help function_name* командасын тере бастаған кезде мәтіннің бірінші жолында пайда болады.

Түсініктемелер. M-файлдар үшін мәтіндерді бір немесе бірнеше түсініктемелер жолдарында енгізе отырып *online*-көмегін жасауға болады.

Каталогтың басының атауы

Contents.m атты арнайы файлды құру арқылы бүкіл католоrқа түсініктемелер құруға болады. Бұл файлда тек түсініктемелер жолдары болу керек.

Matlab help <*каталог_аты*> командасының көмегімен *Contents.m* файлының жолдарын шығарады. Егер *Contents.m* файлы ол каталогта жоқ болса, онда *help* <*каталог_аты*> командасы арқылы осы каталогтың әрбір *M*-файлы үшін түсініктемелердің бірінші жолдары шығады.

Функция денесі

Функция денесі *Matlab* тілінің кодын сақтайды, ал ол өз кезінде шығыс аргументтеріне мән береді, есептеулерді орындайды. Функция денесіндегі операторлар функция шақыруынан, команда ағымдарын басқару программаларының конструкцияларынан, интерактивті кіріс/шығыстан, есептеулерден, түсініктемелерден, меншіктеулерден, бос жолдардан тұрады.

Түсініктемелер % белгісімен белгіленеді. Түсініктемелер жолы *M*-файлдың кез-келген жерінде орналасуы мүмкін, сонымен қатар ол кез-келген жолдың аяғында да кездеседі.

Мысалы:

% x векторының барлық элементтерінің сомасын табу керек.

y = sum(x) % sum функциясы қолданылған.

M-функция аттары

M-функциясы бар файл аты функция атынан және "*.m*" түріндегі кеңейтілуден тұрады. Мысалы: *average.m.*

Егер функцияны анықтау жолындағы файл аты және функция аты әртүрлі болса, онда файл аты қолданылады да, ал ішкі ат ескерілмейді. Бірдей аттарды қолданған жөн.

Функция мен командалардың екіжақтылығы

Matlab жүйесінің командалары – бұл *load* түріндегі операторлар. Көп командалар операндылардың қосылуымен өзгеріске ұшырауы мүмкін.

```
>> load August17.dat
>> help magic
>> type rank
```

Өзгерістерді альтернативті беру тәсілі – оларды функциялардың жолдық аргументтері ретінде анықтау:

>> *load('August17.dat')*

>> *help('magic')*

>> *type('rank')*

Бұл – *Matlab* жүйесіндегі командалар мен функция түсініктерінің екі жақтылығы. *load* түріндегі кез-келген команда *command('argument')* формасындағы функция ретінде жазылуы мүмкін.

М-функцияларды орындау

Matlab жүйесінен немесе *М*-файлдардың ішінен *М*-функцияларды командалық жолға міндетті түрде қажет болатын атрибуттарды жазу арқылы анықтауға болады. Кіріс аргументтері жай жақшада, ал шығыс аргументтері тік жақшада жазылады.

Қателердің бар екендігі туралы хабарламаны шығару үшін *error('Message')* операторы қолданылады.

Тапсырмалар

1. Берілген өрнектердің $x \in [-1, 1]$ аралығындағы мәндерін есептейтін файл – функция құрыңыз:

$$e^{-x}\sqrt{\frac{x^2+1}{x^4+0.1}}\,.$$

2. Берілген вектор үшін орташа мәндерін және стандартты ауытқуларды көрсететін, матрица үшін сәйкес вектор мәндерін қайтаратын программаны жазыңыз.
3. Файл-функция құрыңыз, ол Евклид алгоритімінің көмегімен екі бүтін сандарды бөлгендегі ең үлкен бөлгішті және қателіктердің бар екендігін көрсететін хабарламаны табуы керек.
4. *subplot* командасын қолдану арқылы үшөлшемдік графиктерді құрастыратын әдістермен $x, y \in [-1, 1]$ аралығындағы *М*-сценарийді құрыңыз: $z = e^{-x^2-y^2}(x-1)^2 \sin 2\pi y$.
5. Үш өлшемді кеңістіктегі радиус-вектордың нүктелерінің ұзындығын есептеу программасын жазыңыз.
6. Элементтерді өсу бойынша әр жолда реттеуді көрсетіңіз.
7. Әр жолдағы барлық теріс элементтерді жол басына, ал оң элементтерді жол соңына қойыңыз.
8. Барлық минималды элементтерді матрицаның оң төменгі облысына (матрицаны жолдар бойынша толтыру керек), ал қалғандарын сол облыстың жоғарғы жағына орналастырыңыз.
9. Анықтаңыз: $y = max(min(a,b), min(c,d))$.

10. *(x1, y1)* және *(x2, y2)* координаталары бар нүктелер арасындағы қашықтықты анықтау
11. Коэффициенттері кез-келген ауқымды айнымалы арқылы берілген квадрат теңдеуді шешу программасын жазып, функция графигін тұрғызыңыз.

Жетінші сабақ. Дифференциалдық теңдеулерді шешу

Сабақтың жоспары

1. Қарапайым дифференциалдық теңдеулер жүйесін шешу үрдісі.
2. Теңдеулер жүйесінің оң бөліктерінің арнайы файл-функцияларын құру.
3. Солверді шақыру.
4. Қарапайым дифференциалдық теңдеулер жүйесінің солверлері.
5. Нәтижелерді көрсету.
6. Есептеулер дәлдіктерін көрсету.
7. *feval* функциясы.

Matlab жүйесінде қарапайым дифференциалдық теңдеулер жүйелерін шешудің процедуралар пакеті бар. Дәлірек айтсақ, олар қарапайым дифференциалдық теңдеулер жүйелерін шешуге қойылатын Коши есептеріне арналған.

Қарапайым дифференциалдық теңдеулердің үлкен класы, дәлірек айтсақ, t уақыты түріндегі бір тәуелсіз айнымалысы бар теңдеулер, оны үлкен туындыға қатысты шешкен кезде келесі алғашқы шарттарымен $y(t_0) = y_0$ бірінші ретті дифференциалдық теңдеулердің жүйесіне айналады:

$y(t) = F(t, y(t))$

Егер сәйкес жүйенің оң бөлігі тегіс болса, онда жүйенің бір ғана шешімі болады, ол негізінде *Matlab* жүйесінде қолданылатын қандай да бір алгоритмнің көмегімен сандық түрде табылуы мүмкін.

Бастапқы шарттары бар есептерді шешу үрдісі

Дифференциалдық теңдеулер үшін Коши есебі, кез-келген ретті дифференциалдық теңдеулерді қанағаттандыратын $t = t_0$ болғанда $y(t_0) = u_0, y'(t_0) = u_1, ..., y^{(n-1)}(t_0) = u_{n-1}$

алғашқы шартын қанағаттандыратын функцияны табудан құралады:

$$y^{(n)} = f(t, y, y', ..., y^{(n-1)}).$$

Мұндай түрдегі есептерді *Matlab*-та шешу өте оңай. Шешудің үрдісі келесідей этаптардан құралады:

1. Берілген дифференциалдық теңдеулерді бірінші реттегі дифференциалдық теңдеулердің жүйесіне келтіру.
2. Теңдеулер жүйесіне арнайы файл-функция жазу.
3. Сәйкес солверді шақыру.
4. Нәтижелерді көрсету.

Қозғалысты анықтайтын теңдеу келесі түрде берілсін: $y''+4y'+6y=cost$

Нүкте координатасы бастапқы жағдайда бірге тең, ал жылдамдығы

нөлге тең. Сонда сәйкес бастапқы шарттар келесідей түрде болады: $y(0)=1, y'(0)=0$.

Енді берілген есепті дифференциалдық теңдеулер жүйесіне келтіреміз. Ол үшін теңдеу ретіне сәйкес қосымша функциялар енгізіледі. Бұл жағдайда келесі формулалармен анықталатын y_1 және y_2 қосымша функцияларын енгіземіз

$y_1 = y$,

$y_2 = y'$.

Бастапқы дифференциалдық теңдеулер жүйесі бастапқы шарттармен келесідей түрге келтіріледі:

$y_1' = y_2$,

$y_2' = -4y_2 - 6y_1 + cost$.

Екінші этап негізінен дифференциалдық теңдеулер жүйесі үшін оның оң бөлігін сипаттайтын файл-функцияларды жазудан құралады. Файл-функцияның екі кіріс аргументі болуы керек. Біріншісі t ол бойынша дифференциалдау орындалатын t айнымалысы да, екіншісі өлшемі жүйедегі белгісіз функция сандарына сәйкес келетін вектор. Егер t жүйеге анық кірмейтін болса да, аргументтердің саны және реті тұрақты. Файл-функцияның шығыс аргументі болып жүйенің оң бөлігінің векторы алынады. Қарастырылып отырған жүйенің файл-функциясының оң бөлігі келесідей түрде табылады.

Листинг 7.1. Теңдеулердің оң бөлігіндегі файл-функция.

```
function F = dif(t, y)
F= [y(2); -4*y(2) - 6*y(1) +cos(t)];
```

Есепті шешуде *ode45* солвері қолданылады. Солвердің кіріс аргументтері қарапайым жағдайда келесілер болып табылады:

- Апострофқа алынған файл-функция аты;
- Тербелістерді бақылау уақытының бастапқы және соңғы мәндерінің векторы;
- бастапқы шарттардың векторы.

Шығыс аргументтері екеу:

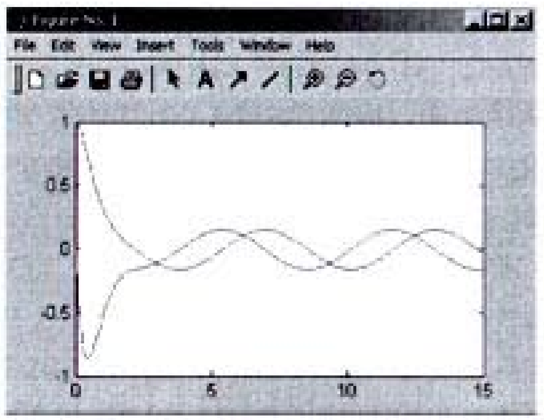
- уақыттың мәнін сақтайтын вектор;
- уақытқа сәйкес белгісіз функция мәндерінің матрицалары.

Функция мәндері матрица бағаналарында орналасады, бірінші бағанада – бірінші функция мәндері, екіншісінде – екінші т.с. $y_1 = y$, $y_2 = y'$ ауыстыру жасалуының күшіне сай матрицаның бірінші бағанасында бастапқы дифференциалдық теңдеулер жүйесіне кіретін белгісіз $y(t)$ функциясының мәнінен, ал қалған бағаналар оның туындыларынан құрылады. Алынған нәтижелерді графиктерде көрсетуге болады. $t < 15$ болғанда шешімін табу үшін солверді қолдану және нәтижелерді *solvdif* файл-программасының мысалында көрсету келтірілген.

Листинг 7.2. Дифференциалдық теңдеулері шешу

```
% Дифференциалдық теңдеулері шешу үшін solvdif файл-программасы
y0 = [1; 0];
[T,Y] = ode45('dif', [0 15], y0);
plot(T, Y(:,1))
hold on
plot(T, Y(:,2))
tit1e(дифференциалдық теңдеідің шешуі')
ylabel('y, y'' ')
legend (' координата' , 'жылдамдық', 4)
grid on
hold off
```

Файл прграмманы орындаған кезде экранға 7.1-суретінде көрсетілген графиктер шығады және олар уақытқа сай нүкте координаталары және оның жылдамдығының өзгерістерін көрсетеді. Графикте көрсетілгендей, жуық шешім мен оның туындысы бастапқы шарттарды қанағаттандырады және t = 5 нүктесінен бастап тербеліс орнықты тәртіпте жүреді.



7.1 сурет. Дифференциалдық теңдеуді шешу графигі

Бастапқы шарттарды қанағаттандыратын берілген дифференциалдық теңдеулер жүйесін шешу төртінші реттегі Рунге-Кутта әдісін пайдаланатын *ode45* солверінің көмегімен алынды. *Matlab*-та *ode45* солверінен басқа, тағы да басқа солверлер бар. Дифференциалдық теңдеулер жүйесін шешетін солверді таңдаған кезде, оның қасиеттерін ескеру қажет, әйтпесе дәл нәтижені алу мүмкін болмайды немесе тым көп уақыт жұмсауға тура келеді.

Қарапайым дифференциалдық теңдеулерді шешу әдістері

Әртүрлі есептерді шығарған кезде қарапайым дифференциалдық теңдеулерді сандық шешудің түрлі процедураларын таңдауға болады. Мысалы, жоғарыда келтірілген мысалда *ode45* функциясы қолданылған. Қатаң емес теңдеулер жүйелерін шешу үшін *Matlab*-та келесі функциялар бар:

- *ode45* – Рунге-Кутта нақты әдісіне негізделген: Бұл бірқадамдық алгоритм – $y(t_n)$-шығару үшін алдыңғы нүктенің біріндегі $y(t_{n-1})$ мәнді білу керек. Бұл функция көп жағдайда бастапқы шешімдер үшін тиімді.
- *ode23* – ол да нақты Рунге-Кутта әдісіне негізделген, бірақ реті төмен, сондықтан дәлдігі төмен және аздаған қатаңдыққа сай шешімдер үшін тиімді келеді. Ол да бірқадамдық әдіс.
- *ode113* – Адамс-Бэшфорт-Милтонның айнымалы ретті әдісін қолданады. Ол *ode45* әдісіне қарағанда, әсіресе ерекше жоғарғы дәлдік қажет болған жағдайда және теңдеудің оң жағын есептеу күрделі кезде анағұрлым тиімді болуы мүмкін. Көпқадамдық әдіс, сондықтан шешуді бастамас бұрын алғашқы бірнеше нүктелердегі шешімдерді білу керек.

Matlab жүйесінде қатаң теңдеулер жүйесін шешу үшін төрт функция қарастырылған:

- *ode15s* – кері сандық дифференциалау әдісіне негізделген, ол Гир әдісі ретінде әйгілі. *Ode113* әдісі сияқты бұл әдіс те көпқадамдық болып келеді.
- *ode23s* – Розенброктың екінші ретті әдісін қолданады. Бұл бірқадамдық әдіс болғандықтан, *ode15s* әдісіне қарағанда жоғары емес дәлдік жағдайлары үшін тиімдірек болады.
- *ode23t* – бос көбейткіші бар трапециялар ережелерін іске асыру болып табылады. Бұл әдісті егер есеп онша күрделі емес болса, және есепті сандық демпфирлеу керек емес болса қолданудың мәні бар.
- *ode23tb* – Рунге-Куттаның айқын емес формуласы бойынша екі деңгейлі шешімді іске асырады. *ode23s* әдісі тәрізді бұл әдіс шешімнің жоғары емес дәлдігін қажет етпейтін кезде тиімді.

Дәлдік немесе есептеулер қателігі алынған шешімнің сапасына әсер етеді. *ode45*, *ode23*, *ode15s*, *ode23s*, *ode23t*, *ode23tb* солверінің жұмысын басқару, дәлдікті белгілеу үшін options қосымша параметрі қолданылады. Оны *odeset* функциясымен құру керек.

options = odeset (..., қадағалау түрі, мәні, ..)

Есептеулердің салыстырмалы түрдегі қателіктерін төмендету үшін *odeset*-ті қолдану арқылы *options* параметрін қалыптастыру керек және *options* солвердің төртінші қосымша аргументі ретінде болу керек. Салыстырмалы қателікті көрсету үшін *'RelTol'* аргументі, ал абсолютті қателіктер үшін *'AbsTol'* аргументі қолданылады. Мысалы:

>> options = odeset (' RelTol', 1.0e-04, 'AbsTol',1.0e-03)

Солверді шақыру мына түрде жазылады:

>> y0 = [1; 0];

>> [T,Y] = ode45('dif', [0 15], y0,options);

Мүмкін параметрлердің толық тізімі *Matlab*-тың анықтамалық жүйесінде берілген.

feval процедурасы

Matlab-та кез-келген функция (процедура), мысалы *FUN1*, тек қарапайым хабарласу көмегімен ғана емес

[y1, y2, . . . , yk] = FUN1 (x1,x2, . . . ,xn)
арнайы процедура *feval* арқылы жүзеге асырылуы мүмкін:
[y1,y2, . . .,yk] = feval (' FUN1' ,x1,x2, .. . ,xn)
мұндағы *FUN1* функциясының атауы кіріс мәтіндік айнымалыларының бірі, сол себептен апострофка орналастырылады.

Функцияның екінші формада шақырылу ерекшелігі, функцияның атауы өзгерген кезде шақыру өз формасын жоғалтпайды, мысалы, *FUN2*. Кіріс және шығыс параметрлерінің саны бірдей нақты типті барлық функцияларға хабарласуларды бірыңғайлауға болды. Мұнда функция атауы кез-келген және қайта хабарласу кезінде өзгеруі мүмкін.

Себебі функцияны *feval* процедурасы көмегімен шақырған кезде функция атауы процедураның кіріс параметрі ретінде қарастырылады. Бұл функция атауын айнымалы ретінде қолдануға және оны функцияның нақты атауын білмей тұрып *M*-файлға хабарласу ретінде рәсімдеуге болады.

Анықталған интегралды жуықтап есептеу

Физика, химия, экология, механика және басқа да жаратылыстану ғылымдардағы көптеген есептерді шешу анықталған интегралды есептеуге тіреледі.

Күнделікті өмірде Ньютон-Лейбниц формуласын қолданудың реті бола бермейді. Ондай жағдайда сандық интегралдау әдістері қолданылады. Ол әдістер келесілерге негізделген: $\int_a^b f(x)dx$ анықталған интегралы геометриялық тұрғыдан қарастырғанда, қисық сызықты трапецияның ауданын көрсетеді. Сызықты интегралдау идеясы $[a; b]$ интервалын кішігірім интервалдарға бөле отырып ізделінді ауданды элементарлы аудандардың қосындысы ретінде қарастыруға негізделген. Пайдаланылған аппроксимацияға қатысты сандық интегралдаудың әртүрлі дәлдікке қол жеткізуге мүмкіндік беретін формулалары табылады. Трапеция және Симпсон (парабола) әдістерін қарастырайық.

Трапеция әдісі

Бұл әдісте сызықтық аппроксимациялау қолданылады, $y = f(x)$ функциясының графигі y_i нүктелерін қосатын қисық түрінде көрсетіледі. $h = \frac{b-a}{n}$ қадамы тұрақты болғанда, мұндағы n – учаскелер саны трапеция формуласы төмендегідей болады:

$$\int_a^b f(x)dx = h\left(\frac{y_0 + y_n}{2} + \sum_{i=1}^{n-1} y_i\right).$$

Matlab ортасында бұл формуланы *trapz (x,y)* программасы орындайды.

Симпсон әдісі

Интеграл астындағы функцияны параболамен ауыстыратын болсақ, тұрақты интегралдау қадамымен Симпсон формуласы келесі түрге келеді

$$\int_a^b f(x)dx = \frac{h}{3}\left[y_0 + 4(y_1 + y_3 + ... + y_{n-1}) + 2(y_2 + y_4 + ... + y_{n-2}) + y_n\right].$$

Matlab ортасында Симпсон формуласы *quad* программасы арқылы орындалады. Интеграл астындағы функция @ дескрипторы көмегімен берілген жағдайда, ол файл – функцияда программаланады, немесе апострофтар көмегімен *quad* программасының өзінде жазылады. Интегралдау дәлдігі келісім бойынша $1 \cdot 10^{-6}$ ге тең.

Мысал. Анықталған интегралдың мәнін трапеция және Симпсон әдістерімен есептеп, баспаға шығару керек:

$$\int_0^1 \frac{dx}{1+x^2}$$

Анықталған интегралдың мәнін трапеция әдісімен есептеуді келесі түрде ұйымдастыруға болады:

```
>> x = 0:0.0001:1.0;
>> y = 1./(1 + x.^2);
>> z = trapz(x, y)
z =
   0.7854
```

Анықталған интегралдың мәнін Симпсон әдісімен есептеу:

```
>> quad ('(1./(1 + x.^2))', 0, 1)
ans =
    0.7854
```

Интегралдың дәл мәні *0.785398163* тең.

Мысалдан көріп отырғанымыздай, алынған нәтижелер дәл және есептеулердің өзі қарапайым.

1 Тапсырмалар

1. Дифференциалдық теңдеудің шешімін табу және ізделінді функциясының және туындысының графигін құрыңыз:
$y'' + 2y' + 10y = sint, t \in [0\ 12], y(0) = 1, y'(0) = 0.$
2. Лотка-Вольтер теңдеуінің шешімін табыңыз. $y_1(t)$ – құрбандар саны, ал $y_2(t)$ – жыртқыштар саны. Уақыт өтуіне сай мына заңдар

бойынша өзгеріп отырады: $\begin{matrix} y_1' = Py_1 - py_1y_1 \\ y_2' = -Ry_2 + ry_1y_2 \end{matrix}$

мұндағы P – жыртқыштар жоқ кездегі құрбандар санының ұлғаюы, R – құрбандар жоқ кездегі жыртқыштар санының кемуі. Жыртқыштардың құрбандарды жеуінің ықтималдығы олардың санына пропорцианалды және py_1y_2 – құрбандардың жоғалып бітуіне, ал ry_1y_2 – жыртқыштардың пайда болуына сай пропорционалды түрде өзгеріп отырады. Коэффициенттерді ауқымды етіп хабарлап, олардың $P = 3$, $R = 2$, $p = r = 1$ мәндері үшін және бастапқы кезеңде құрбан саны үш, ал жыртқыштар саны – төрт деп алыңыз.

3. Дифференциалдық теңдеудің шешімін табу және ізделінді функциясының және туындысының графигін құрыңыз:

$$\begin{matrix} y_1' = y_2 \\ y_2' = -\dfrac{1}{t^2} \end{matrix}, \ t \in [a\ 100],\ y(a) = 1,\ y'(a) = 1/a,\ a = 0.001.$$

4. Ван-дер-Поль теңдеуінің шешімін табыңыз:

$$\begin{matrix} y_1' = y_2 \\ y_2' = \mu(1 - y_1^2)y_2 - y_1 \end{matrix}, \ t \in [0\ 30],\ y_1(0) = 0,\ y_2(0) = 1,$$

μ коэффициентінің әртүрлі мәндері үшін теңдеу шешімінің графигін тұрғызыңыз.

5. Теңдеудің шешімін табыңыз:
$y'' = 2yy'$ $t \in [0\ 10]$, $y_1(0) = 1$, $y_2(0) = 1$.

6. Теңдеудің шешімін табыңыз: $tyy'' - ty'^2 = yy'$ $t \in [0\ 20]$, $y_1(0) = 1$, $y_2(0) = 0$.

7. Теңдеудің шешімін табыңыз: $ty'' = y' + t(y'^2 + t^2)$ $t \in [0\ 50]$, $y_1(0) = 0$, $y_2(0) = 0$.

8. Коши түріндегі қарапайым дифференциалдық теңдеулер жүйесі беріледі:

$$\varphi'' + \sin\varphi = S(t, \varphi, \varphi')$$

$$S = -2\varsigma \cdot \varphi' - n_1 \sin(\vartheta \cdot t + \varepsilon_1)\cos\varphi - n_2 \sin(\vartheta \cdot t + \varepsilon_2)\sin\varphi.$$

мұндағы φ-жүйенің қалып-күйінің айнымалыларының векторы, t – аргумент. Теңдеуді шешкенде, коэффициенттерді бір векторға $k = [\varsigma,\ \upsilon,\ n_1,\ n_2,\ \varepsilon_1,\ \varepsilon_2]$ біріктіруге болады және осы векторды ауқымды деп көрсету керек. S функция жеке есептеу керек және *feval* функциясының көмегімен теңдеудің оң бөлігіне қою керек. $t \in [0\ 50]$, $\varphi(0) = 8/9\pi$, $\varphi'(0) = 0$.

2 Тапсырмалар

Анықталған интегралдың мәнін трапеция және Симпсон әдістерімен есептеп, баспаға шығару керек. Берілгендер 7.1-кестесінен алынады.

7.1-кестесі

№	Интеграл астындағы функция $f(x)$	Интегралдау интервалы $[a;\ b]$	Интегралды есептеу дәлдігі
1	$\ln x / x\sqrt{1+\ln x}$	[1; 3.5]	0.001
2	$\mathrm{tg}^2 x + \mathrm{ctg}^2 x$	[π/6; π/3]	0.002
3	$1/x\ln x$	[1.5; 3]	0.0001
4	$\ln^2 x / x$	[1.0; 4.0]	0.003
5	$\sqrt{e^x - 1}$	[0; ln2]	0.0015
6	$xe^x \sin x$	[1.0; 4.0]	0.002
7	$x\dfrac{e^x - e^{-x}}{2}$	[0.0; 2.0]	0.001
8	$1/\sqrt{9+x^3}$	[2.0; 5.0]	0.001
9	$\sin(1/x)x^4$	[1.0; 2.5]	0.0005
10	$x^3 \mathrm{arctg} x$	[0.0; $\sqrt{3}$]	0.003
11	$\arcsin\sqrt{x/(1+x)}$	[0.0; 3,0]	0.001
12	$x^2(1+\ln x)$	[1.5; 3.0]	0.0025
13	$1/\sqrt{1+3x+2x^2}$	[0,0; 5.0]	0.001
14	$\sqrt{x^2 - 0.14/x}$	[2.3; 6.0]	0.002
15	$2^{3x}\ln\lvert\cos x\rvert$	[0.0; π/2]	0.001

Сегізінші сабақ. *Matlab* ортасында программалау негіздері

Сабақтың жоспары

1. Қайталау операторлары.
2. Тармақталу операторлары.
3. Қайталауды үзу. Ерекше жағдайлар.
4. Массивтер мен сандардан құрылған логикалық өрнектер.
5. *find* функциясы.

Алдында карастырылған файл-функция және файл-программалар программалардың мысалдарының ішіндегі ең қарапайымы. *Matlab*-тың барлық командалары бірінен кейін бірі тізбектей орындалады. Көптеген цикл бойымен қайталанатын күрделі есептерді орындау үшін арнайы программалар қажет.

Бұл сабақта *Matlab*-тың негізгі операторлары туралы сөз қозғаймыз. Операторларды файл-сценарийлерде және файл-функцияларға қолдануға болады, ал ол өз кезегінде бұтақталған құрылымы бар күрделі функцияларды құруға мүмкіндік береді.

