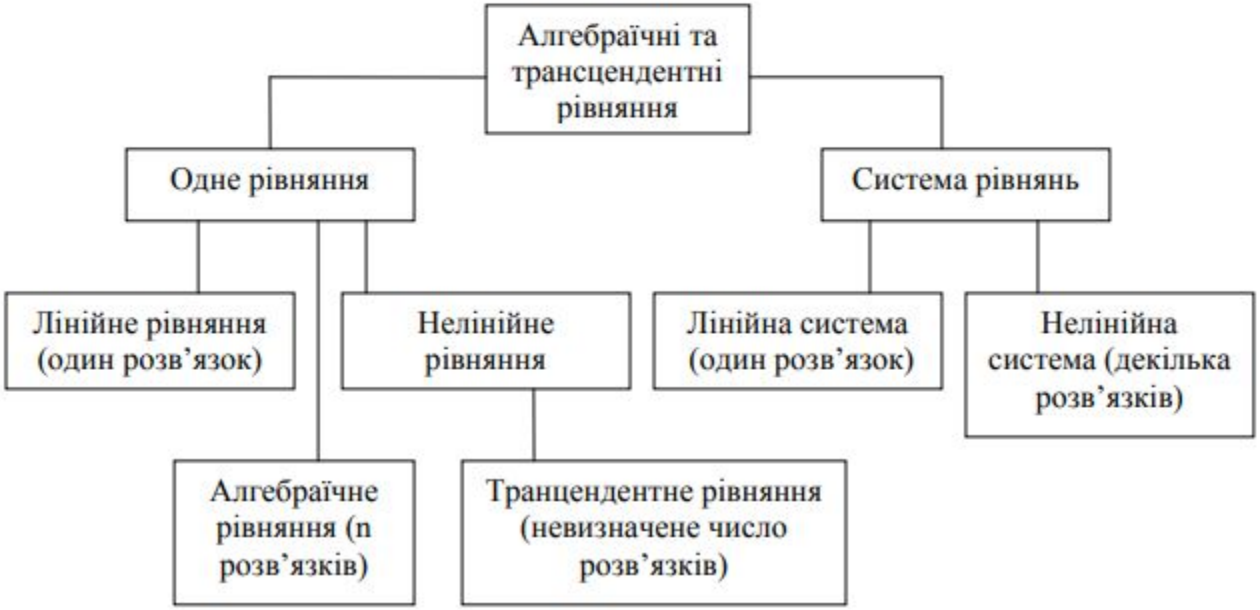
Розв’язання лінійних рівнянь та систем лінійних рівнянь в пакеті MATLAB.



Мал. 6.1 – Класифікація рівнянь

Будемо називати рівняння алгебраїчним або трансцендентним залежно від того, чи має воно n розв'язків, або має невизначене число розв'язків. У випадку, коли $n=1$, алгебраїчне рівняння називається лінійним.

Систему рівнянь будемо називати лінійною або нелінійною в залежності від виду рівнянь (лінійних або нелінійних), які формують систему.

Трансцендентне рівняння — рівняння, що містить трансцендентну функцію.

Приклади таких рівнянь:

$$x = e^{-x}$$

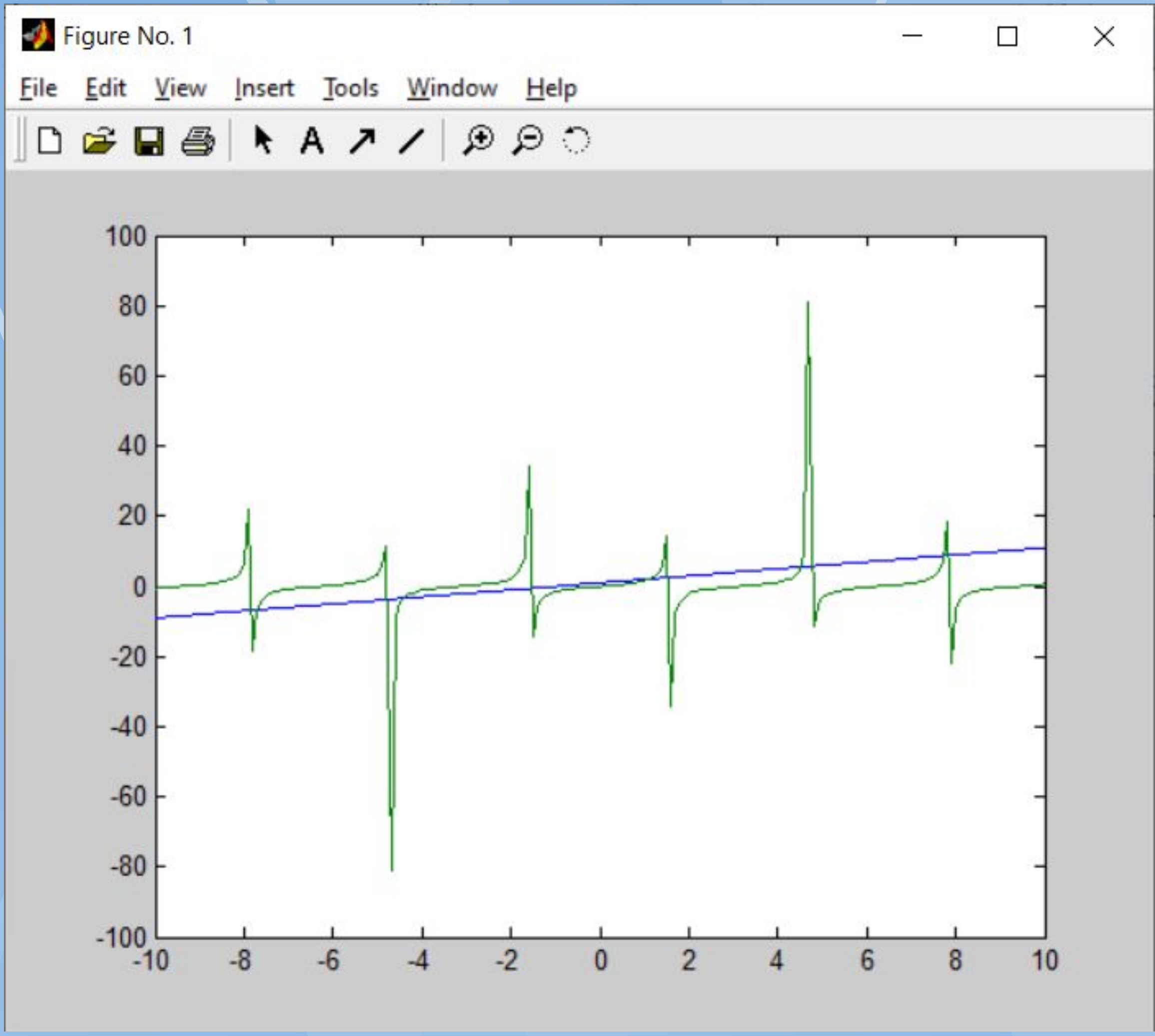
$$x = \sin(x)$$

Деякі підходи пошуку розв'язків трансцендентних рівнянь використовують графічні або чисельні методи.

Один з методів графічного розв'язку полягає в тому, що необхідно прирівняти частини рівняння із залежною змінною до частин рівняння із незалежною змінною і побудувати графіки функцій, отриманих по обидва боки знаку рівності. Точки перетину графіків цих функцій і є розв'язками рівняння.

Розв'язати трансцендентне рівняння x-tg(x)+1=0

>> x=-10:0.1:10;
>> plot(x,x+1,x,tan(x))