Ағымдарды басқару

Matlab-та ағымды басқарудың бес құрылымдық түрі бар:

- *if* операторы;
- *switch* операторы;
- *for* операторы;
- *while* операторы;
- *break* операторы.

for, while қайталау операторлары

for-ды ең қарапайым түрде қолдану келесідей:

```
for count = start:step:final
Matlab командалары
end
```

мұндағы *count*-циклдық айнымалы, *start*-оның бастапқы мәні, *final*-соңғы мәні, ал *step*-циклға келесі кірген кездегі *count* мәніне қосылып отыратын цикл қадамы. Цикл *count* мәні *final*-дан үлкен болған кезде бітеді. Циклдың айнымалысы тек бүтін ғана емес, сонымен қатар кез келген таңбалы нақты мәнді қабылдайды. *for* циклының операторларын қолдану мысалы:

Листинг 8.1. *for* операторын қолдану

```
for i = 1:m
for j = 1:n
H(i, j) = 1/(i+j-1);
```



```
end
end
H
```

Командалардың бұл терімі арқылы $m \times n$ өлшемді Гильберт матрицасы құрылады және экранға шығады. Ішкі операторды аяқтайтын нүктелі үтір экранға қажетсіз аралық мәндердің шықпауын қадағалайды. *H* операторы соңғы нәтижені экранға шығарады.

while циклі

Жалпы түрде *while* операторы келесідей қолданылады:

```
while <шарт>
<операторлар>
end
```

<операторлар> *<шарт>* шындық болғанша орындала береді. Мысалы, берілген *a* саны үшін операторлар тізбегі берілген шартты қанағаттандыратын ең кіші теріс емес *n* санын есептейді және экранға шығарады:

Листинг 8.2. *while* операторын қолдану

```
n = 0;
while 2^ n < a
n = n + 1;
end
n
```

Шартты if операторы

Жалпы түрде қарапайым *if* операторы келесідей қолданылады:

```
if <шарт>
<операторлар>
end
```

<операторлар> *<шарт>* тек шындық болғанда ғана орындалады. Келесі мысалда көрсетілгендей көптеген тармақталу мүмкіндігі бар

```
if n < 0
parity = 0;
elseif mod(n,2) == 0
parity = 2;
else
parity = 1;
end
parity
```

Таңдау операторы switch ... case

Екіден артық логикалық шарттармен тармақталу конструкцияларын құру қажет болса *if* операторы орнына *switch ... case* таңдау

операторын қолданған ыңғайлы. Ол оператор құрылымы келесідей:
switch <өрнек >
case < мән 1>
операторлар
case < мән 2>
операторлар...
otherwise
операторлар
end

Шарттар (қатынас операторлары)

Matlab-та келесі қатынас операторлары қолданылады:
< кіші
> үлкен
<= кіші немесе тең
>= үлкен немесе тең
= = тең
~= тең емес
= белгісі меншіктеу операторларында, ал = = белгісі қатынас амалдарында қолданылады. Қатынас операторлары келесі логикалық операторлар арқылы біріктірілуі мүмкін
& ЖӘНЕ
| НЕМЕСЕ
~ ЕМЕС

Егер бұл операторлар скалярлармен орындалса, нәтиженің шындықтығына немесе жалғандығына байланысты *1* немесе *0* скалярлары шығады. *3 < 5, 3 > 5, 3 ==5, және 3 == 3* есептеуге тырысыңыздар. Қатынас операторлары бірдей өлшемді матрицаларға қолданылған жағдайда, нәтижесінде берілген матрицаның сәйкес элементтерінің қатынасына байланысты элементтер ретінде *0* немесе *1* тұрған сондай өлшемді матрицаны аламыз. *a = rand(5), b = randn(a), a == b* есептеуге тырысыңыздар. *while* және *if* операторлары егер нәтижедегі матрицада *0*-ге тең мәндер болмаса, матрицалардың арақатынастарын шындық түрінде түсінеді.

find функциясы

k = find(x) операторы *x* вектор матрицасының нөлге тең емес элементтерінің нөмірлерін *k* векторына қайтарады. Егер *x* матрица болса, индекстерді анықтағанда, ол матрицаның бағаналарынан тізбектей біріктірілгеннен құрылған вектор ретінде қарастырылады. *find(x)* векторын қатынас операторымен қолдануға болады, өйткені қатынас операторын матрицаға қолданғандағы нәтижесі *0* және *1*-лерден тұратын матрица. Сонымен, бір *find* операторы арқылы кейбір шарттарды қанағаттандыратын барлық мартица индекстерін бірден анықтауға

және жазуға болады. Егер *for* циклды операторын *for k = KK* түрінде жазуға болатынын еске түсірсек, ондағы *KK*-бүтін вектор, онда оларды бірге қолдану ыңғайлы. Мысалы, егер *<операторды>* матрицаның тек *3*-тен үлкен элементтеріне ғана орындау керек болса, онда келесі түрдегідей қолдану ыңғайлы:
for i = find(A > 3)
<оператор>
end;

Тапсырмалар

1. $n = 20;\ m = 10;\ x_i,\ i \in [0,\ 6]$, берілген, экранға $y(x_i)$ мәнін әрқайсысы қосылғыштың бір мәнін есептейтін екі функцияны қолдана отырып табыңыз

$$y(x_i) = f(x_i) = \underbrace{\left(\frac{x^2}{1!} + \frac{x^3}{2!} + \frac{x^4}{3!} + \ldots + \frac{x^{n+1}}{n!}\right)}_{1\,қосылғыш} + \underbrace{\left(e^x - sin(x)\right) + \left(e^{2x} - sin(x)\right) + \ldots + \left(e^{mx} - sin(x)\right)}_{2\,қосылғыш}.$$

2. $x_i,\ i \in [0,\ 4],\ n = 8;\ m = 15$. берілген, экранға $y(x_i)$ мәнін әрқайсысы қосылғыштың бір мәнін есептейтін екі функцияны қолдана отырып табыңыз

$$y(x_i) = f(x_i) = \underbrace{\left(\frac{1}{3} + \frac{1}{5} + \frac{1}{7} + \ldots + \frac{1}{2n+1}\right)}_{1\,қосылғыш} + \underbrace{\left(1 + \frac{\cos x}{1!}\right) + \left(1 + \frac{\cos 2x}{2!}\right) + \ldots + \left(1 + \frac{\cos mx}{m!}\right)}_{2\,қосылғыш}.$$

3. $x_i,\ i \in [0,\ 4],\ n = 8;\ m = 15$ берілген, экранға $y(x_i)$ мәнін әрқайсысы қосылғыштың бір мәнін есептейтін екі функцияны қолдана отырып табыңыз

$$y(x_i) = f(x_i) = \underbrace{\left(\frac{1}{3^2} + \frac{1}{5^2} + \frac{1}{7^2} + \ldots + \frac{1}{(2n+1)^2}\right)}_{1\,қосылғыш} + \underbrace{\left(1 + \frac{tgx}{1!}\right) + \left(1 + \frac{tg2x}{2!}\right) + \ldots + \left(1 + \frac{tgmx}{m!}\right)}_{2\,қосылғыш}.$$

4. x_i, $i \in [0, 4]$, $n = 15$; $m = 10$ берілген, экранға $y(x_i)$ мәнін әрқайсысы қосылғыштың бір мәнін есептейтін екі функцияны қолдана отырып табыңыз

$$y(x_i) = f(x_i) = \underbrace{\left(-\frac{2}{1!}+\frac{3}{2!}-\frac{4}{3!}+\ldots+\frac{(-1)^n(n+1)}{n!}\right)}_{1\ \text{қосылғыш}}+ \underbrace{\left(1+e^x\right)+\left(1+e^{2x}\right)+\ldots+\left(1+e^{mx}\right)}_{2\ \text{қосылғыш}}.$$

5. x_i, $i \in [0, 4]$, $n = 20$; $m = 10$ берілген, экранға $y(x_i)$ мәнін әрқайсысы қосылғыштың бір мәнін есептейтін екі функцияны қолдана отырып табыңыз

$$y(x_i) = f(x_i) = \underbrace{\left(-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\ldots+\frac{(-1)^n}{2n+1}\right)}_{1\ \text{қосылғыш}}+ \underbrace{\left(1+ln(|x|)\right)+\left(2+ln(|2x|)\right)+\ldots+\left(m+ln(|mx|)\right)}_{2\ \text{қосылғыш}}.$$

6. x_i, $i \in [0, 4]$, $n = 15$; $m = 12$ берілген, экранға $y(x_i)$ мәнін әрқайсысы қосылғыштың бір мәнін есептейтін екі функцияны қолдана отырып табыңыз

$$y(x_i) = f(x_i) = \underbrace{\left(\frac{1}{3^2}+\frac{1}{5^2}+\frac{1}{7^2}+\ldots+\frac{1}{(2n+1)^2}\right)}_{1\ \text{қосылғыш}}+ \underbrace{\left(1+\frac{tgx}{1!}\right)+\left(1+\frac{tg2x}{2!}\right)+\ldots+\left(1+\frac{tgmx}{m!}\right)}_{2\ \text{қосылғыш}}.$$

7. x_i, $i \in [0, 4]$, $n = 20$; $m = 18$ берілген, экранға $y(x_i)$ мәнін әрқайсысы қосылғыштың бір мәнін есептейтін екі функцияны қолдана отырып табыңыз

$$y(x_i) = f(x_i) = \underbrace{\left(\frac{1}{1!}+\frac{1}{2!}+\frac{1}{3!}+\ldots+\frac{1}{m!}\right)}_{1\ \text{қосылғыш}}+ \underbrace{\left(\frac{1}{2}-cos|x|\right)+\left(\frac{2}{3}-cos^2|x|\right)+\ldots+\left(\frac{n}{n+1}-cos^n|x|\right)}_{2\ \text{қосылғыш}}.$$

8. x_i, $i \in [0, 4]$, $n = 8$; $m = 6$ берілген, экранға $y(x_i)$ мәнін әрқайсысы қосылғыштың бір мәнін есептейтін екі функцияны қолдана отырып табыңыз

$$y(x_i) = f(x_i) = \underbrace{\left(\frac{1!}{x}+\frac{2!}{x^2}+\frac{3!}{x^3}+\ldots+\frac{n!}{x^n}\right)}_{1\ \text{қосылғыш}}+ \underbrace{\left(\frac{1}{2}-tg(|x|)\right)+\left(\frac{2}{3}-tg^2(|x|)\right)+\ldots+\left(\frac{m}{m+1}+tg^m(|x|)\right)}_{2\ \text{қосылғыш}}.$$

9. x_i, $i \in [0, 4]$, $n = 15$; $m = 20$ берілген, экранға $y(x_i)$ мәнін әрқайсысы қосылғыштың бір мәнін есептейтін екі функцияны қолдана отырып табыңыз

$$y(x_i) = f(x_i) = \underbrace{\left(\frac{x}{1!}+\frac{x^2}{2!}+\frac{x^3}{3!}+\ldots+\frac{x^n}{n!}\right)}_{1\ \text{қосылғыш}}+ \underbrace{\left(1+lg(|x|)\right)+\left(2+lg(|2x|)\right)+\ldots+\left(m+lg(|mx|)\right)}_{2\ \text{қосылғыш}}.$$

10. x_i, $i \in [0, 4]$, $n = 20$; $m = 15$ берілген, экранға $y(x_i)$ мәнін әрқайсысы қосылғыштың бір мәнін есептейтін екі функцияны қолдана отырып табыңыз

$$y(x_i) = f(x_i) = \underbrace{\left(0+\frac{\left(x-\frac{x}{2}\right)^2}{2!}+\frac{\left(x-\frac{x}{3}\right)^3}{3!}+\ldots+\frac{\left(x-\frac{x}{n}\right)^n}{n!}\right)}_{1\ \text{қосылғыш}}+ \underbrace{\left(\frac{e^x}{sin\,x}+\frac{e^{2x}}{sin\,2x}+\ldots+\frac{e^{mx}}{sin\,mx}\right)}_{2\ \text{қосылғыш}}.$$

Тоғызыншы сабақ. Полиномдар және интерполяция

Сабақтың жоспары

1. Полиномдармен орындалатын амалдар.
2. Деректерді интерполяциялау және апроксимациялау.
3. Полиномдық регрессия.
4. Фурье қатарының периодты функцияларын интерполяциялау.
5. Бірөлшемді кестелік интерполяция.
6. Екіөлшемді кестелік интерполяция.
7. Көрсету.

Matlab-та полиномдар векторының коэффициенттерімен анықталады, мысалы, $p = x^7 + 3.2x^5 - 5.2x^4 + 0.5x^2 + x - 3$ полиномі келесі вектор арқылы анықталады:

>>p = [1 0 3.2 -5.2 0 0.5 1 -3];

Векторының элемент саны, яғни полиномдар коэффициенті оның дәрежесінен әрқашан бір бірлікке үлкен, нөлдік коэффициенттер векторларда болуы керек.

polyval функциясы кейбір аргументтен полином мәнін табу үшін қолданылады:

>> polyval(p, 1)

ans = -2.5000

Аргумент матрица немесе вектор түрінде болуы мүмкін, бұл жағдайда полином мәнін элементтері бойынша табу қолданылады және нәтижесі аргументінің өлшеміне сәйкес матрицаны немесе векторды көрсетеді.

Полиномның барлық түбірін бірден табу *roots* функциясы арқылы жүреді. Аргумент ретінде полином коэффициенттерінен құрылған вектор алынады. *roots* функциясы полином түбірінің векторын қайтарады, ол кешенді сан болуы да мүмкін.

>> r = roots(p)

r =

-0.5668 + 2.0698i
-0.5668 - 2.0698i
0.5898 + 0.6435i
0.5898 - 0.6435i
0.6305 + 0.5534i
-0.6305 - 0.5534i

Полином түбірлерінің саны оның дәрежесіне сәйкес келеді.

Деректерді интерполяциялау және апроксимациялау

Аппроксимация, әдетте, мәліметтердің тәуелділігін немесе бірлігін басқа қарапайым немесе бірқалыпты тәуелділік арқылы суреттеу

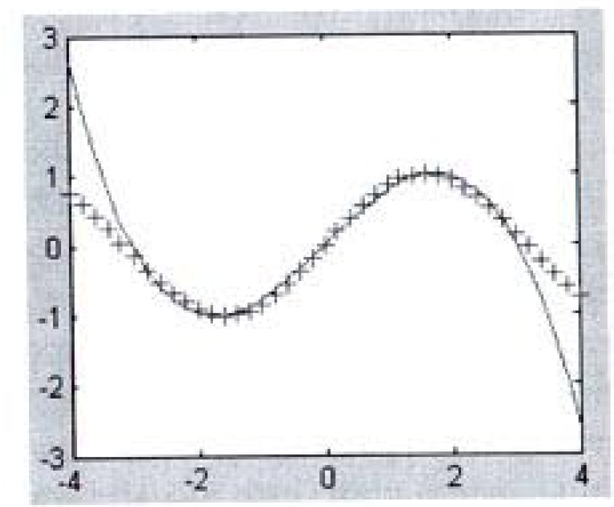
болып табылады. Көбінесе, деректер координаталары кесте түрінде берілген жеке түйіндердегі нүктелерде орналасады.

Аппроксимация нәтижесі түйін нүктесінен өтпеуі мүмкін. Керісінше, интерполяция мақсаты – ортадағы түйін нүктелеріндегі деректерді табу. Бұл үшін түйін нүктелеріндегі мәндері нүктелер координаталарымен сәйкес келетін функция қолданылады. Мысалы, сызықты интерполяциялық тәуелділікте *y(x)* түйін нүктелер бірбірімен түзу үзінділер арқылы бірігіп, аралық нүктелер құрайды. Ізделінді аралық нүктелер осы үзінділерде орналасқан деп есептеледі.

Интерполяцияның дәлдігін жоғарылату үшін парабола (квадратты интерполяция) немесе жоғары дәрежелі полином қолданылады. *Matlab* мәліметтерін өңдеу үшін интерполяцияның және аппроксимация әртүрлі функциялары қолданылады.

Полиномдық регрессия

Аппроксимацияның белгілі түрлерінің бірі – полиномдық. *Matlab* жүйесінде деректерді полиномдар арқылы ең кіші квадраттар әдісімен аппроксимациялау функциясы полиномдық регрессия анықталады. Бұл функция төменде көрсетілген (9.1-сурет):



9.1-сурет. *polyfit* функциясын қолдану мысалы

- *polyfit (x,y,n) – n* дәрежелі полином коэффициенттерінің векторын *p(x)* қайтарады, ол орта квадратты қателікпен *y(x)* функциясын аппроксимациялайды. Нәтижесінде ұзындығы *(n+1)*-ге тең вектор-жол алынады, онда полином коэффициенттері *x* дәрежесінің кему ретімен орналасқан;
- *[p,S] = polyfit(x,y,n)* полином коэффициенттері *p*-ны және *S* құрылымын *polyval* функциясымен бірге орындауға, бағалауға және қателік аралығын анықтауға қолданылады;
- *[p,S] = polyfit(x,y,n,mu)* полином коэффициенттері *p*-ны және *S* құрылымын *polyval* функциясымен бірге орындауға, бағалауға және қателік аралығын анықтауға қолданылады. Ол тек *x*-ті нормалау және масштабтау арқылы жүреді *xnorm = (x-mu(l))/mu(2)*, онда *mu(l) = mean(x)* және *mu(2) = std(x)*. Орталықтандыру және масштабтандыру тек *polyval* арқылы алынған дәрежелі көпмүшелігінің қасиетін жақсартпайды және де алынған аппроксимация алгоритімінің сапалық сипаттамасын жоғарылатады.

Мысалы:

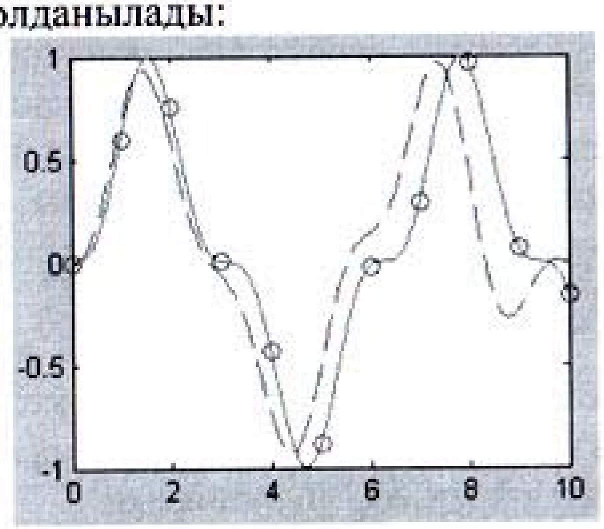
sin(x) функциясы үшін полиномиалды регрессия:

```
>>x = (-3:0.2:3)';
>>y = sin(x);
>>p = polyfit(x,y,3);
>>x = (-4:0.2:4)';
>>y = sin(x);
>> f = polyval(p,x);
>> plot(x, y, 'r+', x, f)
```

Фурье қатарының периодты функцияларын интерполяциялау

Интерполяция дегеніміз – *f(x)* функциясының мәнін түйін нүктелерінің аралығында есептеп шығару. Сызықты, квадратты және полиномиалдық интерполяция полиномиалдық аппроксимация кезінде орындалады. Периодты функциялар үшін оларды Фурье қатарының тригонометриялық функцияларымен интерполяциялау жақсы нәтижелер береді. Бұл үшін келесі функция қолданылады:

- *interpft(x,n)* – *n* бірқалыпты орналасқан нүктелерде анықталған периодты функциясының мәнінен тұратын *y* векторын қайтарады. Егер *length(x) = m* және *x*-тің дискреттеу интервалы *dx* болса, онда *y*-тің дискреттеу интервалы *dy = dx*m/n, n* саны *m*-нен кіші бола алмайды:



9.2-сурет. *interpft* функциясын қолдану мысалы

Мысалы (9.2-суретте көрсетілген)

```
>>x = 0:10; y = sin(x).^3;
>>x1 = 0:0.1:10; y1 = interpft(y,101);
>>x2 = 0:0.01:10; y2 = sin(x2).^3;
>>plot(x1,y1,'--'), hold on, plot(x,y,'o',x2,y2)
```

Кей жағдайда кесте түрінде берілген функцияны сплайндық интерполяциялау және аппроксимациялау өте ыңғайлы. Аралық нүктелер үшінші дәрежелі полином үздіктерінде ізделінеді – бұл кубтық сплайндық *интерполяция.* Сонымен бірге, әдетте, полиномдар тек координаталар түйіндерінің нүктелерінде оның мәндерінің бірдейлігімен ғана емес, түйін нүктелерінде бірінші және екінші туындылар үзіліссіз болатындай есептеледі. Мұндай әрекет түйін нүктелерінде бекітілген икемді сызғышқа тән *spline* (сплайн) аты да интерполяция (аппроксимация) түріне осыдан шыққан.

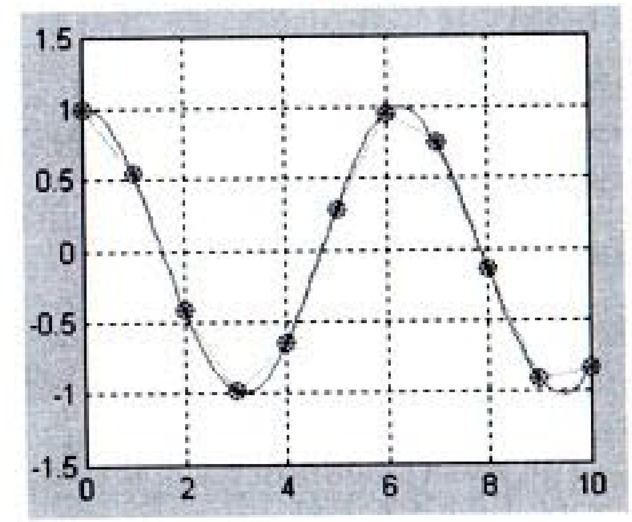
Бірқалыпты кестелі интерполяция үшін келесі функция қолданылады:

- *y1 = interp1(x,y,x1)* *x* және *y* векторлық интерполяциясынан алынған элементтері *x1* элементтеріне сәйкес *y1* векторын қайтарады. *x* векторы *y*-тің мәні берілген нүктені анықтайды.
- *y1 = interp1 (x,y,x1,method) method* параметрі арқылы интерполяция әдісін таңдауға мүмкіндік береді:
  - *'nearest'* – баспалдақты интерполяция;
  - *'linear'* – сызықты интерполяция;
  - *'spline'* – кубтық сплайндық-интерполяция;
  - *'cubic'* – Эрлисттің көпмүшелік интерполяциясы.

Интерполяцияның барлық әдісі *x*-тің монотонды өзгеруін талап етеді. *x* векторы теңдей нүктелерін белгілегенде тез интерполяциялауда *'*linear'*, *'*cubic'*, *'*nearest'*, немесе *'*spline'* әдістерін колданады. Бұл жағдайда әдіс жұлдызшамен белгіленеді.

Мысалы (косинус функциясының интерполяциясы, 9.3-сурет)

```
>> x = 0:10; y = cos(x);
>>x1 = 0:0.1:10;
>> y1 = interp1(x,y,x1);
>> plot(x, y,'*',x1,y1,'g'), hold on
>> yi = interp1(x,y,x1,'spline');
```



9.3-сурет. *Interp 1* функциясын қолдану мысалы

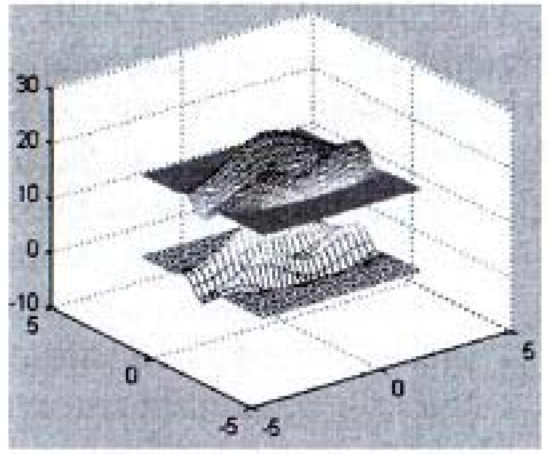
Екі өлшемді кестелік интерполяция

Екі өлшемді интерполяцияның мақсаты түйінді нүктелердің кеңістігіне жақын орналасқан аралық нүктелердің кейбір тәуелділіктерін *z(x,y)* табу. Екі өлшемді кестелі интерполяция үшін *interp2* функциясы қолданылады:

- *Z1 = interp2(X, Y ,X1, Y1)* – *X, Y, Z* матрицалары арқылы екіөлшемді тәуелді интерполяция арқылы алынған *X1, Y1* аргументтерінде берілген функция мәнінің нүктелерінен тұратын – *Z1*-матрицасын қайтарады. *x* пен *y* монотонды болуы және *meshgrid* функциясы арқылы алынғандай форматта болу керек. *x* және *y* матрицалары z мәні берілген нүктелерді анықтайды.
- *Z1 = interp2(Z, X1, Y1)* – *X = 1:n* және *Y = 1:m* болатынындай алады, ондағы *[m, n] = size(Z).*
- *Z1 = interp2(X,Y,Z,X1,Y1,method)* – *method* опциясы арқылы интерполяция әдісін шақырады:
- *'nearest'* – көрші нүктелер бойынша интерполяция;
- *'linear'* – сызықты интерполяция;

- *'cubic'* – кубтық интерполяция (Эрмиттің көпмүшелігі);
- *'spline'* – сплайндық интерполяция.

Егер Y пен X біркелкі орналасқан нүктелер векторлары болса, тез интерполяция үшін *'*linear', '*cubic' немесе '*nearest'* әдістерін қолдану керек 9.4-сурет *interp2* функциясының екіөлшемді интерполяцияда қолданылуын көрсетеді (*peaks* функциясы мысалында).



9.4-сурет. *Interp2* функциясын қолдану мысалы

*>> [X,Y] = meshgrid(-3:0.25:3); Z = peaks(X/2,Y*2);*
>> [X1,Y1] = meshgrid(-3:0.1:3); Z1 = interp2(X, Y, Z, X1, Y1);
>> mesh(X,Y,Z), hold on, mesh(X1, Y1, Z1+15), hold off

Тапсырмалар

1. $p = x^7 - 2x^6 + 4x^5 - 2x^4 + 3x^3 + x^2 + 3$ полиномы берілген. Полиномның 1, 2, 6, 10 нүктелерінде мәнін табыңыз. Полином түбірін табыңыз.
2. Аргумент мәнінің массиві берілсін:

$x = [1\ 2\ 3\ 4\ 5\ 6\ 7\ 8]$,

ал функция мәнінің массивінің сәйкес мәні:

$y = [-1.1\ 0.2\ 0.5\ 0.8\ 0.7\ 0.6\ 0.4\ 0.1]$.

Интерполяцияның әр түрлі әдістерін қолдана отырып, аппроксимациялы функцияны табу және алынған функциялардың графигін екі әдіс арқылы табу: бір координаталық жазықтықта, бір графиктік терезеде ішкі графиктер түрінде көрсетіңіз.
3. Екі мәтіндік файлды x және y мәндері үшін құру керек, x уақытты анықтайды, кейбіреулерінде өлшемдер жүргізілген, ал y өлшемді мәнді құрайды. Аппроксимациялы функцияның мәнін берілген интервалды мәтіндік файлда сақтау керек.
4. 0.3 қадамымен берілген функцияның мәнін беру керек, функция мәнін есептеп, берілген нүктелерде интерполяция жүргізіп, шығыс графигін және аппроксимациясын құрыңыз

$z(x,y) = 4\sin(2\pi x)\cos(1.5\pi y)(1-x^2)y(1-y)$.

5. №4 тапсырманы мына функция үшін орындаңыз

$z = e^{-x^2-y^2}(x-1)^2\sin 2\pi y,\ x, y \in [-1, 1]$.

Оныншы сабақ. *Matlab* пакетінің *Simulink* бағыныңқы жүйесі

Сабақтың жоспары

1. *Simulink* бағыныңқы жүйесінің негізгі қасиеттері.
2. *Simulink* бағыныңқы жүйесін іске қосу.
3. *Simulink* блоктарының кітапханасы.
4. *Sources* кітапханасы.
5. *Sinks* кітапханасы.
6. *Discrete* кітапханасы.
7. *Continuous* кітапханасы.
8. *Functions & Tables* кітапханасы.
9. *Nonlinear* кітапханасы.
10. *Math* кітапханасы.
11. *Signals & Systems* кітапханасы.
12. Мысалдар.

Simulink пакеті динамикалық сызықтық емес жүйелердің іс-әрекетін зерттеуге және модельдеуге мүмкіндік береді. Зерттелінетін жүйелердің сипаттамаларын енгізу, қарапайым стандартты ұяшықтарды біріктірудің графикалық жинақтау үрдісі арқылы диалогтық режимде өтеді. Мұндай жинақтаудың нәтижесінде зерттелінетін жүйенің моделі пайда болады. Ол модель *S*-моделі деп аталады. Модель *.mdl* кеңейтілуімен файлда сақталады.

Simulink пакетінде модельдерді құру *Drag-and-Drop* технологиясын қолдануға негізделген. *S*-моделін құру үшін қолданылатын "кірпіштер" ретінде *Simulink* кітапханасында сақталатын модульдер (блоктар) қолданылады. Әрбір *S*-модель тармақтық құрылымға ие, яғни ол саны шексіз төменгі дәрежелі модельдерден тұрады. Модельдеу кезінде жүйеде жүріп жататын процестерді бақылауға мүмкіндік бар. Ол үшін *Simulink* кітапханасына кіретін арнаулы терезелер қолданылады.

Simulink терезесінің негізгі қасиеттері

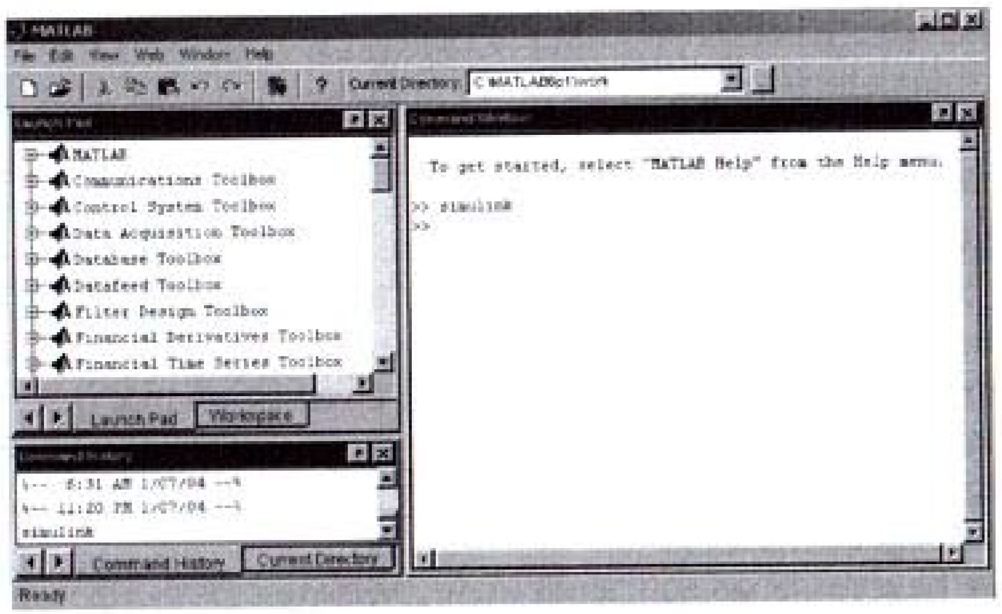
Simulink бағыныңқы жүйесі:

- Сызықтық, сызықтық емес, дискретті, гибридті, үздіксіз жүйелерді модельдеуге мүмкіндік береді;
- Құрамында жаңа жүйелерді құру үшін қолдануға болатын ауқымды блоктар кітапханасы бар (үздіксіз элементтер, дискретті элементтер, математикалық функциялар, сызықсыз элементтер, сигналдар көзі, көрсету әдістері);
- Блок-диаграммаларды құрамдас блоктарға біріктіреді. Ол өз алдында модельдің тармақтық құрылымын қамтамасыз етеді;

- Қолданушы өзі анықтайтын блоктар мен кітапханаларды құруға қажетті құралдары бар;
- Уақыт бойынша құрылымын өзгертетін бағыныңқы жүйелерді жобалауға мүмкіндік береді.

Simulink бағыныңқы жүйесін іске қосу

Модельді құру алдында *Matlab* жүйесін жүктеп, *Simulink* бағыныңқы жүйесін іске қосу керек. *Simulink* бағыныңқы жүйесін іске қосу негізгі терезеде іске асады. Ол үшін терезенің үстінің оң жағында орналасқан жүйені қосу батырмасын тышқанның сол жақ батырмасын басу арқылы іске асады немесе 10.1-суретте көрсетілгендей негізгі терезеде *Simulink* командасын теру керек.



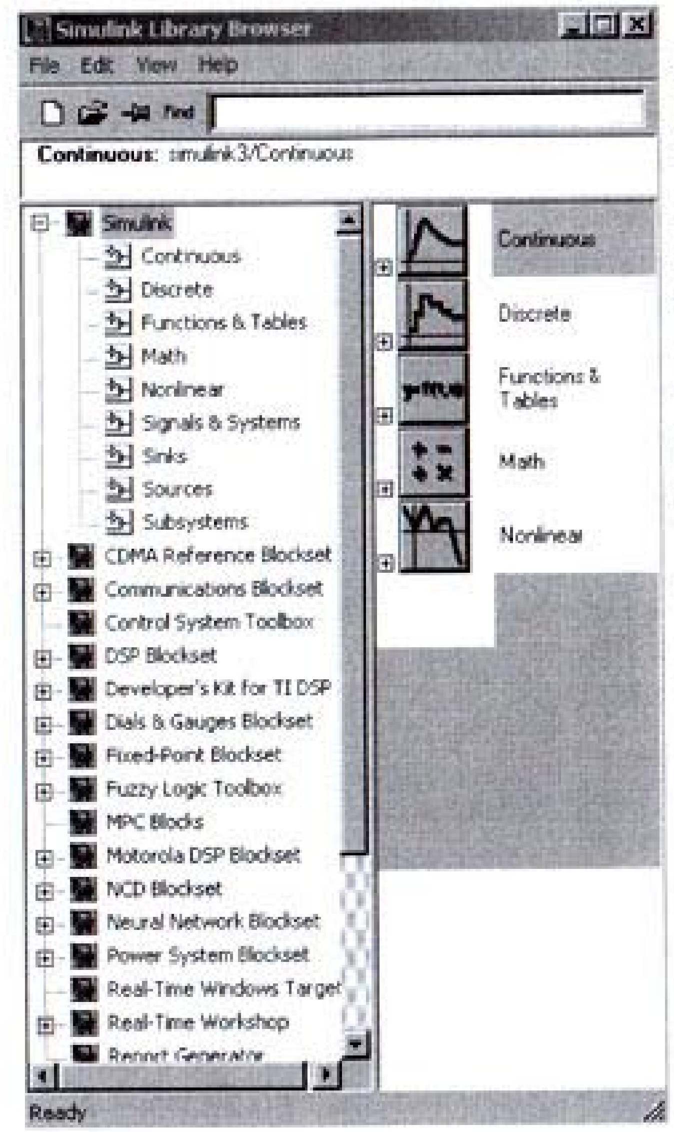
10.1-сурет. *Matlab* ортасынан *Simulink* пакетін шақыру

Бұл екі әдісте де *Simulink Library Brouser* (*Simulink* кітапханаларын көріп шығу жүйесі) ашылады, 10.2-суретте. Бұл терезенің үстіңгі бөлігіндегі шеткі екі сол жақ батырмалар сәйкесінше алдыңғы және жаңа модельді ашу үшін қолданылады.

Simulink моделін құрастыру процесін керекті параметрлерді енгізуді және құрастыруды керек етеді. Құрастыру *Simulink* кітапханаларының керекті блоктарын таңдап, олардың ашылған терезеде орналастыруға және блокаралық байланыстарды енгізуге негізделген. Ары қарай әрбір блок үшін модельденетін жүйе талаптарына жауап беретін сәйкес параметрлер қойылады.

Simulink блоктарының кітапханалары

Simulink пакетінің моделін құру процесіндегі негізгі құрамдас материалы блок болып табылады. Блокқа визуалды объектілер жиынтығы енеді. Ол арқылы функцияларды байланыс сызықтар модулін біріктіре отырып, әрбір құрылғының блок-үрдісін құруға болады.



10.2-сурет. *Simulink* бағыныңқы жүйесінің кітапханалар құрылымы

Блоктар кітапханасы тоғыз бөлікке бөлінген:

1. *Continuous*
2. *Discrete*
3. *Function and Tables*
4. *Math*
5. *Nonlinear*
6. *Signals & Systems*
7. *Sinks*
8. *Sources*
9. *Subsystems*

Байланыс арналары блоктар жеке даналары бірегей үрдіге біріктіріледі. Ол модель құру терезесінде немесе жеке даналары терезеде алдын-ала енгізілген құрылғы контекстерде сол арада құрылады.

Керекті блокты кітапханадан модель құру терезесіне ауыстыру үшін, оны алдымен стандартты *Simulink* блоктарының тізімінен табу керек.

Ол үшін *Simulink Library Brouser* терезесінде *Simulink* пунктін таңдап, пайда болған кітапханалар тізімінде сәйкес пунктті белгілеп, ашу керек. Орын ауыстыру үшін тышқан меңзерін керекті блокқа орнатады. Одан кейін, тышқанның сол жақ батырмасын баса отырып, блокты модель терезесіне ауыстырамыз. Модель құру терезесінде блок пайда болғаннан кейін, оған сәйкес параметрлерді орнатуға болады. Ол үшін тышқанның сол жақ батырмасын екі рет блоктың белгішені басып, керекті параметрді орнату керек.

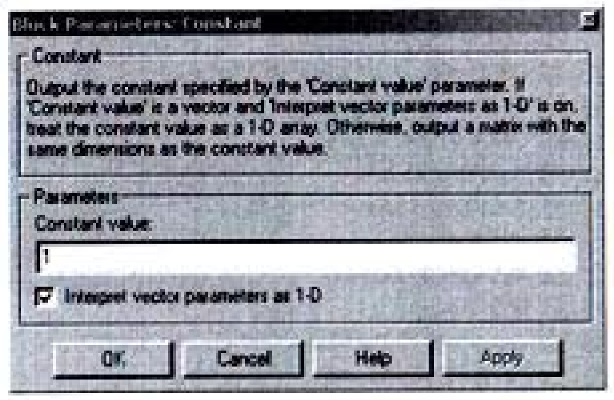
Sources кітапханасы

Кітапхананың құрамына он алты түрлі блок кіреді. Оның ішінде жиі қолданылатындары мыналар:

- *Clock* (сағат) – үздіксіз уақыттық сигналдың генераторы;
- *Constant* (тұрақты әрдайым өлшемнің көзі);
- *Digital clock* (сандық сағат) – дискретті уақыттық сигналдың көзі;
- *Discrete Pulse Generator* (дискретті импульсті генератор) – дискретті импульсті сигналдардың генераторы;

- *Pulse Generator* (импульсті генератор) – импульсті сигналдардың генераторы;
- *Random Number* (кездейсоқ сан) – дискретті сигналдардың көзі, оның амплитудасы кездейсоқ өлшемді болып келеді;
- *Signal Generator* (генератор сигналы) – бос формадағы үздіксіз сигналдың генераторы;
- *Step* (такті) параметрлері берілген бірліктік дискретті сигнал көзі;
- *Sine Wave* – гармониялық толқулардың генераторы;
- *From File* (файлдан ензізу) – *MAT*-файлда сақталынатын мәліметтерді *S*-модельге енгізу;
- *From Workspace* (жұмыс аумағынан енгізу) – *Matlab* жұмыс аймағынан мәліметтерді модельге енгізу.

Constant блогы скалярлы немесе векторлы сигналдардың құрылуын қамтамасыз етеді. Сигналдың мәні блокты құру терезесінің алаңында *Constant value* және де келесі форматта берілуі мүмкін (10.3-сурет):



10.3-сурет. *Constant* блогының икемдеу терезесі

- Сандық тұрақты түрінде;
- Вектор түрінде (тік жақшаға алынған сандар тізбегі);
- Матрица түрінде (тік жақшаға алынған векторлар тізбегі);
- Есептелінетін өрнек түрінде, оның ішінде қолданушымен жазылған функциялар түрінде немесе *Matlab*-тың кітапханалық функциялары ретінде.

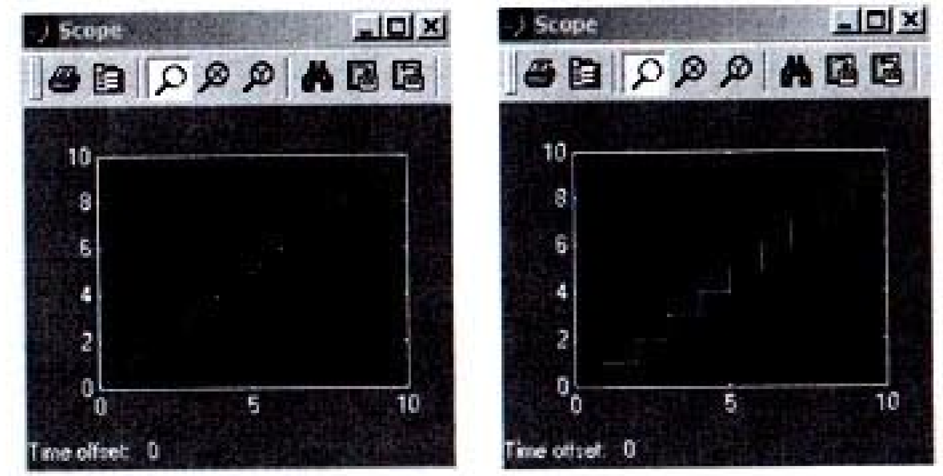
Digital clock блогының бір ғана құру параметрі бар – ол модельді уақыттың өзгеру қадамының өлшемін *Sample time* (10.4-сурет) көрсетеді. Бұл блок дискретті жүйе модельдерінде қолдануға бағытталған.

10.4-сурет. *Clock* блогының икемдеу терезесі

Clock блогы модельдеудің келесі қадамының модельдік уақытының ағымдық мәнін қамтамасыз етеді және үздіксіз жүйелерді модельдеу кезінде қолданылады. Блок екі құру параметрлеріне ие (10.4-сурет):

- *Display time* (ағымдық уақыт көрсету) – жалауша тұрса, онда блоктың ішінде модельдік уақыттың ағымдық мәнін сандық формада енгізу;
- *Decimation* (дискреттік) – бүтін сан, модельдік уақыттың блогының құрылу периодтылығын анықтайды.

Берілген параметрлерге тәуелсіз, блок арқылы құрылатын уақыт мәні үздіксіз кезекті құрады. 10.5-суретте *Scope* блогының 2 с модельдік уақыттағы "көрсетулер" бейнеленген, бұл *Clock* (сол жағында) және *Digital Clock* (оң жағында) блоктарының айырмашылығын көрсетіп тұр.



10.5-сурет. *Clock* және *Digital Clock* блоктарының құрған модельдік уақыт мәндері

Sinks кітапханасы

Бұл кітапхананың құрамына блоктың шығу кезіндегі сигналдарды бейнелеу құралдары бар. Оларды үшке бөлуге болады.

Модельдеу кезінде "көрме терезелері" ретінде қолданылатын блоктар:

- *Scope* блогы;
- *XYGraph* блогы (өлшемді график) – координаталардың тік бұрышты жүйесінде екі өлшемді графиктер құруға көмектеседі;
- *Display* блогы – өлшемдік мәндерді сандық түрде бейнелейді.

Модельдеудің шыққан немесе аралық нәтижелерін сақталуын қамтамасыз ететін блоктар:

- *To File* блогы (файлға жазу);
- *To Workspace* блогы (жұмыс аймағына жазу).

Модельдеуді басқару блогы *Stop Simulation* (модельдеуді тоқтату) – кейбір шарттары орындалғаннан кейін модельдеуді тоқтата алады.

Scope блогы зерттеушіні қызықтыратын жүйенің сипаттамаларының өзгеру динамикасын модельдеуді бақылау процесіне мүмкіндік береді. Ордината осі бойынша бақыланатын өлшемнің мәндері, ал абцисса осі бойынша қадағаланатын модельдік уақыттың мәні салынады. Блок-диаграмма *Scope* блогының кірісіне векторлық өлшем түсіп жататындай етіліп құрылуы мүмкін. Бұл жағдайда терезеде әрбір вектор элементі үшін оның өзгеру динамикасын көрсететін жеке қисық құрылады.

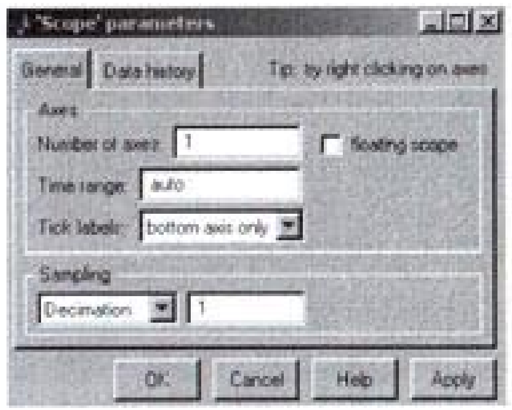
Scope блогының *Properties* пернесін басқанда, екі пункті бар қасиеттер терезесі ашылады:

- *General* (жалпы касиетер) – графиктерді шығару форматын басқару параметрлері;
- *Data history* (мəліметтерді хаттамалау) – мəліметтер графиктерінде көрінетін *Matlab* жұмыс аймағына жазылу параметрлері.

General қабықшасы интерфейстің келесі элементтерін қамтиды (10.6 сурет):

- *Number of axes* (графиктер саны) – *Scope* терезесінде құрылатын графиктердің санын енгізу үшін қолданылады;
- *Time range* алаңы (уақыттық диапазон) – уақыт осі бойынша (*X* осі) диапазонының шекаралық мəнін енгізу үшін қолданылады;
- Ашылатын тізім *Tick labels* (тактілік белгілер) – *Scope* терезесінде бірнеше графиктер пайда болса, қолданылады.

- Ашылатын тізім *Sampling* (периодтылық) – графиктердің суреттелу периодтылығын басқару варианттарын таңдау үшін қолданылады:
  - *Decimation* (дискретік) – шыққан нəтижелерді "шығару" коэффициенті ретінде түсіндіріледі;
  - *Sample time* (эталондық уақыт) – периодтылық модельдеу сеансына берілген модельдік уақыттың қадам өлшемімен, мөлшерімен анықталады.



10.6-сурет. *General* қабықшасы

Discrete кітапханасы

Бұл кітапхананың құрамында дискреттік уақытта жұмыс істейтін *Discrete-Time Integrator*, *Discrete Filter* сияқты блоктары енеді. Кітапхана құрамына сипаттаушы жүйелерді, əртүрлі теңдеулерді шешетін блоктарды қамтиды.

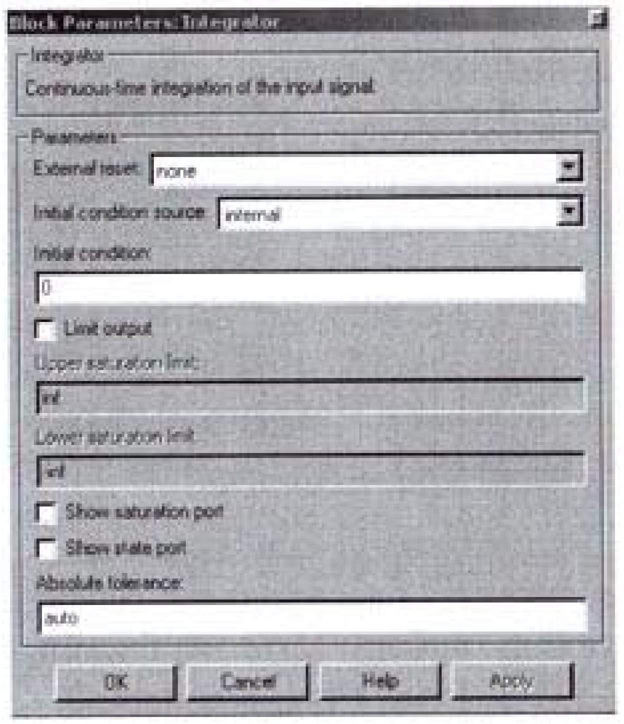
Continuous кітапханасы

Берілген кітапхана өзіне *Intergrator*, *Derivative* (Дифференциатор), *State-Space* жəне т.б. үздіксіз элементтерді қамтиды.

Integrator блогы шығу сигналының "өмір сүру уақытын" есептейді жəне модельденетін жүйенің уақыттық мінездемелерін анықтау үшін қолданылады. Блоктың келесі құру параметрлері бар (10.7-сурет):

1. Ашылатын тізім *External reset* (сыртқы басқарушы сигнал) басқару əдісін таңдауға мүмкіндік береді:
   - *None* (жоқ) қосымша басқарушы сигнал қолданбайды;
   - *Rising* (өрлеу) басқару үшін арнайы сигнал қолданылады;
   - *Falling* (төмендеу) басқару үшін төмендететін сигнал қолданылады;

   - *Either* (əрбір) блоктың жұмысына басқарушы сигналдық амплитудасының əрбір өзгерту əсер етеді.



10.7-сурет. *Intergrator* блогының икемдеу терезесі

2. Ашылатын тізім *Initial condition source* (бастапқы күй қайнар көзі) екі мəннің біреуін таңдауға мүмкіндік береді:
   - *Internal* (ішкі) – сумматордың бастапқы мəнінің өз нұсқауы қолданылады;
   - *External* (сыртқы) – бастапқы мəн-мағынаны орнату сырттан жүргізіледі;
   - *Initial condition* алаңында мəн сандық тұрақты түрінде немесе есептелінетін өрнек түрінде енгізіледі.

4. *Limit output* жалауша қалған төрт параметрлер қолданылатынын анықтайды:
   - *Upper Saturation limit* (үстіңгі шекті мəн), үнсіздік бойынша параметр мəні шексіз (*inf*);
   - *Lower saturation limit* (төменгі шекті мəн) үнсіздік бойынша параметр мəні шексіз (*-inf*);
   - *Show Saturation port* жалаушасы (толықтық портын көрсету);
   - *Show State port* жалаушасы (қалып-күй портын көрсету);

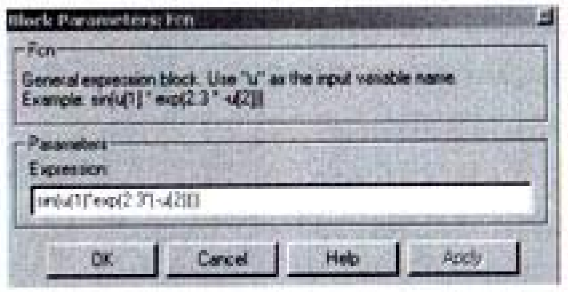
5. *Absolute tolerance* (есептеулердің нақтылығы).

Function & Tables кітапханасы

Берілген кітапхана құрамына математикалық функциялар кітапханасында жоқ функциялармен жұмыс істеуге, сонымен қатар кестелік функциялармен жұмыс істеуге мүмкіндік беретін блоктар енеді. Оған *Fcn*, *Matlab Fcn* (*Matlab* пакетінің функциялары), *S-Function* (*S*-функция) жəне т.б. жатады.

Fcn блогы – бұл əмбебап "есептеуші" блок. Құру параметрі ретінде əрбір есептеуіш мəнді енгізуге болады, оның аргументі кіріс сигналының мəні болады.

Кіріс сигналы белгілеу үшін *u* символы қолданылады. Егер кіріс сигналы вектор болып келсе, онда оның жеке элементтері үшін жүргізілетін операцияларға аргумент міндетті түрде берілу керек. Мысалы, екі кіріс сигналының қосындысы мына түрде жазылу керек (10.8-сурет): *u(1)+u(2)*.



10.8-сурет. *Fcn* блогының икемдеу терезесі

Nonlinear кітапханасы

Бұл кітапхананың құрамында сызықтық емес функциялармен жұмыс істейтін блоктар бар. "Қайта қосу блоктар" жеке топ құрады, яғни сигналды жіберу бағытын басқарушы блоктар. Ондай блоктар төртеу:

- *Switch* (қайта қосушы құрал);
- *Manual switch* (қолдан қосатын құрал);
- *Multiport switch* (көп кірісті қайта қосқыш);
- *Reley* (реле).

Switch блогының үш кірісі бар: екі ақпараттық (бір және үш) және бір басқарушы.

Жұмыстың мәні келесіде. Егер екінші кіріске түсетін ақпараттың амплитудасы берілген кіреберіс мәнінен кіші болмаса, онда блок шығысына бірінші кірістен сигнал жіберіледі, кері жағдайда – үшінші кірістен түседі.

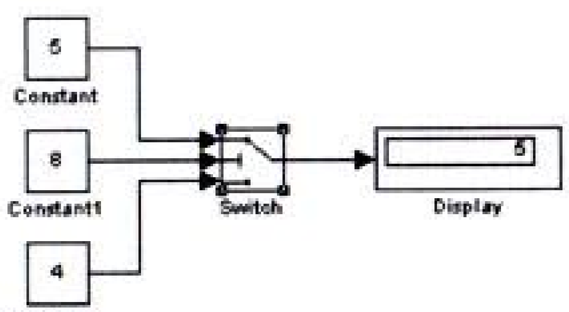
Блоктың жалғыз құру параметрі бар – *Threshold* (кіреберіс). Ол сандық тұрақты немесе есептелінетін мән ретінде берілуі мүмкін. *Switch* блогының жұмыс істеу периодтылығы басқарушы кіріс жеріне қосылған *Sample time* эталондық уақыт параметрінің мәнімен анықталады.

Switch блогының қолдану мысалы келтірілген (10.9-сурет) (*Threhold* параметрінің мәні беске тең).

Айта кеткен жөн, сигналды жіберу бағыты өзгерсе, блок ішіндегі "ұстатқыш" жағдайы өзгермейді.

Manual Switch блогының құру параметрлері жоқ және екі кіріс портынан біреуін "қолмен" таңдауға мүмкіндік береді, оның сигналы шығу блогына жіберіледі.

Manual Switch блогының шығу портын кіріс портымен байланыстыратын "ұстатқыштың" орны ауысуы үшін, блокты екі рет басу керек.



10.9-сурет. *Switch* блогын қолдану мысалы

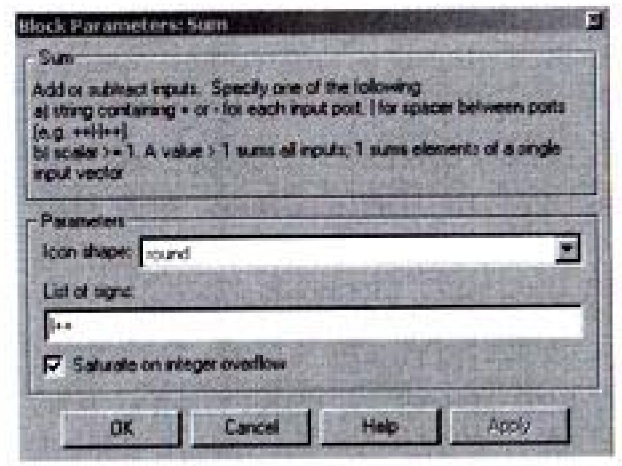
Math кітапханасы

Берілген кітапхана құрамына *Abs* (абсолютті мәндер), *Combinatorial Logic* (комбинаторлық логика), *Complex to Real-Imag* және т.б. стандарты математикалық функциялары бар блоктар енді.

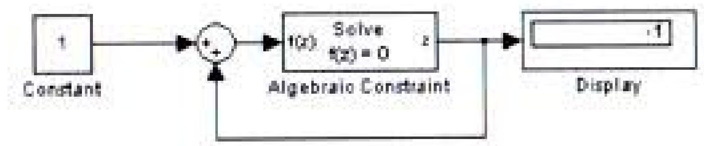
- *Sum* блогы кіріс сигналдарын екі тәртіпте жүре есептеуін жүргізеді (10.10, 10.11-сурет);
- Кіріс сигналдарын қосу (әр түрлі таңбалармен) кірісіне келіп түскен вектор элементтерін есептеу.

Product блогы бірнеше кіріс сигналдарын көбейту немесе бөлуді орындай, кіріске түсетін сигналды көбейткіш функциясын орындайды.

Algebraic Constraint блогын қолданудың қарапайым әдісі 10.12-суретте көрсетілген. Бұл мысалда ол $z + 1 = 0$ теңдеуін шешу үшін қолданылған.



10.10-сурет. *Sum* блогының икемдеу терезесі



10.11-сурет. *Sum* блогын қолдану мысалы

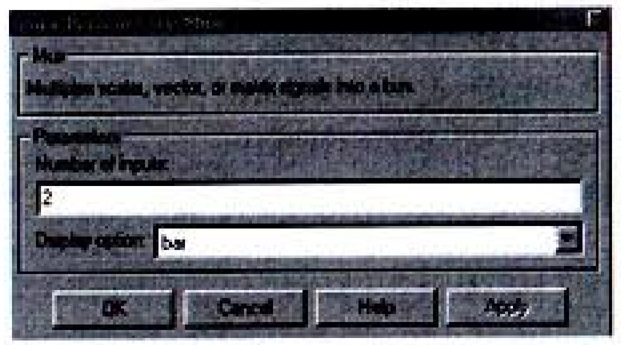
10.12-сурет. *Algebraic Constraint* блогын қолдану мысалы

Signals & Systems кітапханасы

Құрылымы бойынша *Signals & Systems* бөлімі – ең үлкен және *Simulink* кітапханаларының имитациялық модельдеу бөлімінің тиімдісі. Оның құрамында отыз блоктар бар. Оның ішінде жиі қолданылатындары мыналар:

- *In* (кіріс порты) және *Out* (шығыс порты) – модельдің бағынынқы жүйелері аралығы мәліметтермен алмасуды қамтамасыз етеді, сонымен қатар *Matlab* жұмыс аймағы мен модель арасында жүреді;
- *Bus Selector* (шина селекторы) – сигналдар тобының ішінен бір берілген сигналды ерекшелейді;
- *Mux* (араластырғыш) – кіріс сигналдарын бір векторлық сигналға біріктіреді;
- *Demux* (бөлгіш) – *Mux* блогының функцияларына қарама-қарсы функцияларды іске асырады;
- *Subsystems* – бағынынқы жүйелерді құрайтын блок.

Mux блогы кіріс шамаларын бір сызықтық векторға біріктіреді. (10.13-сурет). Мұндағы кіріс шамалары скалярлық та, векторлық та бола алады. Нәтижедегі вектордың өлшемі кіріс портына түсетін элементтердің барлық сандарының қосындысына тең.



10.13-сурет. *Mux* блогын қолдану мысалы

Блоктардың қосылуы және көшірмесін құру

Блоктарды жүйеге қосу үшін олардың шығыс және кіріс порттарын біріктіру керек, олар блоктардың пиктограммаларында ">" таңбасымен белгіленген. Екі блокты өзара біріктіру үшін, тышқанның меңзерін қосылушы блоктың біреуінің портына әкеліп, тышқанның сол жақ батырмасын баса отырып, меңзерді басқа блоктың портына әкеліп, ауыстыру керек.

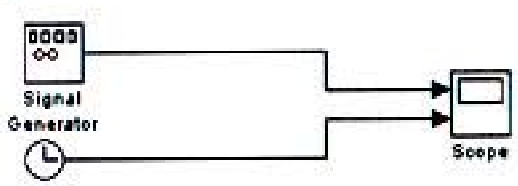
Көшірмесін құру үшін модель терезесіндегі блокқа меңзерді әкеліп, пернетақтада <*Ctrl*> батырмасын басып тұрып, тышқанның сол жақ батырмасын басу керек. Содан кейін, батырманы басқан қалпында меңзерді керекті орынға ауыстырып, батырмаларды жіберу керек.

Модельді құру кезінде блоктарды өзара қосып қана қоймай, байланыстырушы желілерді тармақтау керек. Қай блоктың кіріс блогы мен бар желінің арасындағы байланысты орнату <*Ctrl*> батырмасын басып тұрып жүзеге асырылады.

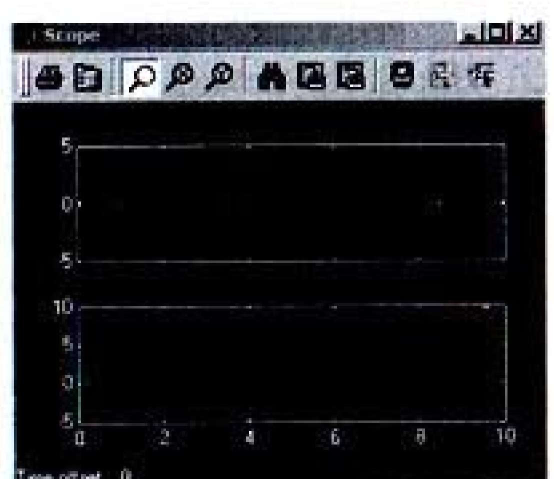
Simulink пакетін қолдану мысалдары

1-мысал

10.14-суретте *Sources* кітапханасының блоктарының жұмысы көрсетілген: *Signal Generator* және *Clock*. Нәтижелерді көріп шығу үшін *Scope* блогы қолданылады. (10.15-сурет).



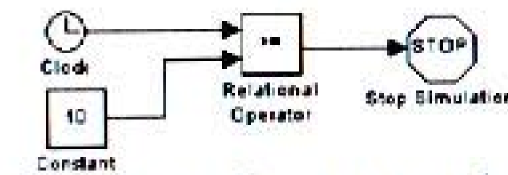
10.14-сурет. *Scope* блогын қолдану мысалы



10.15-сурет. *Scope* блогын көру терезесі

2-мысал

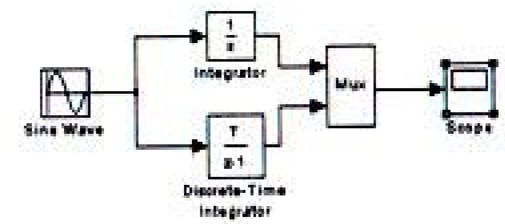
Келесі мысал *Sinks* кітапханасының *Stop* блогының қолданылуы көрсетілген. Уақыт мәні *10с* ≤ болған жағдайда, модель жұмысын тоқтатады.



10.16 сурет. *Stop* блогының көмегімен модельдеуді тоқтату

3-мысал

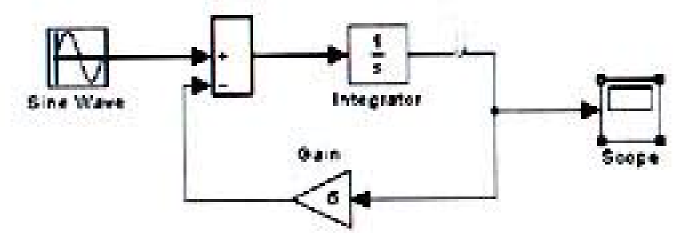
Бұл мысалда *Integrator Disclete-Time Integrator* блогының қолдануымен синусоида бірігеді.



10.17-сурет. Дискретті және үздіксіз құрылымдарды қолдану мысалы

4-мысал

Дифференциалдық теңдеуі берілген. Оның сандық шешімін құру керек. Бұл есепті шығару үшін *Integrator* блогын қолдану керек. Оның кірісіне *x'* келеді де, ал шығысына *x* келеді. Басқа 2 блок *–sum* және *Gain* – жазылған теңдеуге сәйкес *x'* мәнін құру үшін керек. (10.18-сурет).

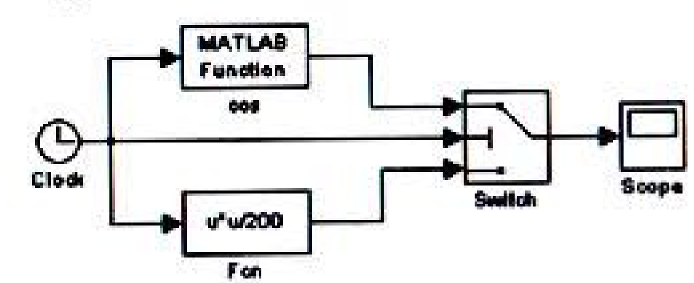


10.18-сурет. Екінші ретті сызықтық дифференциалдық теңдеуді шешу мысалы

5-мысал

Алғашқы *20 с* ішінде $t^2/200$ мәнін, қалған уақытта – *cost* мәнін қабылдайтын сигнал алу керек. Бұл үшін *Switch* блогын қолданған жөн. *Switch* блогында бастапқы шама 20, 1-ші кіріске "ара тәріздес" сигнал, 2-ге – сағат сигнал, 3-не тікбұрышты толқын қойылады.

Алғашқы 20 с-та уақыт мәні кіреберіс мәннен кем болады, сондықтан 3 (төменгі) кіріс белсенді болады. Уақыт кіреберістен өткеннен кейін 1-ші кіреберіс белсенді бола бастайды. (10.19-сурет).

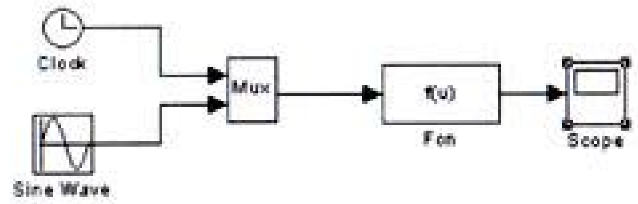


10.19-сурет. *Switch* блогының көмегімен модель өзгерісін орындау

6-мысал

Келесі теңдеуді құру керек болсын: *cos(u(1)*exp(2.3*(-u(2))))*.

Мұнда *Fcn* блогын қолданған жөн. Блокты құруда қажетті функцияларды сипаттап алу керек. Оны *Scope* блогының шығысына қосылып көруге болады. (10.20-сурет).



10.20-сурет. Арнайы *Fcn* блогын қолданып жаңа функцияларды құру

Тапсырма

Simulink пакетінің компоненттерін берілген материал негізінде зерттеп, жоғарыда келтірілген мысалдарды орындаңыз.

Он бірінші сабақ. Бағыныңқы жүйелерді құру

Сабақтың жоспары

1. *Subsystem* блогын қосу арқылы бағыныңқы жүйелерді құру.
2. Бар блоктарды топтау арқылы бағыныңқы жүйелерді құру.
3. Мысалдар.

Егер модельдің блок-үрдісі өте күрделі және үлкен болса, онда блоктарды бағыныңқы жүйеге топтастыру арқылы жеңілдетуге болады. Бағыныңқы жүйелерді қолдану келесі мүмкіндіктерді береді:

- модель терезесінде шығатын блок көлемі қысқартылады;
- функционалды байланысқан блоктарды бір топқа біріктіруге мүмкіндік туады;
- тармақтық блок-үрділерінің құрылуына мүмкіндік туады.

Бағыныңқы жүйені екі әдіспен құруға болады:

- *Subsystem* блогын модельге қосып, содан кейін бұл блокқа кіріп, пайда болған бағыныңқы жүйе терезесіне келесі бағыныңқы жүйені құру;
- Модельдің блок-үрділерінің бөлігін ерекшелеп, бір бағыныңқы жүйеге біріктіру.

Subsystem блогын қосу арқылы бағыныңқы жүйені құру

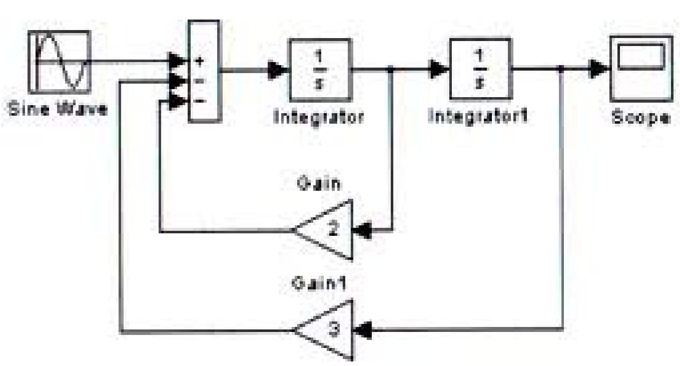
Бұл жағдайда:

1. *Subsystem* блогын *Connections* бөлімінен тартып, модель терезесіне көшіру керек.
2. Блок-үрдідегі блоктың бейнесіне екі рет басу арқылы *Subsystem* блогының терезесін ашу керек.
3. Модельдің бос терезесінде бағыныңқы жүйе құрылады. Мұнда бағыныңқы жүйенің кірісі мен шығысын құру *In* және *Out* блоктарын қолдану арқылы жүзеге асады.

Бар блоктарды топтастыру арқылы бағыныңқы жүйені құру

Егер блок-үрді құрамында бағыныңқы жүйеге біріктіретін блоктар бар болса, онда соңғысын мына түрде құруға болады:

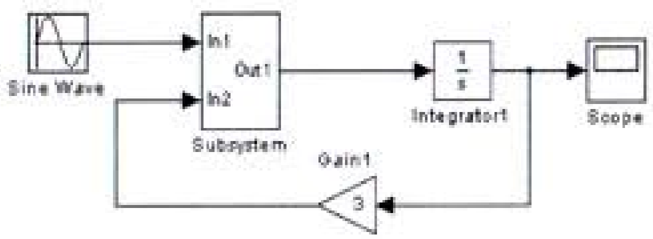
1. Бағыныңқы жүйе құрамына енгізу керек болатын блоктарды және оларды біріктіретін сызықтарды жиек арқылы ерекшелейді (11.1-сурет).



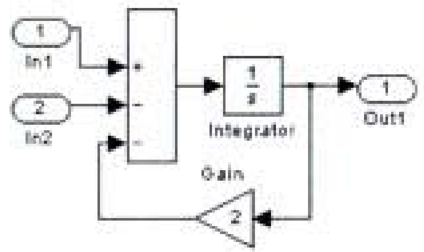
11.1-сурет. Дифференциалдық теңдеуін шешу үрдісі

2. *Edit* мәзірінен (менюінен) *Create Subsystem* командасын (бағыныңқы жүйені құру) таңдалады. Нәтижесінде, *Simulink* ерекшеленген блоктарды *Subsystem* блогымен ауыстырады (11.2-сурет).

Құрылған бағыныңқы жүйенің блок-үрдісін көру үшін *Subsystem* блогын екі рет басу керек. (11.3-сурет). Суреттен көрінетіндей, *Simulink* блок-үрдісіне *In* және *Out* блоктарын қосты.



11.2-сурет. *Subsystem* блогын қолдану мысалы



11.3-сурет. *Subsystem* блогымен құрылған үрді

S-моделін басып шығару және жазу

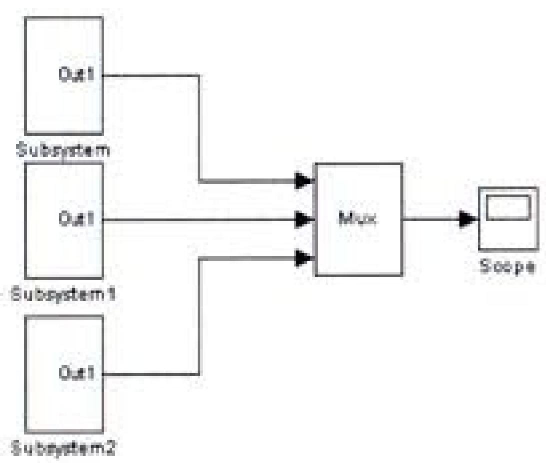
Модельді (блок-үрдіні) дискіге жазу үшін – модель терезесіндегі *File* (файл) менюінен *Save* (сақтау) немесе *Save As* (қалай сақтау ...) командаларын таңдау керек. Сонымен қатар *Simulink* сіз көрсеткен бумаға файлды берілген атымен жазады, оған *.mdl* кеңейтуін қоса тіркейді.

Модельді (блок-үрдіні) қағаз бетіне шығару үшін модель терезесіндегі *File* менюінен *print* командасын қолдану керек.

Блок-үрдіні кез-келген мәтіндік редактор ішіне енгізуге болады. Мысалы, *Word*. Ол үшін алдымен модель терезесіндегі *Edit* менюінен, *Copy Model* командасын шақырып, кейін мәтіндік редактор терезесіне ауысып, *Shift+Ins* батырмаларын басу керек (11.1-11.3-сурет).

11.1 мысал

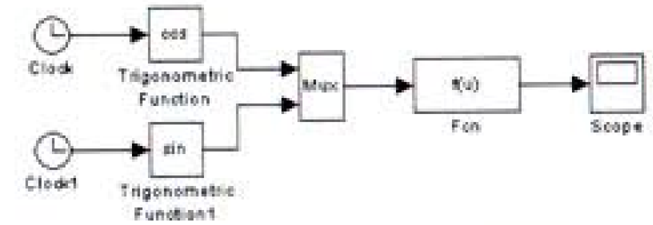
Мына функцияны құру керек болсын: *sin(cos(t)·exp[2.1·(-sin(t))]*, сонымен қатар, берілген функциядан экспонента көрсеткішіндегі коэффициент мәнімен ерекшеленетін тағы екі функцияны құру керек *sin(cos(t)·exp[1.1·(-sin(t))])* және *sin(cos(t)·exp[3.1·(-sin(t))])*. Ол үшін бірінші функцияны бейнелейтін үрдіні *Subsystem* жеке блогына енгізу керек, содан кейін, бұл блокты көшіріп алып, үрдіге



11.4-сурет. *Subsystem* блогын қолдану арқылы тармақтық үрді құру мысалы

аздаған түзетулер жүргізу, яғни, *Fnc* блогының бейнелеулерінде сәйкес

коэффициент мәндерін енгізу керек. Барлық бағыныңқы жүйелер бірдей болады. Оның біреуі 11.5-суретінде көрсетілген.



11.5-сурет. Алдыңғы мысалдағы тармақтық үрдінің төменгі деңгейі

11.2 мысал

Математикалық маятниктің қозғалысы туралы есеп қарастырылады – *m* массалы жүк ұзындығы *l* салмақсыз сырықтың ауырлық өрісінде ілінген. Маятник бір жазықтық бойымен жүреді деп саналады.

Маятникке жүктің жылдамдығына пропорционалды үйкеліс күші $F_{үй} = -Av$ және көлденең бағытталған сыртқы айнымалы күш $F(t) = F\cos\Omega t$ әсер етеді. Жіптің бұрышының тік жолдан ауытқуын мына теңдеу арқылы жазуға болады

$$ml\frac{d^2x}{dt^2} = -mg\sin x - Al\frac{dx}{dt} + F\cos x \cdot \cos\Omega t.$$

Уақыттың *t* орнына келесі қатынас арқылы жаңа айнымалы τ енгізіледі, ол өлшемсіз:

$$t = \sqrt{\frac{l}{g}} \cdot \tau.$$

Онда берілген теңдеу келесі түрге келеді:

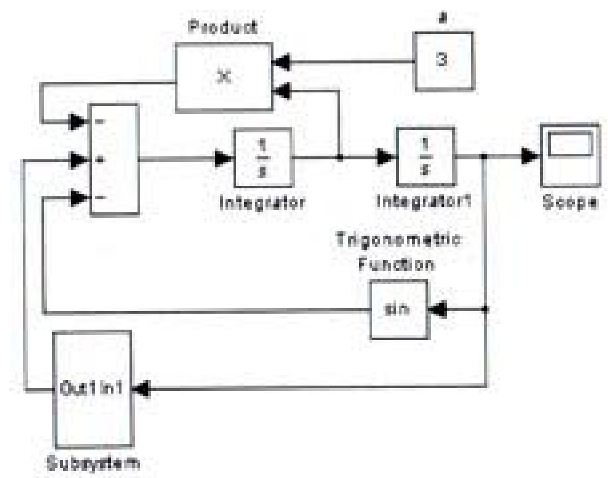
$$\frac{d^2x}{d\tau^2} = -\sin x - a\frac{dx}{d\tau} + f\cos x \cdot \cos\omega t,$$

мұндағы

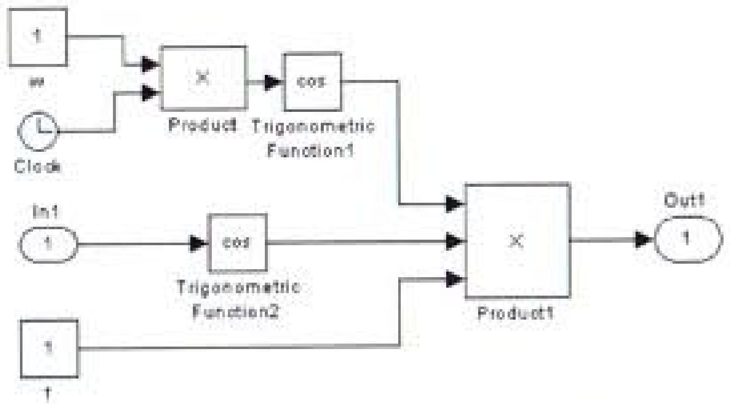
$$a = \frac{A}{m}\sqrt{\frac{l}{g}}, f = \frac{F}{mg}, \omega = \Omega\sqrt{\frac{l}{g}} -$$

өлшемсіз шамалар.

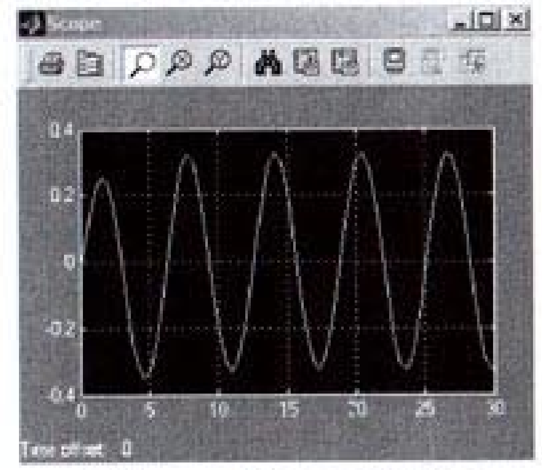
Бұл тапсырмалардың толық зерттеулердің көлемін қысқартуға мүмкіндік береді, өйткені алты параметрлердің орнына үш түрлі параметрлерді қарастыру жеткілікті. *Simulink* пакетін қолдануда математикалық маятниктің іс-әрекеті моделі құралады (11.6-11.8-сурет).



11.6-сурет. Математикалық маятниктің моделінің блок-үрдісі



11.7-сурет. Ішкі модельдің блок-үрдісі



11.8-сурет. Математикалық маятниктің тербелмелі қозғалысы

Тапсырмалар

Төменде көрсетілген тапсырмалар [7] кітаптан алынған. Берілген есептерге *Simulink* бағынынқы жүйесінде орындалатын модель құру керек. Бір жағдайдан екінші жағдайға көшуді *Stateflow* пакеті арқылы немесе *Switch* компонентінің көмегімен іске асыруға болады.

№1 есеп

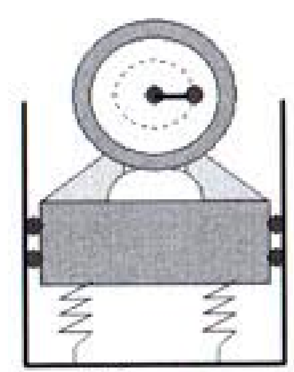
Q айнымалы күші фундаменттегі теңтерілмеген роторлы машинаға беріледі (11.9 сурет). Q келесі гармониялық заңы бойынша өзгереді:

$Q_1 = H_1 sin(w_1 t)$, (11.1)

$Q_2 = H_2 cos(w_2 t)$, (11.2)

мұндағы H_1 және H_2 – ұйткушы күш амплитудасы, w_1 және w_2 – ұйткушы күштің жиілігі.

$H_1 = 2$ *см*, $H_2 = 3$ *см*, $w_1 = 0.3\ c^{-1}$, $w_2 = 0.5\ c^{-1}$ болсын. Гармониялық заңдар бір-бірін 50 *сек* периодымен ауыстырып отырады.



11.9-сурет. Теңтерілмеген роторлы машинаның үрдісі

$k = 0.4\ c^{-1}$ карастырылатын жүйенің өзіндік жиілігі болсын. Егер ол қазіргі уақыт аралығында фундаментке берілетін ұйткушы күштің жиілігіне тең болмаса, яғни (яғни, $k < w_1$ немесе $k < w_2$), онда тербеліс болатын өс бойынша жүретін q координатасының мәні келесі теңдеулермен анықталады:

(11.1) гармониялық заңдар үшін:

$$q''+k^2 \times q = \frac{Q_1}{a}. \qquad (11.3)$$

(11.2) гармониялық заңдар үшін:

$$q''+k^2 \times q = \frac{Q_2}{a}, \qquad (11.4)$$

мұндағы $a = 1\ c^{-2}$ – инерциялық коэффициент.

Алғашқы шарт:

$$q\big|_{t=0} = 0. \qquad (11.5)$$

Егер қазіргі уақыттағы фундаментке берілетін жүйенің өзіндік жиілігі ұйткушы күштің жиілігіне сәйкес келсе ($k = w_1$ немесе $k = w_2$), онда резонансты болдырмау үшін фундаментке әсер ету тоқталады, фундаментке берілетін жиіліктің мәні резонанстың пайда болуының кезінде 10 пайызға төмендейді және 1 секундтан кейін тербеліс жаңа жиілікпен үзіліске дейінгі заң бойынша қайтадан қалпына келеді.

Берілген жүйеге және бір уақытта істейтін және бір-бірінен тәуелсіз екі осындай машинаға модель құрыңыз. Екінші машина да осындай іс-әрекетке ие, онда Q' айнымалы күші сондай заңдармен, бірақ басқа параметрлермен беріледі

$Q_1' = H_3 sin(w_3 t)$, (11.6)

$Q_2' = H_4 cos(w_4 t)$, (11.7)

мұндағы H_3 және H_4 – ұйткушы күш амплитудасы, w_3 және w_4 – ұйткушы күштің жиілігі.

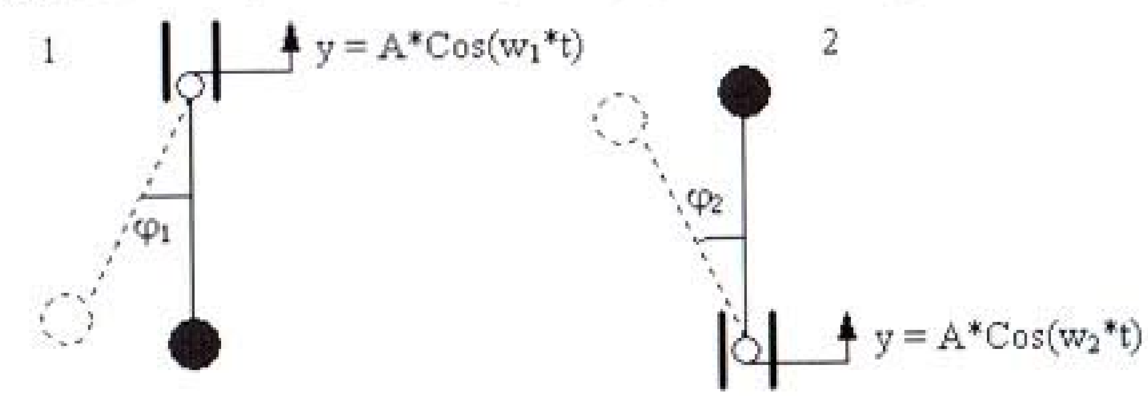
№2 есеп

Екі маятниктен құрылған жүйе берілген: маятник 1 және қайтарылған маятник 2 (11.10-сурет). 1-ші маятниктің ілінген нүктесі 1-заң бойынша тік жол бойының орта бөлігінде гармониялы түрде тербеледі, ал 2-ші маятниктің ілінген нүктесі 2-заң бойынша тік жол бойының орта бөлігінде гармониялы түрде тербеледі:

$y_1 = Acos(w_1 t)$, (11.8)

$y_2 = Acos(w_2 t)$, (11.9)

мұндағы A – тербеліс амплитудасы, w_1 және w_2 – тербеліс жиілігі.



11.10-сурет. Екі маятниктен құрылған жүйе

$A = 2$ *м*, $w_1 = 3\ c^{-1}$, $w_2 = 3\ c^{-1}$ болсын. Осы жағдайда A-ның мәні әрбір 10 сек сайын ±0.5 шамасына кезекпен өзгереді. Әрбір маятниктің ұзындығы $l = 40$ м-ге тең.

φ_1-тік жол бойымен 1-ші маятниктің ауытқу бұрышы, ал φ_2-тік жол бойымен 2-ші маятниктің ауытқу бұрышы болсын.

1-ші маятникте φ_1 келесі теңдеуге сәйкес өзгереді:

$$\varphi_1''+\left(\frac{g}{l}+\frac{A\omega_1^2}{l}\cos(\omega_1 t)\right)\varphi_1=0. \qquad (11.10)$$

Ал 2-ші маятниктің жағдайы басқаша: егер $A\omega>\sqrt{2gl}$, мұндағы g – еркін құлау үдеуі, онда маятник орнықтылық жағдайында болып және $\varphi_2 = 0$ болады.

Ал кері жағдайда φ_2 келесі теңдеуге сәйкес өзгереді:

$$\varphi_2''+\left(-\frac{g}{l}+\frac{A\omega_2^2}{l}\cos(\omega_2 t)\right)\varphi_2=0. \qquad (11.11)$$

Бастапқы кезде екі маятниктің де тепе-теңдік жағдайынан ауытқуы 0.1 градусты құрайды. φ_1 және φ_2 мәндері 0-ден 360 градусқа дейін өзгереді. Берілген жүйенің моделін құрыңыз.

№3 есеп

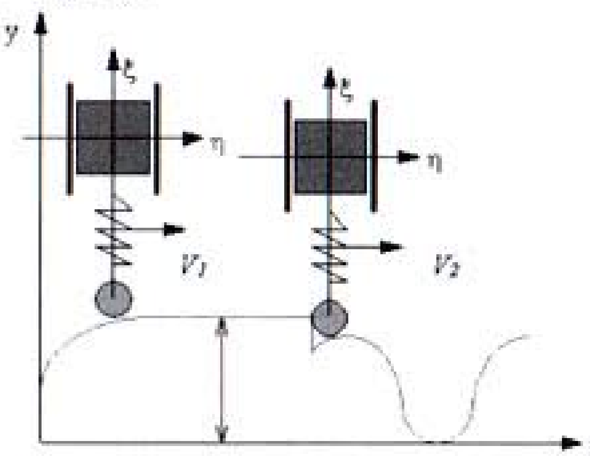
Күрделі кескінді тегіс емес учаске берілген. Онымен екі бірдей жүйе жүреді, олардың әркайсысына массасы $m = 0.7$ *кг* -ға тең подрессорлы жүкті көрсетеді, олар тұрақты көлденең жылдамдықпен жүретін-дөңгелекке бекітілген, мұнда $V_1 = 0.9$ *м/с* (1-ші жүйе үшін) және $V_2 = 1$*м/с* (2-ші жүйе үшін) (11.11-сурет).

Учаскенің кескіні мынадай: V_1 жылдамдықпен жүйе белгілі бір уақыт бөлігінде жүреді $Time_1 = 20$ *с* (11.12) теңдеуге сәйкес бөлігі, ары қарай $Time_2 = 30$ *с* периодымен V_2 жылдамдығы үшін ($Time_2' = 27$ *с*) учаскенің кескіні (11.13) және (11.14) теңдеумен көрсетіледі.

$y = h\,(1 - e^{-rx})$ (11.12)

$y = A_1\cos x$ (11.13)

$y = A_2\sin x$ (11.14)



11.11-сурет. Екі подрессорлы жүктен құрылған жүйе

мұндағы h – тегіс еместіктің биіктігі ұмтылатын шек, g – кескіннің қисықтығын көрсететін параметр, A_1 және A_2 – тербеліс амплитудалары.

$h = 1$ *м*, $g = 1$, $A_1 = 0.8$ *м*, $A_2 = 0.7$ *м* болсын.

1-ші жүйе алғашқы уақытта жолдың басында орналасқан, ал 2-шісі косинусоидалық учаскенің басында орналасқан. Алғашқы уақытта 1-ші жүйе $V = V_1$, жылдамдықпен жүре бастайды да, ал 2-ші сол орнында қалады. 1-ші жүйе экспоненциальді бөліктің соңына жеткенде $V = V_2$ ($V_2 > V_1$), жылдамдықпен 2-ші жүйе жүре бастайды да, содан кейін 2 жүйе бірге жүреді.

1-ші жүктің x тік жол бойымен тербелісін көрсететін $x = Vt$ (мұндағы $V = V_1$) дифференциалдық теңдеуі келесідей (11.15-11.17)

$$\xi''=-\frac{c}{m}\xi+h\gamma^2V^2e^{-\gamma Vt}, \qquad (11.15)$$

теңдеуі (11.13) үшін:

$$\xi''=-\frac{c}{m}\xi+A_1V^2\cos(Vt), \qquad (11.16)$$

теңдеуі (11.14) үшін:

$$\xi''=-\frac{c}{m}\xi+A_2V^2\sin(Vt), \qquad (11.17)$$

мұндағы $c = 0.5$ *кг/с²* – серпімді аспаның қатаңдығы.

2-ші жүктің қозғалысы $V = V_2$ шарты бойынша (11.16) және (11.17) теңдеулерімен сипатталады.

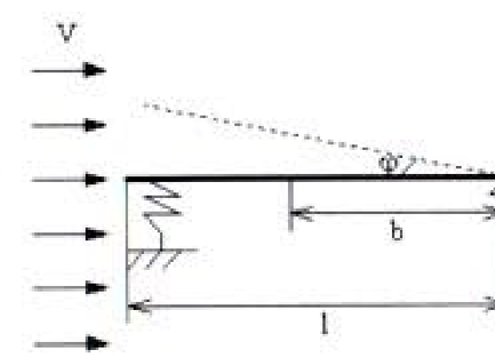
Егер h шегі $h_{min} = 0.001$ *м* -ден кіші болса, немесе A_1 және A_2, $A_{min} = 0.01$ *м* амплитудасынан кіші болса, онда учаскенің кескіні түзу сызықтық болады да және жүктің тербелісі келесі теңдеулермен сипатталады:

$$\xi''=-\frac{c}{m}\xi. \qquad (11.18)$$

Берілген жүйенің моделін құрыңыз.

№4 есеп

Ұзындығы $l = 5$ *м* қатты жайпақ пластинка газ ағымында орналасқан, оның жылдамдығы $V = 1$ *м/с* тепе-теңдіктің ұйтқусыз (мызғымас) жүйесіндегі жайпақ пластинканың ортасына бағытталған. Бұл жағдайда аэродинамикалық күштер нөлге тең, пластинка ауырлық күшінің және тірек реакциясының әсерінен тепе-теңдікте тұр (11.12-сурет).



11.12-сурет. Аэродинамикалық жүйе

φ пластинканың ауытқу бұрышынан тәуелді пластинканың ауытқу жағдайындағы аэродинамикалық қысым пайда болады. Алғашқы уақытта пластинка тепе-тең жағдайдан *0.01* градус бұрышына ауытқиды.

$I = 1$ *кг·м²* – шарнирдің өсіне сәйкес пластинкадағы инерцияның моменті болсын, онда қозғалыстың дифференциалдық теңдеуі:

$$I\varphi''+\left(c_0 l-k_y\frac{\rho V^2}{2}b\right)l\varphi=0, \qquad (11.19)$$

мұндағы c_0 – серіппенің қатаңдық коэффициенті, r – газдың тығыздығы, b – шарнир өсінен арақашықтығы, ол біркелкі әсер етуші аэродинамикалық қысымның пластинкаға әсер нүктесін анықтайды, K_y – тұрақты аэродинамикалық коэффициент.

$c_0 = 0.5$ *кг/с*^{2}, $K_y = 0.5$ *м·с*^{2}, $r = 2$ *кг/м*^{3}, $b = 1$ *м* болсын.

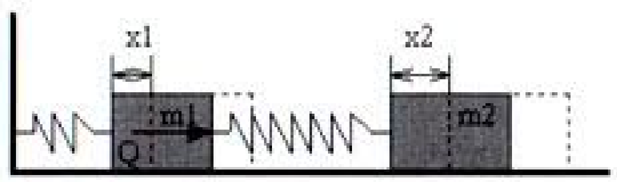
(11.19) теңдеуі $c_0 l - k_y \dfrac{\rho V^2}{2} b > 0$ шартында ғана орындалады. Кері жағдайда аэродинамикалық күштің әсерінен пластинка тепе-теңдік жағдайына қайта келеді.

Әрбір *5 с* сайын берілген газдың жылдамдығы *50%* өседі, не алғашқы қалпына қайта оралады. Сонымен қатар әрбір *10 с* сайын берілген газдың қысымы *50%* өседі, не алғашқы қалпына қайта оралады.

Берілген жүйенің моделін құрыңыз және бір-бірімен байланысы жоқ, газ ағымындағы пластинка типті жүйеден тұратын модель жүйесін құрыңыз. Газ ағымындағы пластинка 2-ші жүйесі біріншімен бірдей, тек мұнда пластинканың ұзындығы $l_1 > l$, газ ағымы жылдамдығы $V_1 > V$ және тығыздығы $r_1 < r$ бөлек.

№5 есеп

Массасы m_1 және m_2 денелерден құрылған, қатаңдығы c_1 және c_2-ге тең серіппемен қосылған жүйе берілген. $m_1 = 1$ *кг*, $m_2 = 1$ *кг*, $c_1 = 1$ *кг/с*^{2}, $c_2 = 1$ *кг/с*^{2} болсын (11.13 сурет).



11.13-сурет. Серіппемен қосылған екі денелерден құрылған жүйе

Жүйенің сол жақ жүгіне (11.20) немесе (11.21) заңмен берілген интервалы *20 с-ке* тең болатын Q гармониялық ұйтқушы күш әсер етеді.

$$Q = H_1 \sin(wt), \tag{11.20}$$

$$Q = H_2 \cos(wt), \tag{11.21}$$

мұндағы H_1 және H_2 – тербеліс амплитудалары, w – тербеліс жиілігі.

$H_1 = 1$ *м*, $H_2 = 1.5$ *м*, $w = 2$ *с*$^{-1}$ болсын.

Тербеліс жиілігі ауытку заңынан тәуелсіз әрбір *25 с* сайын *50%* төмендейді немесе алғашқы қалпына қайта келеді. x_1 және x_2 – тепе-теңдік жағдайынан жүктердің көлденең ауытқуы. Онда қозғалыстың теңдеуі мынадай:

$$m_1 x_1'' + (c_1 + c_2)x_1 - c_2 x_2 = Q, \tag{11.22}$$

$$m_2 x_2'' - c_2 x_1 + c_2 x_2 = 0. \tag{11.23}$$

Берілген жүйенің моделін және бір-бірімен байланысы жоқ "екі жүкті" екі жүйеден құрылған жүйе моделін құрыңыз. "Екі жүкті"

екінші жүйе біріншімен бірдей, бірақ ондағы сол жақтағы жүк $m_2 < m'_1 < m_1$ басқа массаға ие, ал оң жақтағы пружина басқа қатаңдыққа $c'_2 > c_2$ ие.

№6 есеп

Массасы жоқ тік серпімді қатты сырық берілген. Ұзындығы $l = 5$ *м* тұрақты қиылысу қатаңдығы $EJ = 1$ *кг·м*^{3}*/с*^{2}. Сырықтың соңына массасы $m = 1$ *кг* жүгі бекітілген. Сырықтың жоғарғы тірегінде қозғалмалы топса, ал төменгі тірегінде шарикті ішпекті қимылдайтын төлке бар (11.14-сурет).

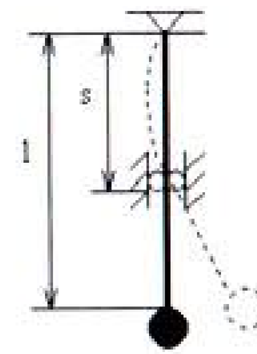
Тіректердің арасындағы қашықтық $s < l$ тұрақсыз және екі гармониялық заңдылықтың біреуімен әрбір *30 с* сайын төлкенің қимылы арқылы өзгеріп отырады:

$$s = H_1 \cos(w_1 t), \tag{11.24}$$

$$s = H_2 \sin(w_1 t), \tag{11.25}$$

мұндағы H_1 және H_2 – тербеліс амплитудалары, w_1 – тербеліс жиілігі.

$H_1 = 1.5$ *м*, $H_2 = 2$ *м*, $w = 1$ *с*$^{-1}$ болсын.



11.14-сурет. №6 есепке арналған механикалық жүйе

Тербеліс жиілігі тербеліс заңына сәйкессіз әрбір *20 с* сайын *40%*-ке төмендейді, не алдыңғы қалпына қайта келеді.

Тепе-теңдік жағдайына сырықтың өсінің түзу сызықтық түрі және жүктің тік жағдайы сәйкес келеді. Ұйтқу жағдайында жүк бір жаққа ауытқиды, арқалықтың өсі иіледі және келесі қозғалыстар дифференциалдық теңдеулермен сипатталады:

$$x'' + \frac{3EJ}{ml[l-s]^2} x = 0. \tag{11.26}$$

Алғашқы кезеңде жүктің ауытқуы $x_0 = 0.1$ *м* арақашықтықта болады. Егер x шекті мәнін $x_{max} = 3$ *м* жоғарылатса, онда сырық бұзылады. Берілген модельдің жүйесін құрыңыз және бір-біріне тәуелсіз, бір уақытта жұмыс істейтін жүйеге 2 модель құру керек. 2-ші жүйе дәл сондай әрекетте ие, онда айнымалы күш s' сол заңдармен, бірақ басқа параметрлермен жүреді

$$s' = H_3 \cos(w_2 t), \tag{11.27}$$

$$s' = H_4 \sin(w_2 t), \tag{11.28}$$

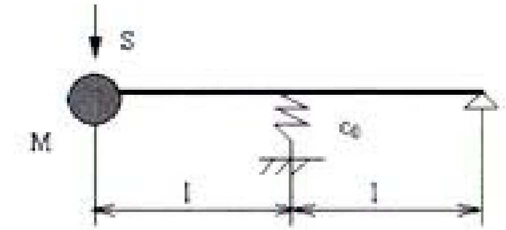
мұндағы H_3 және H_4 – тербеліс амплитудалары, w_2 – тербеліс жиілігі.

№7 есеп

Массасы $M = 1$ *кг* салмақсыз, арқалықпен қатаң серпімді бекітілген жүктен тұратын жүйе берілген.

$2l$ – арқалықтың ұзындығы, c_0 – серіппенің қатаңдық коэффициенті болсын. Балканың бір соңы қимылдамайтын тіректе орналасқан топсаға бекітілген.

$L = 1$ *м*, $c_0 = 1$ *кг/с*^{2} болсын (11.15-сурет). Уақыттың бастапқы мезетінде жүкке лездік соққы импульсімен $S = 2H{\cdot}c$ бір ретті тік соққы беріледі. Жүктің жылдамдығы лездік өсімшеге ие болады, одан келесі алғашқы шарттар туындайды:



11.15-сурет. Арқалықпен қатаң серпімді бекітілген жүктен тұратын жүйе

$$\varphi\big|_{t=0} = 0, \tag{11.29}$$

$$\varphi'\big|_{t=0} = \frac{S}{2Ml}, \tag{11.30}$$

мұндағы φ – жүйенің тепе-теңдік жағдайынан ауытқу бұрышы.

Жүйенің қозғалысы келесі теңдеумен көрсетілген:

$$4M\varphi'' + c_0\varphi = 0. \tag{11.31}$$

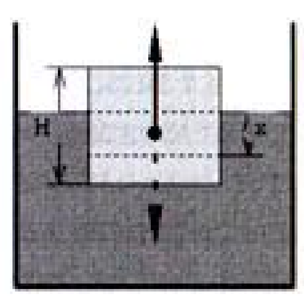
Тербеліс басталғаннан кейін *25 с*-тан соң жүктің массасы *50%*-ке кемиді. Кейін жаңа масса жүгімен жүйенің қозғалысы жалғасады.

Егер φ бұрышы шектеулі мәннен көбірек болса, ол жүйенің бұзылуына әкеледі $\varphi_{\max} = \dfrac{S}{\sqrt{c_0 Ml}}$.

«Арқалық-жүк» жүйесінің және бір-біріне тәуелсіз екі жүйеден құралған «арқалық-жүк» жүйесінің моделін құрыңыз. Екінші арқалық біріншімен бірдей, бірақ оған соққы бірінші жүйедегі «арқалық-жүкке» соққы болғаннан кейін 10 *сек* өткеннен кейін болады.

№8 есеп

Суда ауданы $S = 1$ *м*^{2}, биіктігі $h = 0.5$ *м* параллелепипедке ұқсайтын тығынның бір бөлігі жүзіп жүр. Тығынды суға $x_0 = 5$ *см* тереңдікке батырып, оны жібереді (11.16-сурет). Нәтижесінде тығын тербеліске түседі. Судың кедергісі ескерілмейді. Тығынды суға батырғандағы тереңдіктің өзгеруі x теңдеуімен көрсетіледі:



11.16-сурет. «Су-тығын» тербеліс жүйе

$$x'' + \frac{\rho_{\hat{a}} g}{\rho_{\ddot{i}} H} x = 0 \tag{11.32}$$

алғашқы шарттармен:

$$x\big|_{t=0} = x_0, \tag{11.33}$$

$$x'\big|_{t=0} = 0, \tag{11.34}$$

мұндағы ρ_c = *1000 кг/м*^{3} – судың тығыздығы, ρ_m = *200 кг/м*^{3} – тығынның тығыздығы, g – еркін түсу үдеуі.

Екінші нақ осындай "су-тығын" жүйесі болсын, ондағы тығынды алғашқы кезде суға батырмай жібере салып, оған $v_0 = 1$ *м/с* жылдамдықты береді. Бұл жүйеде судың кедергі күші бар деп ескерсек және ол тығынның жылдамдығына пропорционал $F_c = -r{\cdot}v$, мұндағы $r = 1$ *кг/с* – пропорционалдық коэффициенті. Мұндағы тербелістер келесі теңдеумен көрсетіледі:

$$x'' + \frac{r}{m}x' + \frac{\rho_{\hat{a}} g}{\rho_{\ddot{i}} H} x = 0, \tag{11.35}$$

алғашқы шарттармен:

$$x\big|_{t=0} = 0, \tag{11.36}$$

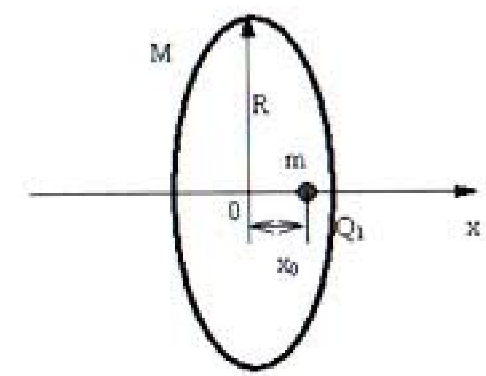
$$x'\big|_{t=0} = v_0, \tag{11.37}$$

мұндағы m – тығынның массасы.

Бір бірімен байланысы жоқ "су-тығын" екі жүйесінен құралған жүйенің моделін құрыңыз. Бірінші жүйеде әрбір *20 с* сайын су сынапқа "айналудың" (және керісінше). Дәл сол жүйеде әрбір *25 с* сайын су спиртке "айналуына" байланысты модель құру керек. Сынаптың тығыздығы – ρ_p = *1360 кг/м*^{3}, спирттің тығыздығы – ρ_c = *790 кг/м*^{3}.

№9 есеп

Массасы M =*1020 кг* және радиусы R = *1 км* жіңішке дөңгелектің тартылыс өрісінде массасы m = *1 кг* болатын материалдық нүкте орналасқан. Уақыттың алғашқы мезетінде нүкте дөңгелектің өсінде, оның жазықтығынан $x_0 < R$ қашықтықтағы Q_1 нүктесіне орналасады және тербеле бастайды (11.17-сурет).



11.17-сурет. "Материалдық нүкте-дөңгелек" тербеліс жүйе

Егер $x_0 < 0.1R$ болса, онда x тербелісі келесі теңдеумен беріледі:

$$mx'' = -\frac{GMm}{R^3}x. \tag{11.38}$$

Егер $x_0 < i < 0.1R$ немесе тербеліс нәтижесінде $x < i < 0.1R$ орындалса, онда онда x тербелісі келесі теңдеумен беріледі:

$$mx'' = -\frac{GMm}{(R^2 + x^2)^{3/2}}x \tag{11.39}$$

мұндағы G – гравитациялық тұрақты.

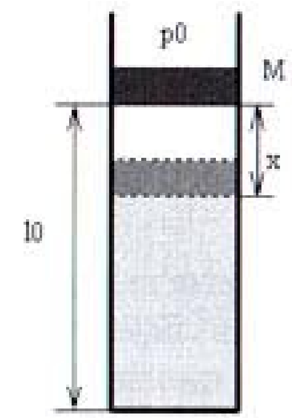
$x_0 = 1$ *м* болсын.

Әрбір *15 с* сайын дөңгелектің радиусы кезекпен лезде кеңейеді және *10* есе сығылады. Егер нүкте дөңгелектен оның алғашқы радиусынан *10* есе артық белгілі-бір қашықтыққа ауытқыса, онда жүйе бұзылады.

Осы жүйенің моделін және бір-біріне байланысы жоқ "материалдық нүкте-дөңгелек" жүйесінің моделін құрыңыз. Екінші "материалдық нүкте-дөңгелек" жүйесі біріншімен бірдей, бірақ онда дөңгелектің радиусы әрбір *20 с* сайын бес рет өзгереді.

№10 есеп

Төменгі жағы жабық ұзын тік цилиндрлі түтікшеде массасы $M = 10$ *кг* піспек үйкеліссіз жүреді, оның массасы трубканың ішінде қамалған идеалды газдың массасымен салыстырғанда үлкен.



11.18-сурет. "Түтікше -піспекпен" тербеліс жүйе

Тепе-теңдік жағдайында піспек мен түтікшенің түбіндегі қашықтық $l_0 = 1000$ *м* тең (11.18-сурет). Түтікшенің көлденең қимасының ауданы $S = 1$ *м*^{2}, піспекке дұрыс атмосфералық қысым p_0 әсер етеді. Алғашқы кезде піспек тепе-теңдік жағдайынан $x << l_0$ қашықтыққа ауытқиды. Осының нәтижесінде піспек тербеліске түседі де, оның қозғалысы келесі теңдеулермен беріледі:

Егер $p_0 < 0$:

$$Mx''+\frac{Mg+p_0S}{l_0}x=0. \qquad (11.40)$$

Егер $p_0 = 0$:

$$x''+\frac{g}{l_0}x=0, \qquad (11.41)$$

мұндағы g – еркін түсу үдеуі.

$x = 1$ *м* болсын.

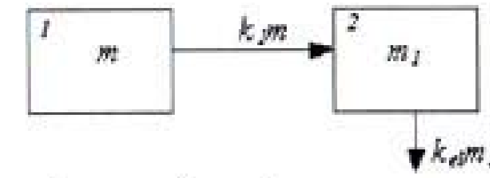
Атмосфералық қысым әрбір *10 с* сайын өзінің мәнін қалыпты жағдайдан *80%*-ке лезде кемітіп өзгертеді және керісінше.

Осы жүйенің моделін және бір-біріне байланысы жоқ екі "түтікше – піспекпен" жүйесінен тұратын жүйенің моделін құрыңыз. Екінші "түтікше – піспекпен" жүйесі біріншімен бірдей, бірақ онда піспектің массасы $M' > M$, түтікшенің көлденең қимасының ауданы $S' > S$ өзгереді.

№11 есеп

Сорғымасы бар бір камералы фармакокинетикалық модель берілген (11.19-сурет).

Камера уақыттың өтуімен өзгермейтін, кеңістікте шектеулі сұйықтың көлемі болып табылады. Дәрілік препараттың арнайы көлемі берілген, ол камераға өзінің массасына пропорционалды мына теңдеуге сәйкес сорылады:



1 – ол дәрілерді салатын орын, *2* – камера.

11.19-сурет. Бір камералы фармакокинетикалық модельдің үрдісі

$$\frac{dm}{dt}=-k_1m, \qquad (11.42)$$

мұндағы m – *1* орынға енгізілетін дәрілік препараттың массасы, k_1 – камерадағы препараттардың түсу жылдамдығы (сору жылдамдығының тұрақтысы).

$k_1 = 0.3\ c^{-1}$ болсын.

Уақыттың алғашқы мезетінде *1*-дегі дәрілік препараттың массасы $M = 30$ *мг*, ол камерада алғашқы кезде препараттар жоқ. Онда камерадағы дәрілік препараттың массасы келесі теңдеумен өзгереді:

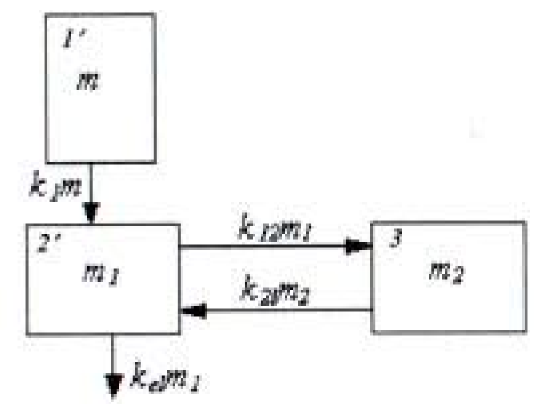
$$\frac{dm_1}{dt}=k_1m-k_{el}m_1, \qquad (1.43)$$

мұндағы m_1 – камерадағы дәрілік препараттың массасы, k_{el} – камерадан препараттардың шығу жылдамдығы (элиминация тұрақтысы).

$k_{el} = 0.5\ c^{-1}$ болсын.

Дәрілік препарат салмағы енгізу жерінде шекті мәннен кем болған жағдайда $\varepsilon = 0.001$ *мг*, уақыт аралығы $Time = 10$ *с* белгіленіп қалады. Бұл уақыт өткеннен кейін, енгізу жерінде алдыңғы дозаның қалдықтары жойылып, жаңа доза енгізіледі $m = m_0$.

Бұдан күрделірек жүйе бар, ол сорғымасы бар екі камералық фармакокинетикалық модель деп аталады (11.20-сурет). Оның ішінде дәрілік препарат әуелде айтылғандай үйлесімді кіріс жеріне *1′* енгізіліп, *2′* камерасынан шығарылады. Бірақ та *3*-і камера бар, ол *2′* камерасына қосылған. *2′* және *3* камералар арасында дәрілік препарат айналып жүре алады. Бұл модельде дәрілік препарат салмағы енгізу жерінде (11.42) теңдеумен және бастапқы шартармен сипатталады *2′* және *3* камерадағы салмақ (11.44) және (11.45) теңдеулерімен бейнеленеді:



11.20-сурет. Екі камералы фармакокинетикалық модельдің үрдісі

$$\frac{dm_1}{dt}=k_1m-(k_{el}+k_{12})m_1+k_{21}m_2, \qquad (11.44)$$

$$\frac{dm_2}{dt} = k_{12}m_1 - k_{21}m_2, \tag{11.45}$$

мұндағы m_2 – 3 камерадағы дәрілік препараттың массасы, k_{12} – $2'$ камерадан 3 камераға препараттардың түсу жылдамдығының тұрактысы, k_{21} – 3 камерадан $2'$ камераға препараттардың шығу жылдамдығының тұрактысы.

$k_{12} = 0.4\ c^{-1}$, $k_{21} = 0.6\ c^{-1}$ болсын.

Бірінші жүйедегідей, бастапкы уақыт мезгілінде $2'$ және 3 камераларда дәрілік препарат жок.

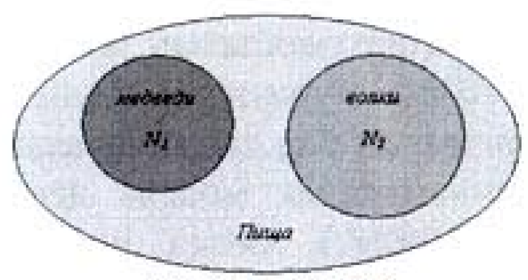
Бірінші жүйедегідей, дәрілік препарат салмағы енгізу жерінде шекті мәннен ε' аз болған жағдайда, уақыт аралығы $Time_1$ белгіленіп қалады. Бұл уақыт өткеннен кейін, енгізу жерінде алдыңғы дозаның қалдықтары жойылып, жаңа доза енгізіледі $m = m_0$.

Бір-бірімен байланыспаған дәріні енгізу жерлері әртүрлі екі фармококинетикалық моделі бар жүйе моделін құрыңыз. Сонымен қатар, препаратты енгізу жері ортақ екі фармакокинетикалық модельдері бар жүйе моделін құрыңыз.

Дәрілік препараттың сорылу жылдамдығы k_1 екі фармакокинетикалық модельдерінде де бірдей, уақыт аралығы $Time_1$ және шекті мән ε' препаратты енгізу жері үшін сорғымасы бар 1 камералы фармакокинетикалық модельдің мәндерімен сәйкес келеді.

№12 есеп

Бір тағамды қолданатын екі биологиялық популяция берілсін. Аюлар популяциясы (N_1 санымен) және қасқырлар популяциясы (N_2 санымен) болсын (11.21-сурет). Екі түрді де қанағаттандыратын азық көлемі үшін популяциялардың өсуінің тұракты оң коэффициенттері бар болсын. Олар аюлар үшін $\varepsilon_1 = 0.7$ ай$^{-1}$ және қасқырлар үшін $\varepsilon_2 = 0.9$ ай$^{-1}$-тең. Әр популяцияның азыққа деген сұранысына сәйкес "қомағайлық коэффициенттері" $\gamma_1 = 0.7\ кг^{-1}$ және $\gamma_2 = 0.5\ кг^{-1}$ белгіленген.



медведи – аюлар
волки – қасқырлар
пища – азық

11.21-сурет. Бір тағамды қолданатын екі биологиялық популяциянің үрдісі

Уақыт бірлігінде екі популяцияда қолданылатын азық көлемі $F(N_1, N_2)$ болсын. Ол (11.46) және (11.47) теңдеулерімен беріледі, мұнда (11.46) теңдеу екі популяция да белсенді болған жағдайына, ал (11.47) теңдеу аюлар ұйқыға кеткен жағдайына сәйкес.

(11.46) және (11.47) режимдер арасындағы ауысу периодты түрде жүреді. (11.46) режимнен (11.47) режимге ауысу $Time_1 = 9$ ай уақыт

аралығы арқылы жүреді, ал (11.47) режимнен (11.46) режимге ауысу $Time_2 = 3$ ай уақыт аралығы арқылы жүреді:

$$F(N_1, N_2) = \lambda_1 N_1 + \lambda_2 N_2, \tag{11.46}$$

$$F(N_1, N_2) = \lambda_2 N_2, \tag{11.47}$$

мұндағы λ_1 және λ_2 – қандай да бір оң коэффициенттер.

$\lambda_1 = 0.01\ кг/(ай \cdot тал)$, $\lambda_2 = 0.02\ кг/(ай \cdot тал)$ болсын.

Бастапқы кезде популяциялар саны $N_1' = 10\ тал$ және $N_2' = 20\ тал$.

Онда популяцияның дамуы келесі теңдеулермен беріледі:

аюлар:

$$\frac{dN_1}{dt} = [\varepsilon_1 - \gamma_1 F(N_1, N_2)]N_1, \tag{11.48}$$

қасқырлар:

$$\frac{dN_2}{dt} = [\varepsilon_2 - \gamma_2 F(N_1, N_2)]N_2. \tag{11.49}$$

Бірінші немесе екінші популяцияның саны 1-ден кем болған кезде (соңғы жеке түр өлгеннен кейін), $Time_3 = 3$ ай (аюлар болса), немесе $Time_3' = 5$ ай (қасқырлар болса) уақыт аралығы белгіленеді. Оның орнына жаңа бастапқы санымен жаңа популяция (N_1' – аюлар, N_2' – қасқырлар) орналасады.

Берілген жүйенің моделін құрыңыз, сонымен қатар өзара байланыспаған "2 популяция" типінен тұратын 2 жүйеден құралған жүйе моделін құрыңыз. "2 популяция" екінші жүйесі біріншісіне ұқсас, бірақ оның ішіндегі аюлар мен қасқырлардың орнына 2 бәсекелес түр ретінде бұғылар мен маралдар алынады. "Қомағайлық коэффициенттері" γ_1' және γ_2', өсу коэффициенттерінің мәндері сәйкесінше ε_1' және ε_2', басқа да коэффициенттер мәндері λ_1 және λ_2-ге тең.

Он екінші сабақ. Сигналдарды спектральды талдау

Сабақтың жоспары

1. Спектральды талдаудың кейбір мәселелері.
2. Фурьенің тура және кері түрлендіруі.
3. Фурьенің дискретті тура және кері түрлендіруі.
4. *Matlab*-тың *fft* және *ifft* процедуралары.
5. Спектральды талдаудың мысалдары.

Спектральды талдаудың кейбір мәселелері

Сигналдарды спектральды талдаудың негізгі мәселесі – олардың гармониялық спектрін анықтау, яғни сигналдың гармониялық құрушыларының жиілігін анықтау (жиілік спектрін анықтау), ол гармониялық құраушылардың амплитудаларын (амплитудалық спектр) және бастапқы фазасын анықтау (фазалық спектр).

Спектрлық талдаудың негізін периоды $T = \dfrac{2\pi}{\omega} = \dfrac{1}{f}$ болатын кез келген периодты үрдісті шексіз, бірақ саны есептеулі жеке гармониялық құраушыларға жіктеуге болатындығы туралы Фурье теориясы құрайды (мұндағы ω – периодты үрдістің шеңберлік жиілігі, ал f – оның герцпен алынған жиілігі).

Алдымен периоды T болатын кез келген периодты үрдісті комплексті Фурье қатары түрінде көрсетуге болады:

$$x(t) = \sum_{m=-\infty}^{+\infty} X^{*}(m) \cdot e^{j(2\pi mf)\cdot t} = \sum_{m=-\infty}^{+\infty} X^{*}(m) \cdot e^{j(m\omega)\cdot t}. \qquad (12.1)$$

Гармониялық құраушылардың *комплексті амплитудалары* деп аталатын $X^{*}(m)$ комплексті сандары келесі формулалар арқылы есептеледі:

$$X^{*}(m) = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) \cdot e^{-j(2\pi mf)t} dt = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) \cdot e^{-j(m\omega)t} dt. \qquad (12.2)$$

Сонымен, периодты тербелістің жиілік спектрі негізгі (бастапқы) f жиілікке еселік болатын жиіліктерден тұрады, яғни:

$$fm = mf \ (m=0,\ 1,\ 2, \ldots). \qquad (12.3)$$

$X^{*}(m)$ комплексті амплитудаларының нақты және жорамал бөліктері периодты тербелістің сәйкесінше *нақты* және *жорамал спектрларын* құрайды. Егер комплексті амплитуданы (12.2) экспоненциалдық түрге келтірсек:

$$X^{*}(m) = \frac{a_m}{2} e^{j\cdot\varphi_m}, \qquad (12.4)$$

онда a_m өлшемі жиілігі $f_m = mf$ болатын гармониялық құраушының амплитудасын, ал φ_m – ол гармониканың косинусоида түріндегі бастапқы фазасын көрсетеді, яғни бастапқы үрдісті келесі түрде жазуға болады:

$$x(t) = a_0 + \sum_{m=1}^{+\infty} a_m \cdot \cos(2\pi mft + \varphi_m), \qquad (12.5)$$

оны *Фурье қатары* деп атайды.

(12.5) және (12.1) түріндегі жіктеулер (12.2) комплексті амплитудалар жиынтығын периодты үрдістің жиілік аймағындағы кескіні ретінде қарастыруға мүмкіндік береді.

Жоғарыда келтірілгендерді кез келген үрдіске оның ішінде периодты емес үрдістерге де қолдануға болады, бұл өз ретінде келесі өрнекпен берілетін *Фурье-кескіндері* түсінігін кіргізуді қажет етеді:

$$X(f) = \int_{-\infty}^{+\infty} x(t) \cdot e^{-j(2\pi f)t} dt. \qquad (12.6)$$

Бұл жағдайда Фурье-кескінді бастапқы үрдіске $x(t)$ кері түрлендіру төмендегі интеграл арқылы анықталады:

$$x(t) = \int_{-\infty}^{+\infty} X(f) \cdot e^{j(2\pi f)t} df. \qquad (12.7)$$

Matlab-тың *fft* және *ifft* процедуралары есептеулерді сәйкесінше келесі формулалар арқылы орындайды:

$$y(k) = \sum_{m=1}^{n} x(m) \cdot e^{-j\cdot 2\pi\cdot(m-1)\cdot(k-1)/n}, \qquad (12.8)$$

жорамал бірлікті көрсетеді; n – берілген x векторының элементтерінің саны (бұл өлшем y шығыс векторына да қатысты). *fft* процедурасы уақыт бойынша дискретті болып келетін берілген $x(t)$ үрдісінің дискретті уақытқа бөлінгендегі Фурье-кескінін табады

$$y(k) = \frac{X(k)}{Ts}, \qquad (12.10)$$

мұндағы

$$X(k) = Ts \cdot \sum_{m=1}^{n} x(m) \cdot e^{-j\cdot(2\pi/n)\cdot(k-1)\cdot(m-1)}, \qquad (12.11)$$

мұндағы $Ts = \Delta t$.

(12.11) қатынасы интегралдау үрдісін қосу үрдісімен (12.7) ауыстыру арқылы алынады. Мұндай көшірілім тек аз уақытқа созылатын үрдістерге ғана қатысты, ал үрдістің өзі бұл уақыт аралығында шектелген нүктелердегі мәндері арқылы берілуге тиіс.

Көптеген стационарлық тербеліс үрдістерінде жиіліктік, амплитудалық және фазалық спектрлер нақты орындаудағы T-ның ұзақтығынан және таңдалған дискретті уақыттан *Ts* тәуелсіз, сондықтан стационарлы үрдістердің спектрлі талдауы үшін *fft* процедурасы қолданылады. Алынған нәтиже соңынан өлшем нүктелерінің санына бөлінеді.

fft (*Fast Fourier Transformation*) және *ifft* (*Invers Fast Fourier Transformation*) функциялары берілген векторды түрлендіруге, сәйкесінше Фурьенің дискретті тура және кері түрлендіруін жүзеге асырады.

Ол функцияларды шақыру:

y = fft(x,n); x = ifft(y, n)

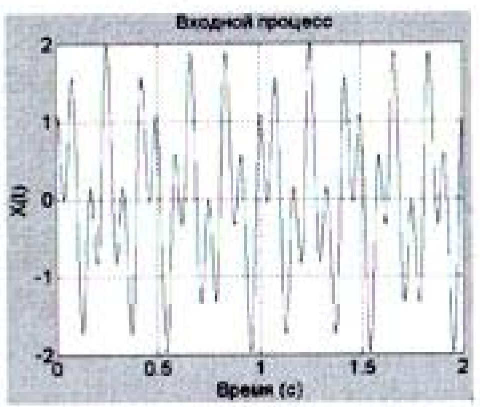
(12.8), (12.9) формулаларының көмегімен бірінші жағдайда *y* және екінші жағдайда *x* векторларының құрылуына мүмкіндік береді.

12.1 мысалы. Сигналдың Фурье-кескіні

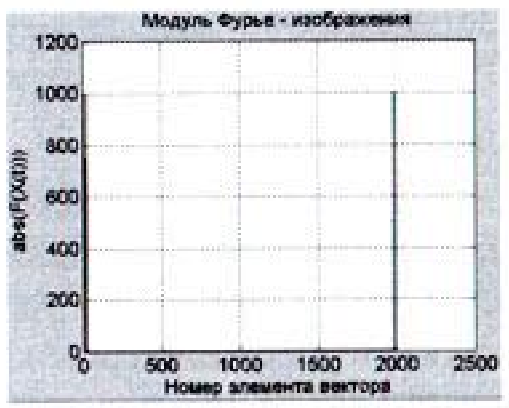
Элементтері жиіліктері 5 болатын синусоида мен 12 Гц болатын косинусоиданың қосындысы болатын функцияның мәндерінен тұратын вектор түріндегі кіріс сигналын құру керек. Ол сигналдың Фурье-кескінін тауып, кіріс үрдісінің графикалық кескінін және Фурье-кескіннің модулін шығару керек:

```
>> t = 0:0.001:2;
>> x = sin(2*pi*5*t) + cos(2*pi*12*t);
>> plot(t, x); grid
>> set(gca, 'FontName', 'Arial Cyr','FontSize',16);
>> title('Кіріс үрдіс');
>> xlabel('Уақыт (с)');
>> ylabel('X(t)');
>> y = fft(x);
>> a = abs(y);
>> plot(a); grid
>> set(gca, 'FontName', 'Arial Cyr','FontSize',16),
>> title(' Фурье-кескіннің модулі');
>> xlabel('вектор элементінің нөмірі');
>> ylabel('abs(F(X(t))')
```

Нәтижелер сәйкесінше 12.1 және 12.2 суреттерінде келтірілген.



2.1-сурет. Кіріс үрдісінің Фурье-кескіні

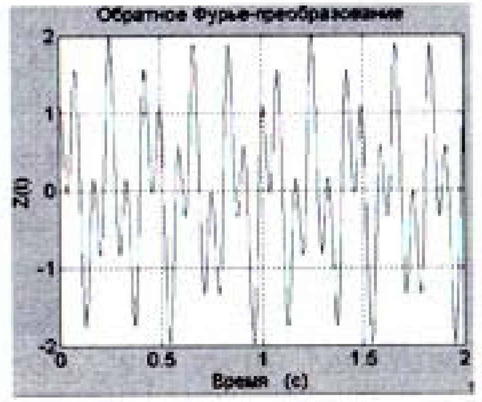


12.2-сурет. Кескін модулінің Фурье-кескіні

Кері түрлендіру *ifft* функциясының көмегімен жүзеге асырылады:

```
>> z = ifft(y);
>> plot(t, z); grid
>> set(gca, 'FontName', 'Arial Cyr',
   'FontSize', 16);
>> title(' Кері Фурье- түрлендіру ');
>> xlabel('Уақыт (с) ');
>> ylabel('Z(t)')
```

12.3-суретінде орындалу нәтижесі келтірілген.



12.3-сурет. Кері Фурье-түрлендіруі

Қарап отырсақ, қайта қалпына келтірілген үрдіс бастапқы үрдіспен бірдей екенін көреміз.

Спектральды талдаудың мысалдары

fft процедурасын уақыт аймағында берілген үрдісті жиілік аймағында берілген үрдіске түрлендіруге қолдану үшін келесілерді орындау қажет:

- Берілген уақыт дискретінің *Ts* мәні бойынша жиілік диапазонының *Fmax* өлшемін (герцпен берілген) төмендегі формула арқылы есептеу керек:

Fmax = 1/Ts;

- Берілген үрдістің ұзақтығы *T* бойынша жиілік дискретін *df* келесі формуламен есептеу керек:

df = 1/T;

- Есептелген деректер арқылы жиілік мәндерінің векторын құрамыз, олар арқылы Фурье-кескін көрсетіледі. Соңғы амалдар келесідей орындалады:

f1 = 0:df:Fmax.

fft процедурасын қолдану нәтижесінде үрдістің жиілік аймағындағы кескіні алынады. *ifft* кері процедурасын бірінші түрлендіру нәтижесіне пайдаланса, бастапқы үрдісті уақыт аймағында қалпына келтіруге мүмкіндік береді.

Дегенмен *fft* процедурасы үрдістің тікелей Фурье-кескінін бермейді. Фурье-кескінді алу үшін келесі іс-әрекеттерді орындау керек:

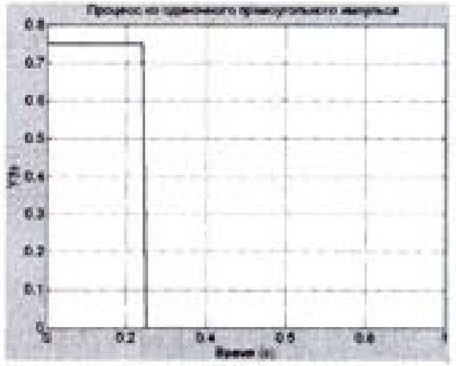
- *fft* процедурасының әрекеттерінің нәтижесіне алынған вектордың бірінші және екінші жартыларының орнын ауыстыратын *fftshift* процедурасын қолданады;
- Жиілік векторымен келесі алгоритм бойынша іс-әрекеттер жасалады:

f = -Fmax/2:df:Fmax/2.

12.2 мысал. Тік төртбұрышты импульстің Фурье-кескіні

Жалғыз тік төртбұрышты импульстен тұратын үрдісті құру керек. Уақыт дискреті *Ts = 0.01 с*, үрдіс ұзақтығы *T = 100 с*, импульс амплитудасы *A = 0.75* және оның ені *w = 0.5 с*.

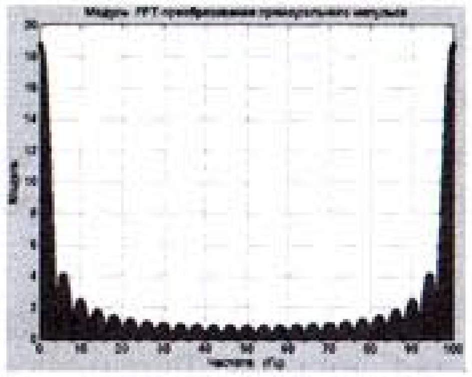
```
>> Ts = 0.01; T = 100; A = 0.75; w = 0.5;
>> t = 0:Ts:T;
>> y = A*rectpuls(t, w);
>> plot(t(1:100), y(1:100)), grid, set(gca, 'FontName', 'Arial Cyr', 'FontSize', 16),
>> title('Жалғыз тік төртбұрышты импульстен тұратын үрдіс');
>> xlabel('Уақыт (с)');
>> ylabel('Y(t)')
```



12.4-сурет. Тік төртбұрышты импульс үшін Фурье-кескін

Орындалу нәтижесі 12.4-суретінде келтірілген. *y* векторына *ffft* процедурасы қолданылады, содан кейін нәтиже модулінің жиіліктен тәуелділігінің графигі тұрғызылады. Жиілік аймағында графиктерді *stem* процедурасының көмегімен шығарған ыңғайлы (12.5-сурет):

```
>> x = fft(y);
>> df = 1/T; Fmax = 1/Ts;
>> f = 0:df:Fmax;
>> a = abs(x);
>> stem(f, a), grid, set (gca, 'FontName', 'Arial Cyr', 'FontSize', 14),
>> title('Тік төртбұрышты импульстің FFT-түрлендіруінің модулі');
>> xlabel('Жиілік (Гц)');
>> ylabel('Модуль')
```



12.5-сурет. Тік төртбұрышты импульстің *FFT*-түрлендіруінің модулінің жиіліктен тәуелділігі

fftshift(x) функциясының көмегімен үрдістің Фурье-кескінінің модулінің графигі тұрғызылады.

Fftshift функциясы (оны шақыру: *z = fftshift(y)* түрінде орындалады) берілген *y* векторынан оның екінші жартысын *z* векторының бірінші жартысының орнына қою арқылы жаңа *z* векторын құру үшін қолданылады. Ал *z* векторының екінші жартысы *y* векторының элементтерінің бірінші жартысынан тұрады. Бұл *z* бойынша периодты (периоды 2π) e^{jz} функциясының қасиетінен көрінеді. Сондықтан теріс жиіліктегі Фурье-кескін туралы ақпарат *y(k)* векторының екінші жартысында орналасады.


```
>> xp = fftshift(x),
>> f1 = -Fmax/2:df:Fmax/2;
>> a = abs(xp);
>> stem(f1,a), grid, set (gca, 'FontName', 'Arial Cyr', 'FontSize',14);
>> title(' Тік төртбұрышты импульстің Фурье-кескінінің модулі ');
>> xlabel('Жиілік (Гц)');
>> ylabel('Модуль')
```

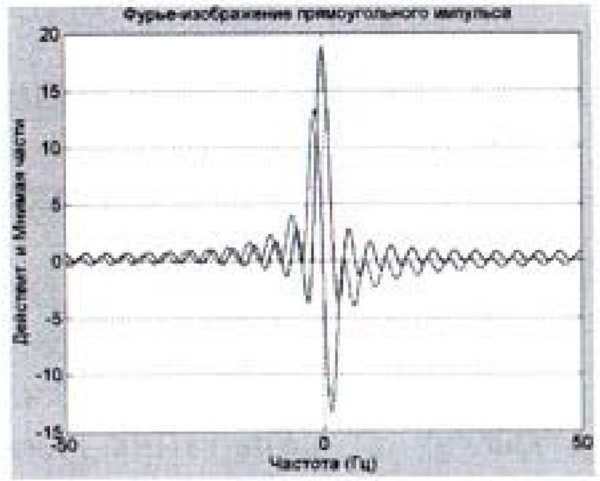
12.6-суретінде көрсетілген нәтиже алынады.

Тұжырымдай келе тік төртбұрышты импульстің Фурье-кескінінің нақты және жорамал бөліктерінің графиктері тұрғызылады

```
>> dch = real(xp), mch = imag(xp);
>> plot(f1,dch,f1,mch), grid, set (gca, 'FontName','Anal Cyr', 'FontSize', 16), title('Тік төртбұрышты импульстің Фурье-кескіні');
>> ylabel('Нақты және Жорамал бөліктер')
>> xlabel('Жиілік (Гц) ');
```

Алынған графиктер12.7-суретінде келтірілген.

12.6-сурет. Тік төртбұрышты импульстің *FFT*-түрлендіруінің модулінің жиіліктен тәуелділігінің графигі



12.7-сурет. Тік төртбұрышты импульстің Фурье-кескінінің нақты және жорамал бөліктерінің графиктері

Тапсырма

1. Жиіліктері $1/\pi$, 1 және 3 Гц, амплитудалары сәйкесінше 0.6, 0.3 және 0.7 болатын үш жиілікті гармониялық тербелістер үшін Фурье-кескінді алу керек:

$y(t) = 0.6\cos(2t) + 0.3\sin(2\pi t) + 0.7\cos(6\pi t + \pi/4)$.

2. *randn* функциясының көмегімен құрылған кездейсоқ стационарлық үрдіс үшін Фурье-кескінді алу керек. *Ts = 0.01, T = 100.*
3. *pulstran, gauspuls, rectpuls* функцияларының көмегімен құрылған типтік импульстік үрдістер үшін Фурье-кескінді алу керек.

4. Элементтері екі синусоиданың (жиіліктері 5 және 10 Гц) қосындысы болатын функцияның мәндерінен тұратын вектор түріндегі кіріс сигналын құру керек. Осы сигналдың Фурье-кескінін тауып, кіріс үрдісін және оның Фурье-кескінінің модулін график түрінде көрсету керек.
5. Элементтері екі синусоиданың (жиіліктері 4 және 15 Гц, амплитудалары сәйкесінше 0.7 және 1) қосындысы болатын функцияның мәндерінен тұратын вектор түріндегі кіріс сигналын құру керек. Осы сигналдың Фурье-кескінін тауып, кіріс үрдісін және оның Фурье-кескінінің модулін график түрінде көрсету керек.

Он үшінші сабақ. Сызықты емес теңдеулер және тиімділеу

Сабақтың жоспары

1. Бір белгісізді теңдеудің түбірін табу. *Fzero* функциясы.
2. Сызықтық емес теңдеулер жүйесін шешу. *Fsolve* функциясы.
3. Тиімділеу есептерін сандық шешу.
4. Бір айнымалылы функцияның минимумын іздеу. *Fminbnd* функциясы.
5. Көпөлшемді шартсыз минимизациялау. *Fminsearch* функциясы.
6. Шарт қою арқылы минимизациялау. *Fmincon* функциясы.

Бір белгісізді теңдеудің түбірін табу

Fun(x) = 0 түріндегі теңдеуді шешу үшін *fzero* функциясы қолданылады. Түбірі ізделіп отырған фунцияны шақырудың ең қарапайым жағдайында оған сілтеуішті қолданады, сонымен қатар түбір ізделінетін аймақтың бастапқы нүктесі *x0* беріледі:

x = fzero(f, x0).

Мұндағы *f* аргументі төмендегілердің бірі арқылы берілуі мүмкін:

- апостроф белгісіне алынған белгісіз *x*-тен құрылған формула;
- *m*-файлдың аты түрінде (апострофқа алынған және *m* кеңейтілуінсіз);
- функцияға сілтеуіш түрінде (мысалы, *@fun_name*);
- аты белгісіз функцияға сілтеуіш түрінде (мысалы, *fun_handle*).

x0 аргументі келесі екі мүмкіндіктің біреуі арқылы беріледі:

- *[a, b]* вектор (*a* < *b*) интервалында анықталған *[a, b]* векторы түрінде және *f* функциясы интервалдың шекті нүктелерінде таңбасын өзгертеді, мұның өзі осы интервалда ең болмағанда бір түбірдің табылатындығының айғағы;
- аймағында түбірдің табылуы мүмкін болып есептелетін скалярлы мән. Бұл жағдайда *fzero* функциясы центрі берілген *x0* нүктесі болатын және шекті нүктелерінде *f* функциясы таңбасын өзгертетін интервалды өзі табуға тырысады.

Бастапқы жуықтауды таңдауды жеңілдету үшін *y = fun(x)* функциясының графигін тұрғызуға болады.

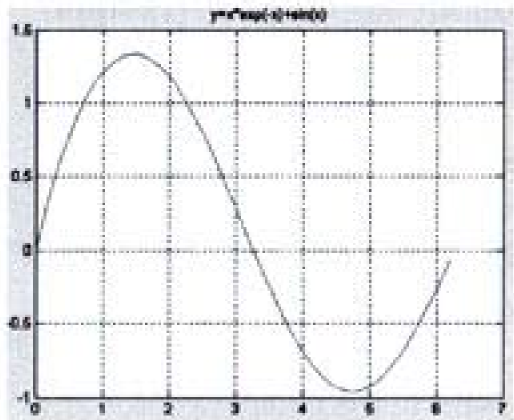
$y = x*e^{-x} + sin(x)$ функциясының графигі 13.1-суретінде көрсетілген. График 13.1 – мысалындағы программаның көмегімен алынған.

13.1 мысал. Түбірді жекешелеу үшін функцияның графигін тұрғызу:

```
>> x = 0:0.1:2*pi
>> plot (x, x.*exp(-x) + sin(x))
```

>> grid on
>> title('y = x*exp(-x) + sin(x)')

Графиктен көрініп тұрғандай, түбірлердің бірі *[3, 4]* интервалында жатыр.



13.1-сурет. Түбірді жекешелейтін функцияның графигі

Осы ақпаратты *fzero* функциясын шақырған кезде қолдану керек:

>> x = fzero('x.*exp(-x)+sin(x)', [3, 4])
x =
    3.2665

f функциясын анықтайтын формуланың орнына сәйкес функцияны хабарлап, оны автономды *m*-файл түрінде сақтаған немесе оны бағыныңқы функция ретінде программа файлына кіргізген ыңғайлы.

Осы хабарлаудан кейін *fzero* функциясы келесі түрде шақырылуы мүмкін:

function fzero1
x = fzero(@f1, [3, 4])
function y = f1(z)
y = z *exp(-z) + sin(z)

Егер теңдеудің түбірін ғана емес, функцияның табылған нүктедегі мәнін де табу қажет болса, онда *fzero* функциясы екі шығыс параметрімен шақырылады:

>> [x, f] = fzero('x.*exp(-x) + sin(x)', [3, 4])
x =
    3.2665
f =
    2.0817e-016

fzero функциясы тағы да екі шығыс параметрін қайтаруы мүмкін:

[x, f, e_flag, inform] = fzero(f, x0)

e_flag-тың оң мәні шекті нүктелерінде *f* функциясы таңбасын өзгертетіндей интервалдың табылғандығын көрсетеді. Ондай интервал табылмаған жағдайда *e_flag = -1*. *inform* құрылымында *algorithm* атымен аталған өрістер орналасқан (түбірді табу үшін қолданылған алгоритм атауы), *fcCount* (*f* шақыру саны), *intervaliterations* (интервалды табу үшін орындалған итерация саны), *iterations* (итерация саны) және *message* (шығыс хабарламасы).

Сызықтық емес теңдеулер жүйесін шешу. fsolve функциясы

fsolve функциясы *F(x) = 0* түріндегі сызықтық емес теңдеулер жүйесін шешуге арналған, мұндағы *x* – белгісіздер векторы немесе матрицасы, ал *F* – мәні вектор немесе матрица болатын функция. Бұл функцияның орындалу алгоритмі *x0* бастапқы мәнін пайдалана отырып *F* функциясының құраушыларының квадраттарының қосындысын Гаусс-Ньютон және Левенберг-Марквардт әдістерімен минимизациялауға негізделген. Қарапайым жағдайда *fsolve* функциясы келесі түрде шақырылады:

x = fsolve(F, x0)

13.2 мысал. fsolve функциясының көмегімен түбірді іздеу:

>> x = fsolve (@sin, 1)
Optimization terminated: first-order optimality is less than options.TolFun.
x =
    0

fsolve функциясының екінші аргументі вектордың бастапқы мәндері түрінде берілуі мүмкін және вектордың әрбір компоненті үшін жақын жатқан шешім табылады (13.3-мысал).

13.3 мысал. Бастапқы мәндер векторы үшін шешімдерді іздеу

>> x = fsolve(@sin, [1 2 3 4 5 6], optimset('fsolve'))
Optimization terminated: first-order optimality is less than options.TolFun.
x =
    0
    3.1416
    3.1416
    3.1416
    6.2832
    6.2832

Сызықтық емес теңдеулер жүйесін шешу *fsolve* функциясының басты мақсаты болып табылады (13.4-мысал).

13.4 мысал. Сызықтық емес теңдеулер жүйесін шешу

$y_1 = x_1 + x_2 - \sin(\pi x_1)$

$y_2 = x_1 - x_2 - \cos(\pi x_2)$

Мәні вектор-баған түрінде құрылатын *funsc* функциясын хабарлаймыз:

function y = funsc(x)
y=[x(1) + x(2) – sin(pi*x(1)); x(1) – x(2) – cos(pi*x(2))];

Әрбір шақыру кезінде бастапқы мәндерді (x_1, x_2) өзгерте отырып *fsolve* функциясына хабарласамыз. *fzero* функциясына ұқсас *fsolve* функциясы да функцияның табылған нүктедегі мәнінен құрылған вектор-бағанды қайтарады, сондықтан шешімнің дәлдігін бағалау үшін екі шығыс параметрін қолданамыз. Бастапқы нүктенің координаталары вектор-баған түрінде беріледі:

```
>> [x, f] = fsolve(@funsc, [0.2; 1], optimset ('Display', 'off '))
x =
    0.3915
    0.5510
f =
    1.0e-010*
    0.8924
    -0.0330
```

Барлық тиімділеу және сызықтық емес теңдеулер жүйесін шешу функцияларының кіріс параметрлерінің қасиеттері итерациялық үрдістердің орындалуына әсер етеді. Ол қасиеттер өрістері *optimset* функциясының көмегімен құрылатын *options* құрылымы арқылы көрсетіледі.

Тиімділеу есептерін сандық шешу

Мүмкін болатын барлық жағдайдың ішінен ең жақсысын таңдауды тиімділеу деп түсінеміз. Тиімділеу есептерін шешу барысында кейбір параметрлердің оңтайлы мәндерін табу қажет. Тиімді шешімді таңдау мақсат функциясы деп аталатын функцияның көмегімен жүзеге асырылады. Мақсат функциясын келесі түрде жазуға болады

$$u = f(x_1, x_2, ..., x_n), \qquad (13.1)$$

мұндағы $x_1, x_2, \dots, x_n$ – параметрлер.

Тиімділеу есептерінің екі түрін бөліп қарастыруға болады – *шартты және шартсыз*. Шартсыз тиімділеу есебінің мәні n нақты айнымалыдан тұратын функцияның (13.1) максимумын немесе минимумын табуға және n-өлшемді кеңістіктің қандай да бір G жиынында аргументтердің сәйкес мәндерін анықтауға тіреледі. Көбінесе минимизациялау есептері қарастырылады; мақсат функциясының таңбасын қарама-қарсыға ауыстыра отырып, максимумды іздеу есептеріне жеңіл көшуге болады. Шартты тиімділеу есептері дегеніміз – оларды құру барысында G жиынында қандай да бір шарттар (шектеулер) қойылады. *Matlab* ортасында шартсыз минимизациялау есебінің шешімдерін табудың әртүрлі мүмкіндіктерін қамтамасыз ететін бірнеше функцияны қарастыруға болады.

Бір айнымалы функцияның минимумын іздеу. fminbnd функциясы

Бір айнымалы функцияның минимумын табу үшін алтын қиық әдісі немесе параболалық интерполяция әдісі (функцияның берілу формасына байланысты) қолданылады және *fminbnd* программасының көмегімен жүзеге асырылады.

13.5 мысал. $f(x) = 24 - 2x/3 + x^2/30$ *функциясының [5; 20] аралығындағы минималды мәнін тауып баспаға жіберу керек.*

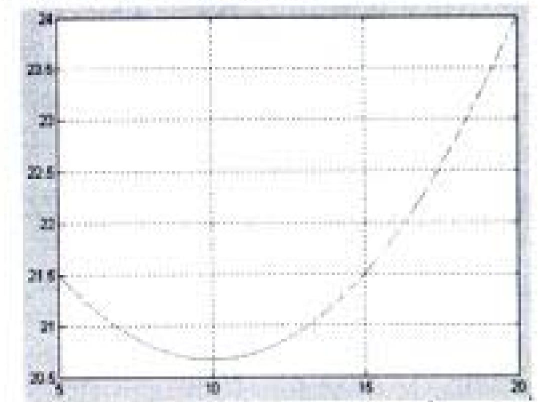
Берілген интервалда функцияның минимумының бар немесе жоқтығын тексеру үшін оның графигін тұрғызамыз.

```
>> x = 5.0:0.001:20.0; y = 24 – 2* x/3 + x.^2/30;
>> plot(x, y); grid on
```

Берілген функцияның графигі тұрғызылған терезенің пайда болуы оның минимумының бар екендігін көрсетеді (13.2-сурет).

Минимумның дәл координаталары мен мәнін анықтау үшін *fminbnd* программасын қолданамыз.

```
>> [x, y] = fminbnd ('(24.0 – 2*x/3 + x.^2/30)', 5.0, 20.0)
x =
    10.0000
y =
    20.6667
```



13.2-сурет. $f(x) = 24 - 2x/3 + x^2/30$ функциясының графигі

Көпөлшемді шартсыз минимизациялау. fminsearch функциясы

fminsearch функциясында симплекстік іздеу алгоритмі қолданылады, оның идеясы келесіде: n-өлшемді кеңістіктің бастапқы нүктесінің маңайында $(n + 1)$ – симплексі тұрғызылады, жалпы жағдайда (ешбір 3 нүкте бір түзудің бойында, ешбір 4 нүкте бір жазықтықта жатпайды және т.б.). Осы нүктелерде мақсат функциясының мәндері есептеліп, функцияның мәні максималды болатын нүкте қарастырылудан алынып тасталады, ал оның орнына ережелерге сай симплекске басқа нүкте қойылып отырады. Симплекс диаметрі берілген аралықтан кіші болған жағдайда үрдіс тоқтатылады. Мақсат функциясы тегіс емес тіпті үзілісті болуы мүмкін.

Бұл функцияға қатынасудың ең қарапайым түрі келесідей:

x = fminsearch (fun, x0).

Екі айнымалыдан тұратын үзілісті, бірақ тегіс емес функцияның мысалын қарастырайық $y = fun(x) = 3|x_1| + |x_2|$, оның жалғыз минимумы координаталар жүйесінің бастапқы нүктесінде орналасқан.

fminsearch функциясына қатынасу арқылы келесі нәтижені аламыз:

```
>> [x, f] = fminsearch('3*abs(x(1)) + abs(x(2))', [1; 1])
x =
    1.0e-004*
    -0.1439
    0.3565
```

```
f =
    7.8809e-005
```

Екі айнымалыдан тәуелді үзілісті болып табылатын *discont* функциясын қарастырайық, оның жалғыз минимумы координаталар басында жатыр:

```
function z = discont (x)
if (x(1) > 0) && (x(2) >= 0) % 1квадрант
z = x(1)+2*x(2);
elseif (x(1) <= 0) && (x(2) > 0) % 2 квадрант
z = -2*x(1)+x(2);
elseif (x(1) < 0) && (x(2) <= 0) % 3 квадрант
z = -x(1)-2*x(2);
elseif (x(1) >= 0) && (x(2) < 0) % 4 квадрант
z = 2*x(1)-x(2);
elseif (x(1) == 0) || (x(2) == 0) % координаталар басы
z = 0;
end
```

fminsearch функциясын шақыру барысында келесі нәтижелер алынды:

```
>> options = optimset('Display', 'off');
>> x0 = [0.5; 0.1];
>> [x, f, e_flag] = fminsearch(@discont, x0, options)
x =
    1.0e-003*
    0.0640
    -0.1342
f =
    2.6229e-004
e_flag =
        1
```

x_0-дің басқа мәндерінде де осыған ұқсас нәтижелер алынды.

13.6-мысалын қарастырайық.

*13.6 мысал. Розенброк банан*ы

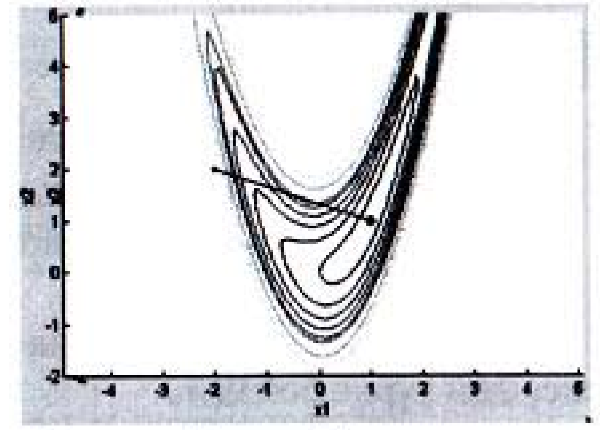
Розенброктың атақты бананы келесі түрде беріледі:

$z = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$.

Графикалық түрде ол жақтаулары тік болып келген ойысты көрсетеді, "жоғарыдан қарағанда" оның түп жағы параболаға ұқсас келеді (13.3-сурет) және ол бірқалыпты $x = [1; 1]$ минимум нүктесіне түседі, ол нүктеде $z = 0$. Оның деңгей сызықтары қисық болып келген ойысты көрсетеді (осыдан "банан"). Жинақы суретке қол жеткізу үшін бірінші қосындының алдындағы коэффициентті *100 5-пен* ауыстырамыз:


```
function f = Rosenbrock(x)
f = 5*(x(2) – x(1)^2)^2 + (1 – x(1))^2;
```

Минимумды іздеу және Розенброк функциясының деңгей сызықтарын тұрғызу үшін (13.3-сурет) *prog13_1.m* программасы қолданылады:



13.3-сурет. "Бананды" минимизациялау

```
function prog13_1
X0 = -3:0.1:3; Y0 = -2:0.1:5;
[X Y] = meshgrid(X0,Y0);
s = size(X); Z = zeros(s);
for i = 1:s(1)
for j = 1:s(2)
Z(i, j) = Rosenbrock([X(i, j); Y(i, j)]);
end
end
axes('Xlim', [-3 3],'Ylim', [-2 5]);
axis equal; grid off; hold on;
v = 1:2:10; V = 10:4:20;
contour(X,Y, Z,[v V]);
xlabel('x1'); ylabel('x2')
x0 = [-2; 2];
line(x0(1), x0(2), 'Marker', '.', 'MarkerSize', 10);
[x, f] = fminsearch('5*(x(2)-x(1)^2)^2+(1-x(1))^2', x0);
line(x(1), x(2), 'Marker', '.', 'MarkerSize', 20);
plot ([x0(1), x(1)], [x0(2), x(2)], 'k-');
colormap copper
```

Табылған минимум:

```
>> x
x =
    1.0000
    1.0000
>> f
f =
    1.8161e-009
```

Шартты минимизациялау. Fmincon функциясы

Егер нақты функцияның аргументіне теңдеулер немесе теңсіздіктер түріндегі қандай да бір шектеулер қойылса, онда $y = f(x)$ векторлық аргументтен тәуелді нақты функцияның минимумын іздеу есебі *шартты* деп аталады. Ондай есептерді шешу Лагранждың көбейткіштерін қолдану әдісіне негізделген. Әрбір $g(x) \le 0$ теңсіздігі $g(x) + v^2 = 0$ теңдеуімен алмастырылады (v^2 қосындысы алдын ала теріс емес). Одан кейін әрбір теңдеудің сол бөлігі қандай да бір көбейткішпен мақсат функциясына қосылады, ол көбейткіштер және v шамалары айнымалылар қатарына қосылады. Осы түрде модификацияланған функция үшін *шартсыз минимизациялау* есебі шешіледі.

f(x) функциясына үшін $A*x \le b$ (*A* – матрица, *b* – вектор) сызықтық шектеулер жағдайында минимизациялау есебін шешу үшін *fmincon* функциясы қолданылады, ол функцияға хабарласу үшін *x0* бастапқы нүктесі алынады:

x = fmincon (fun, x0, A, b).

fmincon функциясы келесі қосымша шектеулер қойылуын:

- $Aeq*x = beq$ – сызықтық теңдеулер (*Aeq* – матрица, *beq* – вектор);
- $lb \le x \le ub$ – координаталарға қойылатын шектеулер (*lb*, *ub* – екі векторы).

Онда *fmincon* функциясына хабарласудың қарапайым нұсқалары келесідей түрде болады:

x = fmincon(fun, x0, A, b, Aeq, beq)

x = fmincon(fun, x0, A, b, Aeq, beq, lb, ub).

Екі компонентті функция арқылы тағы бір сызықтық емес шарт қойылуы мүмкін *[c, ceq] = nonlcon(x)*, ол функция үшін ізделінді минимум нүктесі екі шектеуді қанағаттандыруы керек: $c(x) \le 0$ және $ceq(x) = 0$. Бұл жағдайда кіріс параметрлерінің тізімі *nonlcon* функциясына сілтеуішпен толықтырылады:

x = fmincon(fun, x0, A, b, Aeq, beq, lb, ub, nonlcon).

Егер жоғарыда келтірілген кіріс параметрлерінің кейбірі қолданылмайтын болса, оның орнына бос аргумент (*[]*) беріледі. Егер *x(i)*-тің кейбір компоненттеріне жоғарыдан немесе төменнен шектеулер қойылмаса, онда сәйкес векторлар компоненттері *ub* және *lb* $+\infty$ *(ub(i) = Inf)* немесе $-\infty$ *(lb(i) = -Inf)* түрінде беріледі. 13.6-мысалын қарастырайық.

13.7 мысалы. Жартыжазықтықтағы Розенброк функциясының минимумы

Жартыжазықтықты беретін және $x_1 - x_2 + 4 \le 0$ шектеулерін ескере отырып Розенброк функциясының минимумын табайық. *A = [1 -1]* матрицасын, *b = [-4]* векторын, *x0* бастапқы нүктесін анықтаймыз, *x0 = [-3; 4]* (бұл нүкте шектеуді қанағаттандырады) және *fmincon* функциясын шақырамыз:

```
>> A = [1 -1];
>> b = [-4];
>> x0 = [-3; 4];
>> x = fmincon ('5*(x(2)-x(1)^2)^2+(1-x(1))^2', x0, A, b)
x =
    2.5431
    6.5431
```


[x, fval] = fmincon(...) түріндегі толық шақыру барысында мақсаттық функцияның берілген нүктедегі мәнін табуға болады:

```
fval =
    2.4098
```

1-тапсырма

Сызықты емес теңдеудің түбірін тауып, графигін сызу керек. Берілгендер 13.1-кестесінде көрсетілген.

13.1-кесте

№	$f(x) = 0$ теңдеуі	*[a; b]* аралығы
1	$\mathrm{arctg}(x) - 1 = 0$	$[1.0;\ \sqrt{3}\ ]$
2	$e^{x-2} - \ln(x+2) = 0$	[2.0; 3.0]
3	$x^3 - 9x^2 + 5x - 6 = 0$	[8.0; 9.0]
4	$e^x - \dfrac{1}{x} - 1 = 0$	[0.5; 1.0]
5	$\mathrm{arctg}(2x) - \dfrac{1}{1+x} = 0$	[0.0; 1.0]
6	$e^x - \ln(x) - 20 = 0$	[3.0; 3.2]
7	$\sqrt{x} - \mathrm{tg}x(1-x) = 0$	[0.0; 1.0]
8	$\sin\sqrt{x} - \cos\sqrt{x} + 2\sqrt{x} = 0$	[0.0; 0.2]
9	$x^4 + 2x^3 - x - 1 = 0$	[0.8; 1.0]
10	$x^3 - e^{4x} - 5.5 = 0$	[2.6; 3.0]
11	$x^6 - 3x^2 + x - 1 = 0$	[1.0; 1.5]
12	$\sqrt[3]{5-x} - x = 0$	[1.0; 2.0]
13	$x^2 - \ln(x) = 0$	[0.0; 1.0]
14	$x^2 - \cos(x) = 0$	[0.0; 1.0]
15	$\ln(x) - \mathrm{arctg}(x) = 0$	[3.0; 4.0]

2-тапсырма

F(x) функциясының *[a;b]* аралығындағы минималды мәнін және оның координатасын тауып, қорытындылау керек. Берілгендер 13.2-кестесінде көрсетілген.

13.2-кесте

№	*f(x)* функциясы	*[a; b]* аралығы
1	$f(x)=x/\ln(x)$	[1.2; 4]
2	$f(x)=x-2\sin(x)$	[0; π/2]
3	$f(x)=(x^2-1)/(x^2+1)$	[-2; 2]
4	$f(x)=3x^4-4x^3+12x^2+1$	[-2; 2]
5	$f(x)=x-2\ln(x)$	[1; 3]
6	$f(x)=e^x\cos(x)$	[π; 3π/2]
7	$f(x)=(1-x+x^2)/(1+x-x^2)$	[0; 1]
8	$f(x)=-\sqrt{2x-x^2}$	[0; 2]
9	$f(x)=(x-2)^5(2x+1)^4$	[-0.5; 1.5]
10	$f(x)=x^x$	[0.1; 1.0]
11	$f(x)=e^{-1/x^2}$	[-0.5; 0,5]
12	$f(x)=x\sqrt{1-x^2}$	[-1.0; 0]
13	$f(x)=((e^x+e^{-x})/2)^3+1$	[-0.5; 0.5]
14	$f(x)=-x/(x^3+2)$	[0.5; 1.5]
15	$f(x)=x^2/\sqrt[3]{x^3-4}$	[1.6; 2.2]

3-тапсырма

Екі айнымалы функцияның минималды мәнін және оның координатасын тауып, қорытындылау керек. Іздеуді $M_0(x_0, y_0)$ нүктесінен бастау керек. Берілгендер 13.3-кестесінде көрсетілген.

13.3- кесте

№	*f(x, y)* функциясы	$M_0(x_0, y_0)$ бастапқы нүктесінің координаталары
1	$(2x^2-y-3)^2+x^2+2x+2$	(1; 1)
2	$(xy+2)^2+y^2+2y+4$	(2; 2)
3	$(x^2y^2-y+2)^2+x^2+1$	(2; 2)
4	$(3x^2+2y^2-1)^2+(xy-3)^2$	(2; 2)
5	$(2x^2-7y^2-2)^2+(x^2+y^2-20)^2+3$	(2; 2)
6	$(x^2+y^2-2x-3)^2+(x^2+y^2-2y-3)^2$	(2; 2)
7	$(x^2-6x+y^2+8)^2+x^2y^2+1$	(2; 2)
8	$(x^2-y-2)^2+(x-y+3)^2$	(2; 2)
9	$\ln(1+x^2+y^2)^2+(x-y-1)^2$	(2; 2)
10	$(x^2+y^2-1)^2+(x^2-6x+y^2+8)^2$	(2; 2)
11	x^3+y^3-3xy	(0.5; 0.5)
12	$x^2+xy+y^2-3x-6y$	(0.5; 3.5)
13	$-xy^2(1-x-y)$	(0; 0)
14	$3x^2-x^3+3y^2+4y$	(0.1; -1.0)
15	$xy+50/x+20/y$	(4; 1)

4-тапсырма

Функциялардың шартты минимумдарын табыңыз.

1. $f=x_1^2+x_2^2+x_3$ келесі шарттарды қанағаттандыратындай

$$\begin{cases} x_1+x_2+x_3=4, \\ 2x_1-3x_2=12. \end{cases}$$

2. $f=x_1x_2+x_2x_3$ келесі шарттарды қанағаттандыратындай

$$\begin{cases} x_1+x_2=4, \\ x_2+x_3=4. \end{cases}$$

3. $f=4x_1+x_1^2+8x_2+x_2^2$ келесі шарттарды қанағаттандыратындай

$$\begin{cases} x_1+x_2=180, \\ x_1,x_2\ge 0. \end{cases}$$

4. $f=x_1\cdot x_2$ келесі шарттарды қанағаттандыратындай

$$\begin{cases} 6x_1+4x_2\ge 12, \\ 2x_1+3x_2\le 24, \\ -3x_1+4x_2\le 12, \\ x_1,x_2\ge 0. \end{cases}$$

5. $f = 9\cdot(x_1-5)^2 + 4\cdot(x_2-6)^2$ келесі шарттарды қанағаттандыратындай

$$\begin{cases} 3x_1 + 2x_2 \ge 12, \\ x_1 - x_2 \le 6, \\ x_2 \le 4, \\ x_1, x_2 \ge 0. \end{cases}$$

6. $f = x_1^2 + x_2^2$ келесі шартты қанағаттандыратындай

$x_1 + x_2 = 5.$

7. $f = (x_1-4)^2 + (x_2-3)^2$ келесі шарттарды қанағаттандыратындай

$$\begin{cases} 2x_1 + 3x_2 \ge 6, \\ 3x_1 - 2x_2 \le 18, \\ -x_1 + 2x_2 \le 8, \\ x_1, x_2 \ge 0. \end{cases}$$

8. $f = (x_1-3)^2 + (x_2-4)^2$ келесі шарттарды қанағаттандыратындай

$$\begin{cases} 3x_1 + 2x_2 \ge 7, \\ 10x_1 - x_2 \le 8, \\ -18x_1 + 4x_2 \le 12, \\ x_1, x_2 \ge 0. \end{cases}$$

9. $f = x_2 - x_1^2 + 6x_1$ келесі шарттарды қанағаттандыратындай

$$\begin{cases} 2x_1 + 3x_2 \le 24, \\ x_1 + 2x_2 \le 15, \\ 3x_1 + 2x_2 \le 24, \\ x_2 \le 4, \\ x_1, x_2 \ge 0. \end{cases}$$

10. $f = -x_1x_2x_3$ келесі шарттарды қанағаттандыратындай

$0 \le x_1 + 2x_2 + 2x_3 \le 72$.

Он төртінші сабақ. Сызықтық программалау

Сабақтың жоспары

1. Сызықтық программалау есептерін шешу. *Linprog* функциясы.
2. Сызықтық программалаудың қосалқы есебі.
3. Матрицалық ойындар есептері.

Сызықтық программалау есептерін шешу. Linprog функциясы

Сызықтық программалау есептерін *Matlab* ортасында шешу үшін *linprog* функциясы қолданылады. Ол функция үшін іздеу аймағы келесі шарттармен беріледі:

- $A*x \le b$ – сызықтық теңсіздіктер (A – матрица, b – вектор);
- $Aeq*x = beq$ – сызықтық теңдеулер (Aeq – матрица, beq – вектор);
- $lb \le x \le ub$ – координаталарға қойылатын шектеулер (lb, ub – екі вектор).

*f'*x* в *linprog* мақсат функциясы *f* векторының коэффициенттерімен беріледі.

Ол функцияға хабарласу түрі келесідей:

x = linprog(f, A, b, Aeq, beq)
x = linprog(f, A, b, Aeq, beq, lb, ub)
x = linprog(f, A, b, Aeq, beq, lb, ub, x0)
x = linprog(f, A, b, Aeq, beq, lb, ub, x0, options)
[x, fval] = linprog(...)
[x, fval, exitflag] = linprog(...)
[x, fval, exitflag, output] = linprog(...)
[x, fval, exitflag, output, lambda] = linprog(...)

Кіріс және шығыс параметрлерінің семантикасы есептің қойылымына сәйкес. Көп жағдайда бастапқы *x0* нүктесі қолданылмайды.

linprog функциясында *тікелей қосалқы алгоритм* қолданылады, оны қолдану барысында біруақытта берілген есеппен қатар оған қосалқы есеп қатар шешіледі. Қосалқы есептің шешімі көрсетілмейді. Егер *LargeScale* қасиетіне *off* мәні қойылған болса, онда бәрімізге белгілі сызықтық программалаудың *симплекс-тәсілі* қолданылады, ол үшін бастапқы *x0* нүктесі беріледі. Егер бастапқы нүкте берілмесе, онда ол автоматты түрде таңдалады.

14.1 және 14.2 мысалдарын қарастырайық.

14.1 мысалы. Кірісті максималдау есебі

Шартты түрде үстелдер мен орындықтар шығаратын өндірісті қарастырайық. Өнімнің бір данасына қажетті ресурстардың шығындары, олардың мөлшері және шығарылған өнімнен алынатын кіріс 14.1 кестесінде келтірілген.

Берілген ресурстарды пайдалана отырып және максималды табысқа жеткізетін орындықтар мен үстелдер шығаруды жоспарлау үшін өндірісті жоспарлау есебі қойылады. Оның математикалық моделі стандартты түрдегі сызықтық программалау есебі болып табылады (СПЕ) – келесі шектеулерді қанағаттандыратындай, $f(x) = 10x_1 + 20x_2$ функциясы максималды мәнге ие болатындай x_1 және x_2 мәндерін табу керек:

$5x_1 + 25x_2 \le 500$
$0.5x_1 \le 15$
$100x_1 + 250x_2 \le 7500$
$10x_1 + 10x_2 \le 400$
$x_1 \ge 0,\ x_2 \ge 0.$

14.1-кесте

Ресурстың аты	Өнім		Ресурсқа қойылатын шектеулер
	Орындық	Үстел	
Ағаш (кг/дана)	5	25	500
Тері ($м^2$/дана)	0.5	–	15
Клей (г/дана)	100	250	7500
Еңбек шығыны (адам·сағ/дана)	10	10	400
Кіріс (теңге/дана)	10	20	

linprog функциясын шақыру үшін кіріс ақпараты келесі түрде беріледі:

- мақсат функциясының коэффициенттер векторы *f = [10; 20]*;
- сызықтық теңсіздіктер жүйесінің коэффициенттерінің матрицасы *A=[5 25; 0.5 0; 100 250; 10 10]*;
- сызықтық теңсіздіктер жүйесінің бос мүшелерінің векторы *b = [500; 15; 7500; 400]*;
- айнымалылар үшін төменгі шекаралар векторы *lb = zeros (2, 1)*.

linprog функциясы минимумды іздейді, ал есептің қойылымы бойынша максимумды табуымыз керек, сондықтан *f* мақсат функциясының коэффициенттерінің таңбасын қарама-қарсыға өзгертуіміз керек немесе ол вектордың алдына минус таңбасын қоюымыз керек:

[x, fval] = linprog(-f, A, b, [], [], lb)

prog14_1.m программасын құрайық:

```
function prog14_1
f = [10; 20];
A = [5 25; 0.5 0; 100 250; 10 10];
b = [500; 15; 7500; 400];
lb = zeros (2, 1);
[x, fval] = linprog(-f, A, b, [], [], lb)
```


Келесі нәтижелерді аламыз:

```
x =
    25.0000
    15.0000
fval =
      -550.0000
```

Сонымен мебель шығарудың тиімді шешімі табылды: 25 орындық және 15 үстел шығарсақ, кірісіміз 550 теңге болады. Жоспар бүтін сан түрінде алынды, жалпы жағдайда шешім нақты болуы да мүмкін.

Сызықтық программалаудың қосалқы есебі

Кез келген СП есебі үшін қосалқы деп аталатын есепті құруға болады. $A^*x \le b$ және $0 \le x$ шектеулерін қанағаттандыратындай f'^*x максимумын табу қажет болсын. Онда қосалқы есепте $A'^*y \ge f$ және $0 \le y$ шектеулерін қанағаттандыратындай b'^*y максимумын табу қажет.

14.1-мысалында карастырылған есепке қосалқы есепті құрайық. Теңсіздікті қажетті түрге келтіру үшін оны *-1-ге* көбейте отырып $A'^*y \le -f$ аламыз. Сонымен қатар төменгі шекара векторын да түзету қажет:

lb = zeros(4, 1)

linprog функциясын шақыру келесі түрде болады:

[y, gval] = linprog (b, -A', -f, [], [], lb)

Төмендегі нәтижені аламыз:

```
y =
    0.5000
    0.0000
    0.0000
    0.7500
gval =
      550.0000
```

Көріп отырғанымыздай мақсат функциясының мәндері бірдей (тікелей есептегі мақсат функциясына енгізілген минус таңбасын ескерсек). Мәндердің бірдей болуы кездейсоқ емес, ол туралы сызықтық программалау есебіндегі қосалқылық туралы теоремада айтылған. Айнымалылардың қосалқы мәндерінің экономикалық түсіндірмесі: олардың *ресурстардың бағаларын* көрсететіндігінде. Сонымен, y_2 және y_3, айнымалыларының нөльге тең мәндері *тері* және *клей* ресурстарына қатысты олардың бағалы емес екендігін білдіреді, себебі *25* орындық және *15* үстел шығару үшін ол ресурстардың қолдағы бар мөлшері қажеттілікті артығымен қанағаттандырады, оған көз жеткізу үшін $x_1 = 25$, $x_2 = 15$ мәндерін тікелей есептің теңсіздіктерінің сол жақ бөліктеріне қойып көруге болады. $y_1 = 0.5$ мәні ағаштың мөлшерін Δ_1–

өсімшесіне арттыру арқылы қосымша $0.5 \cdot \Delta_1$ табысқа қол жеткізуге болатындығын көрсетеді. Сонымен қатар $y_4 = 0.75$ мәні *еңбек шығынын* Δ_4 өсімшесіне арттыру арқылы $0.75 \cdot \Delta_4$ қосымша табысқа қол жеткізуге болатындығын көрсетеді.

Мысалы, $\Delta_1 = 0$, $\Delta_4 = 40$ яғни, $b = [500;\ 15;\ 7500;\ 440]$ алайық. Тікелей СПЕ шешу арқылы алатынымыз:

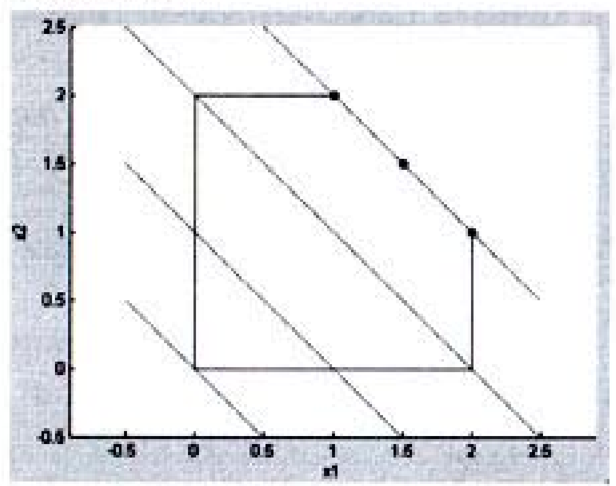
```
x =
    30.0000
    14.0000
fval =
      -580.0000
```

Көріп отырғанымыздай, кіріс шындығында да (*fval* таңбасының өзгергендігін ескерсек) $0.75 \cdot \Delta_4 = 30$ артып отыр.

СП есебінің шешімі, яғни, оптимум нүктесі, жалғыз болмауы мүмкін (мақсат функциясының тиімді мәні бірмәнді анықталғанымен).

14.2-мысал. Сызықтық программалау есебі

$0 \le x_i \le 2$ $(i = 1,\ 2)$ және $x_1 + x_2 \le 3$ шектеулерін қанағаттандыратын: $f = x_1 + x_2$. функциясының максимумын табу керек. Бұл теңсіздіктер жазықтықта 2×2 өлшемді квадрат құрайды, оның оң жақ жоғарғы төбесі қиылған (14.1-сурет).



14.1-сурет. СП есебінің геометриялық интерпретациясы (түсіндірмесі)

Деректерді даярлайық:

```
>> lb = zeros(2, 1);
>> ub = 2*ones (2, 1);
>> A = ones (1, 2);
 >> b = 3;
>> f = A;
```

және *linprog* функциясын шақырамыз:

```
>> [x, fval] = linprog(-f, A, b, [], [], lb, ub)
x =
    1.5000
    1.5000
fval =
      -3.0000
```

Мақсат функциясы *[1 2]*, *[2 1]* нүктелерімен аяқталатын кесіндінің барлық нүктелерінде бірдей мәндерді қабылдайды ($f = 3$). Кесіндінің

шекті нүктелері үшін олардың координаталарын мақсат функциясының өрнегіне қойып оңай тексеруге болады. $x = [1.5\ 1.5]$ нүктесі ол кесіндінің ортасы болып табылады.

СП есебінің екі айнымалы үшін қарапайым геометриялық түсіндірмесі бар. Айнымалылардың мүмкін мәндері тұйық дөңес көпбұрышпен шектелген, ол – айнымалылардың *мүмкін мәндерінің аймағы* деп аталады. Мақсат функциясына түзу сызық сәйкестендіріледі (біздің мысалымызда $x + y = const$). Егер мақсат функциясының сызығын параллель қозғалтатын болсақ, онда ол мүмкін мәндер аймағының алдымен бір төбесіне (немесе жағына), сонан кейін – қарама-қарсы төбесіне (немесе жағына) тиеді. Бұл жағдайдың бірі мақсат функциясының минимумына, ал екіншісі оның максимумына сәйкес келеді. Бұл түсіндірме 14.1-суретінде көрнекі түрде келтірілген, онда *linprog* функциясы арқылы табылған нүкте маркермен белгіленген. Ол суретті тұрғызу үшін *prog14_2* программасы қолданылды және алынған кескін қолмен жөнделді (мақсат функциясы штрихтелген және маркерленген).

```
function prog14_2
axes('Xlim', [-0.5 2.5], 'Ylim', [-0.5 2.5]);
axis equal; hold on;
x = [0 2 2 1 0 0]; y = [0 0 1 2 2 0];
plot(x, y, 'k');
line(2, 1, 'Marker', '.', 'MarkerSize', 20);
line(1, 2, 'Marker', '.', 'MarkerSize', 20);
line(1.5,1.5, 'Marker', '.', 'MarkerSize', 20);
xlabel('x1'); ylabel('x2');
line ([-0.5, 0.5], [0.5, -0.5])
line ([-0.5, 1.5],[1.5, -0.5])
line ([-0.5, 2.5],[2.5, -0.5])
line ([0.5, 2.5], [2.5, 0.5])
```

Матрицалық ойындар шешімдері

Сызықтық программалауды матрицалық ойындарда, яғни ойыншылардың тиімді аралас стратегияларын іздеуде қолдануға болады. Матрицалық ойын $A = (a_{ij})$ төлем матрицасымен беріледі, бұл матрицаның жолдары P бірінші ойыншының таза стратегиясына, бағандары – екінші ойыншының Q таза стратегиясына сәйкестендіріледі. Егер P – i стратегиясын, ал Q – j стратегиясын таңдаса, онда P a_{ij}-ға тең соманы алады (ал Q төлейді). P ойыншысының *аралас стратегиясы* салыстырмалы жиілігінің векторы $p = (p_i)$, ойынды қайталау кезінде ойыншы аралас стратегияның жекелеген бөліктерін қолданады. Дәл осылай Q ойыншысының да аралас стратегиясы анықталады – ол салыстырмалы жиілік векторы $q = (q_j)$. P ұтысының (Q ұтылысы) математикалық үміті олар аралас p және q стратегияларын таңдаған жағдайда $p'*A*q$ формуласымен өрнектеледі (p және q – вектор-бағандар). Кез келген таза стратегия аралас стратегияның жеке жағдайы екендігі белгілі (салыстырмалы жиіліктің бірі *1*, ал қалғандары – *0* болғанда).

Матрицалық ойындар теориясының негізгі нәтижесі – Джон фон Нейманның теоремасы бойынша келесіге тең:

$$\max_p \min_q (p'*A*q) = \min_q \max_p (p'*A*q).$$

Бұл шама v-мен белгіленеді және *ойын бағасы*, ал p және q мәндері ойын қатысушыларының *тиімді аралас стратегиялары* деп аталады. Егер ойын бағасы *0-ге* тең болса, ол *әділ* деп есептеледі.

P және q стратегияларының тиімділігінің мәні мынада: егер P тиімді p стратегиясын ұстанса, Q-дың қандай стратегияны ұстанғанынан тәуелсіз оған v-дан кем емес ұтыс (таза немесе аралас) тағайындалатындығы кепілденеді. Егер Q тиімді q стратегиясын ұстанса, P-ның қандай стратегияны ұстанғанынан тәуелсіз оған v-дан артық емес ұтылыс (таза немесе аралас) беріледі. Кепіл болатын себебі, әрбір ойынға қатысушы қарсыласының жүрісінен хабарсыз бола отырып өз жүрісін жасайды (стратегиясын таңдайды).

"Тар" матрицалы ($2\times n$ немесе $m\times 2$) ойындар үшін бағаларды және тиімді аралас стратегияларды табудың қарапайым графоаналитикалық жолдары бар, бірақ жалпы жағдайда бұл күрделі мәселе болып табылады.

$M\times n$ өлшемді $A = (a_{ij})$ төлем матрицасымен ойнаудың бағасын және ойыншылардың тиімді стратегияларын табу үшін, қосалқы сызықтық программалаудың қос есебін шешу керек:

1. $A'*x \geq 1_n$ (мұндағы 1_n – n бірліктен тұратын вектор-баған) шартын қанағаттандыратын және $F = x_1+...+x_m$ функциясы минималды мәнін қабылдайтындай $x = [x_1,..., x_m]$ векторын табу керек.
2. $A*y \leq 1_m$ (мұндағы 1_m – m бірліктен тұратын вектор-баған) шартын қанағаттандыратын және $G = y_1 + ... + y_n$ функциясы максималды мәнін қабылдайтындай $y = [y_1, ..., y_n]$ векторын табу керек.

Сызықтық программалаудың қосалқылық теоремасынан $F_{min} = G_{max}$ екендігі шығады, онда ойын бағасы $v = 1/F_{min} = 1/G_{max}$, ал тиімді аралас стратегиялар $p = v*x$, $q = v*y$ тең. $V = 0$ жағдайы 14.3 және 14.4 мысалдарында көрсетілген.

14.3 мысалы. Картамен матрицалық ойын

"Жөлік" ойынын қарастырайық. Оның ережелері келесідей: әрбір ойыншыға үш карта таратылады – қызыл тұз, қара тұз және екілік, *P-нің* қызыл, ал *Q*-дікі қара. Мұндай таратылым екі ойыншыға да белгілі. Әрбір ойыншы өзінің үш картасының ішінен біреуін таңдап оны төмен

қаратыпстөлге қояды. Содан кейін карталарды ашып көріп, ойынның аяқталуын анықтайды. Егер екі картаның да түсі бірдей болса, P ұтады, кері жағдайда – Q ұтады. Ұтыстың/ұтылыстың бағасы ұтқан адамның картасына байланысты: егер ол тұз болса – онда бір ұпай, егер екілік болса – екі ұпай. Егер екі жақтан да екіліктер ашылса, тең ойын туралы хабарланады (төлем нөлге тең). Бұл ойынның төлем матрицасы 14.2-кестеде келтірілген.

14.2-кесте

	Қызыл тұз	Қара тұз	Қара екілік
Қызыл тұз	1	-1	-2
Қара тұз	-1	1	1
Қызыл екілік	2	-1	0

Ойынның шешімін *linprog* функциясының көмегімен шешу үшін, тікелей және қосалқы есептердің берілгендерін даярлайық:

```
>> A = [1 -1 -2; -1 1 1; 2 -1 0];
>> % Тікелей есепке хабарласу:
>> [x, fval] = linprog(ones(3, 1), -A', -ones(3, 1), [], [], zeros(3, 1))
Optimization terminated successfully.
x =
    0.0000
    3.0000
    2.0000
fval =
       5.0000
>> % Кері есепке хабарласу:
>> [y, gval] = linprog(-ones (3, 1), A, ones (3, 1), [], [], zeros (3,1))
Optimization terminated successfully.
y =
    2.0000
    3.0000
    0.0000
gval =
       -5.0000
```

qval теріс мәні мақсат функциясының коэффициенттер векторының алдындағы таңбаның өзгерісіне байланысты. Ойын бағасы: $v = 1/fval = 1/5$. Аралас стратегиялар:

$p=v*x=[0\ 3/5\ 2/5]$, $q=v*y=[2/5\ 3/5\ 0]$.

Осы ойынның әділеттілігі туралы мәселені талдайық. Ережелерді ауызша талдау тұрғысынан қарастырғанда, ойын әділетті көрінуі мүмкін. P ("Алдамшы") ойыншысының Q (бұл жағдайда оны "Ашық ауыз" деп қарастыруға болады) ойыншысын ойынға көндіруге тырысқанын жеңіл есестетуге болады. Ашық ауызға барлығы дұрыс сияқты көрінеді де, ол ойнауға келіседі. Дегенмен 100 ойыннан кейін Ашық ауыздың жеңілгендігі белгілі (20 очкодан кем емес). Себебі Алдамшы өзіне тиімді стратегияны ұстанды: ол стөлге ешқашан қызыл тұзды тастаған жоқ, ойынның 60%-нда қара тұзды, ал 40%-нда қызыл екілікті оларды кездейсоқ таңдай отырып кезектей тастады. Бұл жағдайда Алдамшының қандай картаны тастағанын байқап қалмаса, Ашық ауыз не істесе де ойынның өне бойы ұтылысқа ұшырайды. Оның ұтылысы орта есеппен 100 ойында 20 ұпай, егер ол өзі үшін тиімді стратегияны ұстанса: яғни, стөлге ешқашан екілікті тастамаса және ойынның 60%-нда қара тұзды, ал 40%-нда қызыл тұзды оларды кездейсоқ таңдай отырып кезектей тастап отырса ғана жеңіске қол жеткізеді.

1-тапсырма

Сызықтық программалау есебінің шешімін табу керек. Қосалқы есепті құрып оның шешімін табыңыз. Тікелей және қосалқы есептердің арасындағы байланысты түсіндіріңіз.

1.

$$\begin{array}{ccccc} +6x_1 & -4x_2 & +8x_3 & \rightarrow & max \\ -6x_1 & +8x_2 & -8x_3 & = & 5 \\ +7x_1 & +4x_2 & +4x_3 & = & -9 \\ x_1 \geq 0 & x_2 \geq 0 & x_3 \geq 0 & & \end{array}$$

2.

$$\begin{array}{ccccc} -9x_1 & +2x_2 & -2x_3 & \rightarrow & max \\ -6x_1 & -2x_2 & -2x_3 & = & -4 \\ +5x_1 & +4x_2 & +5x_3 & = & +5 \\ x_1 \geq 0 & x_2 \geq 0 & x_3 \geq 0 & & \end{array}$$

3.

$$\begin{array}{ccccc} +5x_1 & -7x_2 & +2x_3 & \rightarrow & max \\ -6x_1 & +6x_2 & -3x_3 & = & +2 \\ +7x_1 & +4x_2 & +5x_3 & = & +2 \\ x_1 \geq 0 & x_2 \geq 0 & x_3 \geq 0 & & \end{array}$$

4.

$$\begin{array}{ccccc} +2x_1 & +8x_2 & -8x_3 & \rightarrow & max \\ -6x_1 & +6x_2 & -8x_3 & = & -3 \\ +7x_1 & +4x_2 & +5x_3 & = & +1 \\ x_1 \geq 0 & x_2 \geq 0 & x_3 \geq 0 & & \end{array}$$

5.

$$\begin{array}{ccccc} -3x_1 & +3x_2 & -7x_3 & \rightarrow & max \\ -6x_1 & +6x_2 & +3x_3 & = & +3 \\ +7x_1 & +4x_2 & +5x_3 & = & +1 \\ x_1 \geq 0 & x_2 \geq 0 & x_3 \geq 0 & & \end{array}$$

6.

$$\begin{array}{cccc} 5x_1 & -9x_2 & \rightarrow & min \\ -6x_1 & +7x_2 & \geq & +6 \\ +8x_1 & +4x_2 & \geq & -4 \\ -8x_1 & +4x_2 & \geq & +8 \\ x_1 - \text{к.к.н.с} & x_2 - \text{к.к.н.с} & & \end{array}$$

к.к.н.с. – кез келген нақты сан

7.

$$\begin{array}{cccc} -4x_1 & +5x_2 & \rightarrow & min \\ -6x_1 & +5x_2 & \geq & -9 \\ -2x_1 & +4x_2 & \geq & +2 \\ -2x_1 & +5x_2 & \geq & -2 \\ x_1 - \text{к.к.н.с} & x_2 - \text{к.к.н.с} & & \end{array}$$

8.

$$\begin{array}{cccc} +2x_1 & +2x_2 & \rightarrow & min \\ -6x_1 & +7x_2 & \geq & +5 \\ +6x_1 & +4x_2 & \geq & -7 \\ -3x_1 & +5x_2 & \geq & +2 \\ x_1 - \text{к.к.н.с} & x_2 - \text{к.к.н.с} & & \end{array}$$

9.

$$\begin{array}{cccc} -3x_1 & +1x_2 & \rightarrow & min \\ -6x_1 & +7x_2 & \geq & +2 \\ +6x_1 & +4x_2 & \geq & +8 \\ -8x_1 & +5x_2 & \geq & -8 \\ x_1 - \text{к.к.н.с} & x_2 - \text{к.к.н.с} & & \end{array}$$

10.

$$\begin{array}{cccc} +3x_1 & +1x_2 & \rightarrow & min \\ -6x_1 & +7x_2 & \geq & -3 \\ +6x_1 & +4x_2 & \geq & +3 \\ +3x_1 & +5x_2 & \geq & -7 \\ x_1 - \text{к.к.н.с} & x_2 - \text{к.к.н.с} & & \end{array}$$

11.

$$\begin{array}{ccccc} +6x_1 & -4x_2 & +8x_3 & \rightarrow & max \\ -6x_1 & +8x_2 & -8x_3 & \leq & 5 \\ +7x_1 & +4x_2 & +4x_3 & \leq & -9 \\ x_1 \geq 0 & x_2 \geq 0 & x_3 \geq 0 & & \end{array}$$

12.

$$
\begin{array}{llllr}
-9x_1 & +2x_2 & -2x_3 & \rightarrow & max \\
-6x_1 & -2x_2 & -2x_3 & \leq & -4 \\
+5x_1 & +4x_2 & +5x_3 & \leq & +5 \\
x_1 \geq 0 & x_2 \geq 0 & x_3 \geq 0 & &
\end{array}
$$

2-тапсырма

Төлем матрицасымен берілген матрицалық есептің шешімін табу керек.

A және *B* ойыншыларының қызығушылықтары әртүрлі. Ойын ($m \times n$) өлшемді A_{ij} төлем матрицасымен берілген. *A* төлем матрицасына параметр түрінде берілетін *M*-файл – функциясын жазу керек. Функцияның орындалуының нәтижесі матрицалық ойынның шешімі болуы тиіс.

1. *A* матрицасы

3	1	4
6	5	3
7	8	7
5	9	9

2. *A* матрицасы

9	1	2	4
10	10	8	7
6	9	1	7
4	3	2	6

3. *A* матрицасы

3	3	4	3	7
7	8	6	4	7
7	1	6	7	9
2	9	10	8	9

4. *A* матрицасы

1	2	2
3	8	5
4	6	4
9	10	8

5. *A* матрицасы

10	5	7	8
4	10	5	5
7	4	7	6
9	2	2	7

6. *A* матрицасы

7	10	2	9	5
9	2	9	6	5
1	2	4	2	8
10	5	6	9	8

7. *A* матрицасы

4	1	1
8	1	3
4	1	1
7	5	1

8. *A* матрицасы

2	2	3	4
1	2	10	10
9	6	5	4
9	8	9	6

9. А матрицасы

7	7	1	1	10
7	9	8	2	3
4	1	10	6	2
8	10	8	2	1

10. *A* матрицасы

8	2	8
7	2	5
9	6	8
10	6	9

11. *A* матрицасы

2	1	2	3
1	7	5	1
9	6	8	10
2	6	1	3

12. *A* матрицасы

1	6	1	5	10
4	4	4	2	8
5	2	8	2	10
10	3	9	4	2

Он бесінші сабақ. Квадраттық программалау

Сабақтың жоспары

1. Квадраттық программалау есебін шешу. *quadprog* функциясы.
2. *Quadprog* функциясын пайдаланып біркелкі жеткізу есептерін шешу.

Квадраттық программалау есебін шешу. quadprog функциясы

Квадраттық программалау есебін шешу үшін *quadprog* функциясы қолданылады.

Бұл есеп үшін іздеу аймағы сызықтық программалау есебін шешудегідей қойылады.

Квадраттық программалау есебінің мақсат функциясы $0.5*x'*H*x + f'*x$ түрінде және оған *quadprog* функциясын екі параметрмен – симметриялық H матрицасымен және f векторымен шақыру арқылы хабарласуға болады.

```
x = quadprog(H, f, A, b)
x = quadptog(H, f, A, b, Aeq, beq)
x = quadprog(H, f, A, b, Aeq, beq, lb, ub)
x = quadprog(H, f, A, b, Aeq, beq, lb, ub, x0)
x = quadprog(H, f, A, b, Aeq, beq, lb, ub, x0, options)
x = quadprog(H, f, A, b, Aeq, beq, lb, ub, x0, options, p1, p2,...)
[x, fval] = quadprog(...)
[x, fval, exitflag] = quadprog(...)
[x, fval, exitflag, output] = quadprog(...)
[x, fval, exitflag, output, lambda] = quadprog(...)
```

Егер *quadprog* функциясына хабарласуда тек жоғарғы және төменгі шекаралары немесе тек сызықтық теңдеулер берілген болса, онда сенім интервалы берілген Ньютон әдісі және сыбайлас градиенттер әдісі қолданылады. Кері жағдайда активті теру әдісі қолданылады, ол үшін бастапқы *x0* нүктесі қажет. Егер бастапқы нүкте берілмесе, онда ол автоматты түрде таңдалады. Егер options параметрінде *'LargeScale'='off'* орнатылған жағдайда да бастапқы нүкте автоматты түрде таңдалады.

15.1 мысалын қарастырайық.

15.1 мысалы. Квадраттық программалау есебі

Оң анықталған квадраттық $x_1^2 + x_2^2$ функциясының $x_1 = 2$, $x_2 = 1$, $x_1 + 2x_2 = 2$ түзулерімен шектелген үшбұрыштағы минимумын табайық. 0.5 коэффициентін есепке ала отырып квадраттық мақсат функциясының $H = 2*eye(2)$ матрицасын және $f = zeros(2, 1)$ векторын кіргіземіз. $X_1 \leq 2$ және $x_2 \leq 1$ теңсіздіктерін координаталарға шектеу ретінде алып, $ub = [2; 1]$ қоямыз, ал $x_1 + 2x_2 \geq 2$ теңсіздігін $A = [-1\ -2]$ матрицасымен және b

$= [-2]$ векторымен $A*x \leq b$ түрінде береміз. *quadprog* функциясына келесі түрде хабарласамыз:

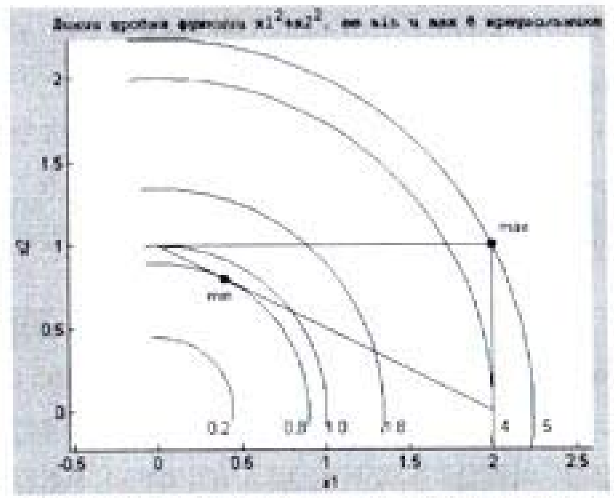
```
>> H = 2*eye(2); f = zeros(2, 1); ub = [2; 1];
>> A = [-1 -2]; b = [-2];
>> [x, fval] = quadprog(H, f, A, b, [], [], [], ub)
```

***Warning**: Large-scale method does not currently solve this problem formulation, using medium-scale method instead. Optimization terminated successfully.*

```
x =
    0.4000
    0.8000
fval =
      0.8000
```

Ескертудің жасалу себебі жоғарғы шекарамен қатар сызықтық теңсіздіктің берілуінде, сондықтан да активті теру әдісіне өту орын алады, ол әдіс үшін бастапқы *x0* нүктесі қажет. Бастапқы нүкте нақты берілмесе де функция өзінің жұмысын сәтті орындап шығады.

Алынған нәтиженің геометриялық түсіндірмесі қарапайым. *x1*, *x2* айнымалыларының мүмкін болатын өзгеру аймағы тік бұрышты үшбұрышпен берілген (15.1-суреті), ал минималдануға тиіс мақсат функциясы центрі координаталар басы болатын концентрлі шеңберлер түрінде берілген.



15.1-сурет. Квадраттық программалау есебінің геометриялық интерпретациясы

Ол шеңберлердің радиусын біртіндеп ұлғайта отырып, біз мүмкін аймақтың қабырғасымен бірінші қиылысу нүктесін табамыз, ол нүктеде мақсат функциясы өзінің минимумына тең болады. Радиусты ары қарай ұлғайтсақ, мақсат функциясы үшбұрыштың ең алыс төбесіне жетеді, ол төбеде функция максимумға ие болады. 15.1-суреті *prog15_1.m* функциясының көмегімен алынды, оның мәтіні төменде келтірілген.

```
function prog15_1
sq5 = sqrt(5);
hold on; axis equal
t = -0.1:0.05:pi/2+0.1;
cost = cos(t); sint = sin(t);
r = [1/sq5 2/sq5 1 3/sq5 2 sq5];
for i=length(r):-1:1
```

```
rc=r(i)*cost; rs=r(i)*sint;
plot(rc, rs);
end
plot([0 2 2 0], [1 0 1 1],'k-');
text(0.3, -0.1, '0.2');
text(0.75, -0.1, '0,8');
text(1.0, -0.1, '1.0');
text(1.35, -0.1, '1.8');
text(2.05, -0.1, '4');
text(2.3, -0.1, '5');
line(0.4, 0.8, 'Marker', '.', 'MarkerSize', 20);
text(0.3, 0.7, 'min');
line (2, 1, 'Marker', '.', 'MarkerSize', 20);
text(2.05, 1.1, 'max'); xlabel('x1'); ylabel('x2');
title('x1^2+x2^2 функциясының деңгей сызықтары, оның үшбұрыштағы min және max', 'FontName', 'Courier');
```

Енді берілген функцияның максимумын табайық. Ол үшін *quadprog* функциясына хабарласуда *H* матрицасы мен *f* векторының алдына минус таңбасын қоямыз:

```
>> [x, fval] = quadprog(-H, -f, A, b, [], [], [], ub)
x =
    2
    1
fval =
      -5
```

Quadprog функциясының көмегімен өнімді біркелкі жеткізу есебін шешу

Қоймаға жеткізушілерден біртекті өнім түседі (мысалы, жанармай). Қоймадан ол өнім өнеркәсіпке жіберіледі (мысалы, жылу және/немесе электрэнергиясы). Жеткізушілерден өнім біркелкі келмеуі мүмкін, бірақ өнеркәсіпке ол біркелкі жеткізілуге тиіс.

Бұл есепті дискретті қойылымда қарастырайық, яғни өнімді қабылдау – тарату $t = 1, 2, \ldots$тактілеріне сәйкес жүрсін. Қойманың жұмыс істеуін сипаттайтын шамаларды енгізейік (15.1-кесте).

Алғашқы бес параметр – кіріс параметрлері (олардың мәндері берілуі тиіс), x векторы – шығыс параметрі (есепті шешу барысында табылуға тиіс). Қоймаға түскен өнімнің жалпы мөлшері одан шығарылатын өнім мөлшеріне шамамен тең болуға тиіс:

$$\sum_{i=1}^{n} x_i \approx \sum_{i=1}^{n} p_i.$$

15.1-кесте

№	Параметр	Белгілеулер
1	Такт саны	N
2	Өнімді жеткізу векторы	$p = [p_1, p_2, \ldots, p_n]$
3	Қойманың сыйымдылығы	$maxV$
4	Қоймадағы азық-түліктің минималды нормативті мөлшері	$minV$
5	Қойманың бастапқы жүктелуі	V_0
6	Өнімді өндіріске жеткізу векторы	$x = [x_1, x_2, \ldots, x_n]$

Өнімді өнеркәсіпке максималды түрде біркелкі жеткізу (шамамен бірдей деп есептегенде), $\sum_{i=1}^{n}(x_i - Mp)^2 \to \min$ мақсат функция (мұндағы Mp – p векторының барлық компоненттері бойынша орташа, ол *mean(p)* функциясының көмегімен жүзеге асырылуы мүмкін).

Жақшаларды ашып ұқсас мүшелерді жинақтай отырып мақсат функциясын түрлендіреміз:

$$x_1^2 + x_2^2 + \ldots + x_n^2 - 2 \cdot Mp \cdot (x_1 + x_2 + \ldots + x_n) + n \cdot (Mp)^2.$$

Бұл өрнектің квадраттық бөлігін матрицалық түрде жазайық: *0.5*x'*H*x*, мұндағы *H = 2*eye(n)* матрица; сызықтық бөлігі *f'*x* тең, мұндағы *f = -2*Mp*ones(n, 1)* вектор. $N{\cdot}(Mp)^2$ бос мүшесі *quadprog* функциясына хабарласуға кіре алмайды, оны бөлек есепке алуға тура келеді.

Есепке қойылатын шектеудің мәні, қоймадағы өнімнің V_t кез келген t уақыт мезетіндегі мөлшерінің $minV \le V_t \le maxV$ теңсіздігін қанағаттандыратындығында. Ол мөлшер төмендегі қарапайым формуламен көрсетіледі:

$$v_t = v_0 + \sum_{i=1}^{t} p_i - \sum_{i=1}^{t} x_i.$$

Бұл өрнекті теңсіздіктің екі жағына қойып оны $\sum_{i=1}^{t} x_i$ қатысты шешсек, есепке қойылатын шектеулерді аламыз:

$$\left(\forall t = \overline{1,n}\right)\ V_0 + \sum_{i=1}^{t} p_i - \max V \le \sum_{i=1}^{t} x_i \le V_0 + \sum_{i=1}^{t} p_i - \min V.$$

s = cumsum(p) бұйрығын орындай отырып $\sum_{i=1}^{t} p_i$ қосындысынан тұратын s векторын алуға болады. Дәл осы іс-әрекетті x векторымен орындауға болмайды, себебі ол алдын ала белгісіз. Оның орнына жаңа

$y_t = \sum_{i=1}^{t} x_i$ айнымалыларын енгіземіз. Бұл қатынасты матрицалық түрде жазуға болады: $y = L*x$, мұндағы L матрицасы:

$$L = \begin{bmatrix} 100..0 \\ 110..0 \\ 111..0 \\ ...... \\ 111.10 \\ 111.11 \end{bmatrix}.$$

Оны L = *tril(ones(n))* бұйрығын орындай отырып құруға болады. Ескі айнымалылар кері $x = L^{-1}*y$ матрицасының көмегімен жаңалары арқылы өрнектеледі. Мақсат функциясының квадраттық бөлігі жаңа айнымалылар арқылы келесідей сипатталады: *0.5*y'*Hnew*y*, где *Hnew = inv(L)'*H*inv(L)*.

Мақсат функциясының сызықтық бөлігі *fnew'*x* тең, мұндағы *fnew = inv(L)'*f* векторы.

Жаңа айнымалыларға қойылатын шектеулерді $lb \le y \le ub$ түрінде береміз, *мұндағы* *lb*, *ub* векторлары келесі формулаларымен беріледі:

lb = (V0 – maxV) + s,

ub = (V0 – minV) + s.

V0 – maxV и *V0 – minV* скалярлары *s* векторының әрбір элементіне қосылады.

Нақты деректерді келтірейік:

```
>> n = 5; p = [10, 25, 5, 10, 30]; maxV = 40; minV = 0; V0 = 5.
```

Жоғарыда келтірілген формулалардың көмегімен *quadprog* функциясына хабарласу үшін деректерді даярлайық:

```
>> Mp = mean(p)
Mp =
      16
>> s = cumsum(p);
>> lb = (V0 - maxV) + s;
>> ub = (V0 - minV) + s;
>> L = tril(ones(n));
>> invL = inv(L);
>> TinvL = invL';
>> H = 2*eye(n);
>> Hnew = TinvL*H*invL;
>> f = -2*Mp*ones(n, 1);
>> fnew = TinvL*f;
```


Есептің шешімін жаңа *y* айнымалылары үшін табамыз:

```
>> [y, fval] = quadprog (Hnew, fnew, [], [], [],[], lb, ub)
Optimization terminated: relative function value changing by less than OPTIONS.TolFun.
y =
    13.7500
    27.5000
    41.2500
    55.0000
    71.0000
fval =
      -1.2598e+003
```

fval мақсат функциясының аномальды үлкен мәнін (оның үстіне теріс) түсіндіру оңай. Мақсат функциясының мәнін құрған кезде, оның тек квадраттық және сызықтық бөліктері есепке алынды, ал *quadprog* функциясын шақыру барысында $n(Mp)^2$ бос мүшесі есепке алынбады. Шынында $f0 = n*Mp^2 = 1280$ қосындысын *fval* функциясына қосу керек – одан қарапайым мәнді аламыз *20.25*.

Бұрынғы айнымалы *x* векторына тоқталайық:

```
>> x = invL*y
x =
    13.7500
    13.7500
    13.750,0
    13.7500
    16.0000
>> Mx = mean(x)
Mx =
      14.2000
```

Енді мақсат функциясының мәнін қалпына келтіруге болады, ол

$$\sum_{i=1}^{n} (x_i - Mp)^2 .$$

```
>> sum((x-Mp).^2)
ans =
      20.2500
```
тең.

Әрбір тактіден кейін қоймадағы қалған өнімнің мөлшерін табамыз:

```
>> V = V0 + cumsum(p) – cumsum(x)'
V =
    1.2500
    12.5000
```

3.7500
0.0000
14.0000

Нәтижеден көріп отырғанымыздай, қойманың сыйымдылығы бірде бір рет ($maxV = 40$) шектен шықпады, бірақ төртінші тактіде қойма толығымен босады.

Тапсырма

Квадраттық программалау есебінің шешімін табу керек. Берілгендер 15.2-кестеде келтірілген.

15.2-кесте

№	Есептер
1	$6x_2 - x_1^2 - 3/2\,x_2^2 + 2x_1x_2 \to max,$ $3x_1 + 4x_2 \le 12,$ $-x_1 + x_2 \le 2$ $x_1 \ge 0,\ x_2 \ge 0.$
2	$6x_2 - x_1^2 - 3/2\,x_2^2 + 2x_1x_2 \to max,$ $-x_1 + 2x_2 \le 2,\ x_1 \le 4,$ $x_1 \ge 0,\ x_2 \ge 0.$
3	$6x_2 - x_1^2 - 3/2\,x_2^2 + 2x_1x_2 \to max,$ $3x_1 + 4x_2 \le 12,$ $-x_1 - 2x_2 \le -2,$ $x_1 \ge 0,\ x_2 \ge 0.$
4	$8x_1 + 12x_2 - x_1^2 - 3/2\,x_2^2 \to max,$ $-2x_1 - x_2 \le -4,$ $2x_1 + 5x_2 \le 10,$ $x_1 \ge 0,\ x_2 \ge 0.$
5	$8x_1 + 12x_2 - x_1^2 - 3/2\,x_2^2 \to max,$ $-x_1 + 2x_2 \le 2,\ x_1 \le 6,$ $x_1 \ge 0,\ x_2 \ge 0.$
6	$8x_1 + 12x_2 - x_1^2 - 3/2\,x_2^2 \to max,$ $-3x_1 + 2x_2 \le 0,$ $4x_1 + 3x_2 \le 12,$ $x_1 \ge 0,\ x_2 \ge 0.$
7	$3x_1 - 2x_2 - 1/2\,x_1^2 - x_2^2 + x_1x_2 \to max,$ $-2x_1 - x_2 \le -2,$ $2x_1 + 3x_2 \le 6,$ $x_1 \ge 0,\ x_2 \ge 0.$
8	$6x_1 + 4x_2 - x_1^2 - 1/2\,x_2^2 - x_1x_2 \to max,$ $x_1 + 2x_2 \le 2,$ $-2x_1 + x_2 \le 0,$ $x_1 \ge 0,\ x_2 \ge 0.$
9	$6x_1 + 4x_2 - x_1^2 - 1/2\,x_2^2 - x_1x_2 \to max,$ $2x_1 + x_2 \le 2,\ x_2 \le 1,$ $x_1 \ge 0,\ x_2 \ge 0.$
10	$6x_1 + 4x_2 - x_1^2 - 1/2\,x_2^2 - x_1x_2 \to max,$ $3x_1 + 2x_2 \le 6,$ $-3x_1 - x_2 \le -3,$ $x_1 \ge 0,\ x_2 \ge 0.$
11	$8x_1 + 6x_2 - 2x_1^2 - x_2^2 \to max,$ $-x_1 + x_2 \le 1,$ $3x_1 + 2x_2 \le 6,$ $x_1 \ge 0,\ x_2 \ge 0.$
12	$8x_1 + 6x_2 - 2x_1^2 - x_2^2 \to max,$ $-x_1 + x_2 \le 1,\ x_1 \le 3,$ $x_1 \ge 0,\ x_2 \ge 0.$
13	$8x_1 + 6x_2 - 2x_1^2 - x_2^2 \to max,$ $-x_1 + x_2 \le 2,$ $3x_1 + 4x_2 \le 12,$ $x_1 \ge 0,\ x_2 \ge 0.$
14	$2x_1 + 2x_2 - x_1^2 - 2x_2^2 + 2x_1x_2 \to max,$ $4x_1 + 3x_2 \le 12,\ x_2 \le 3,$ $x_1 \ge 0,\ x_2 \ge 0.$
15	$2x_1 + 2x_2 - x_1^2 - 2x_2^2 + 2x_1x_2 \to max,$ $2x_1 + x_2 \le 4,$ $-x_1 + x_2 \le 2,$ $x_1 \ge 0,\ x_2 \ge 0.$

Әдебиеттер

1. *Кетков Ю., Кетков А., Шульц М. Matlab* 7. Программирование, Численные Методы. – Санкт-Петербург: БХВ-Петербург, 2005. – 752 с.
2. *Лазарев Ю.* Моделирование процессов и систем в *MATLAB*. Учебный курс. – Санкт-Петербург: Питер, Издательская группа БХВ, 2005. – 511 с.
3. *Поршнев С.В.* Компьютерное моделирование физических процессов в пакете *MATLAB*. – М.: Горячая линия – Телеком, 2003. – 592 с.
4. *Иглин С.П.* Математические расчеты на базе *Matlab*. – СПб.: БХВ-Петербург, 2005. – 640 с.
5. *Гонсалес Р., Вудс Р., Эддинс С.* Цифровая обработка изображений в среде *MATLAB*. – М.: Техносфера, 2006. – 616 с.
6. *Бенькович Е., Колесов Ю., Сениченков Ю.* Практическое моделирование динамических систем. – 2002. – 464 с.
7. *Дьяконов В.П. MATLAB* 7.* /R2006/R2007. Самоучитель. Тип файла: PDF. – 2008.
8. *Ануфриев И.Е., Смирнов А.Б., Смирнова Е.Н. MATLAB* 7 (Наиболее полное руководство в подлиннике) Тип файла: DjVu. – 2005.
9. *Смоленцев Н.К.* Создание *Windows*-приложений с использованием математических процедур *MATLAB*, Тип файла: PDF. – 2008.
10. Консультационный центр *Matlab*: http://www.matlab.ru
11. Образовательный математический сайт Exponenta.ru. Раздел *Matlab*: http://www.exponenta.ru/soft/matlab/matlab.asp

Оқу басылымы

Дүйсебекова Күләнда Сейтбекқызы
Мансұрова Мадина Есімханқызы

MATLAB-та
ПРОГРАММАЛАУ НЕГІЗДЕРІ

Оқу құралы

ИБ № 5138

Басуға 04.05.2011 жылы қол қойылды. Пішімі 60х84 1/16. Көлемі 8,875 б.т.
Офсетті қағаз. Сандық басылыс. Тапсырыс № 389. Таралымы 140 дана. Бағасы келісімді.
Әл-Фараби атындағы Қазақ ұлттық университетінің «Қазақ университеті» баспасы.
050040, Алматы қаласы, әл-Фараби даңғылы, 71.

«Қазақ университеті» баспаханасында басылды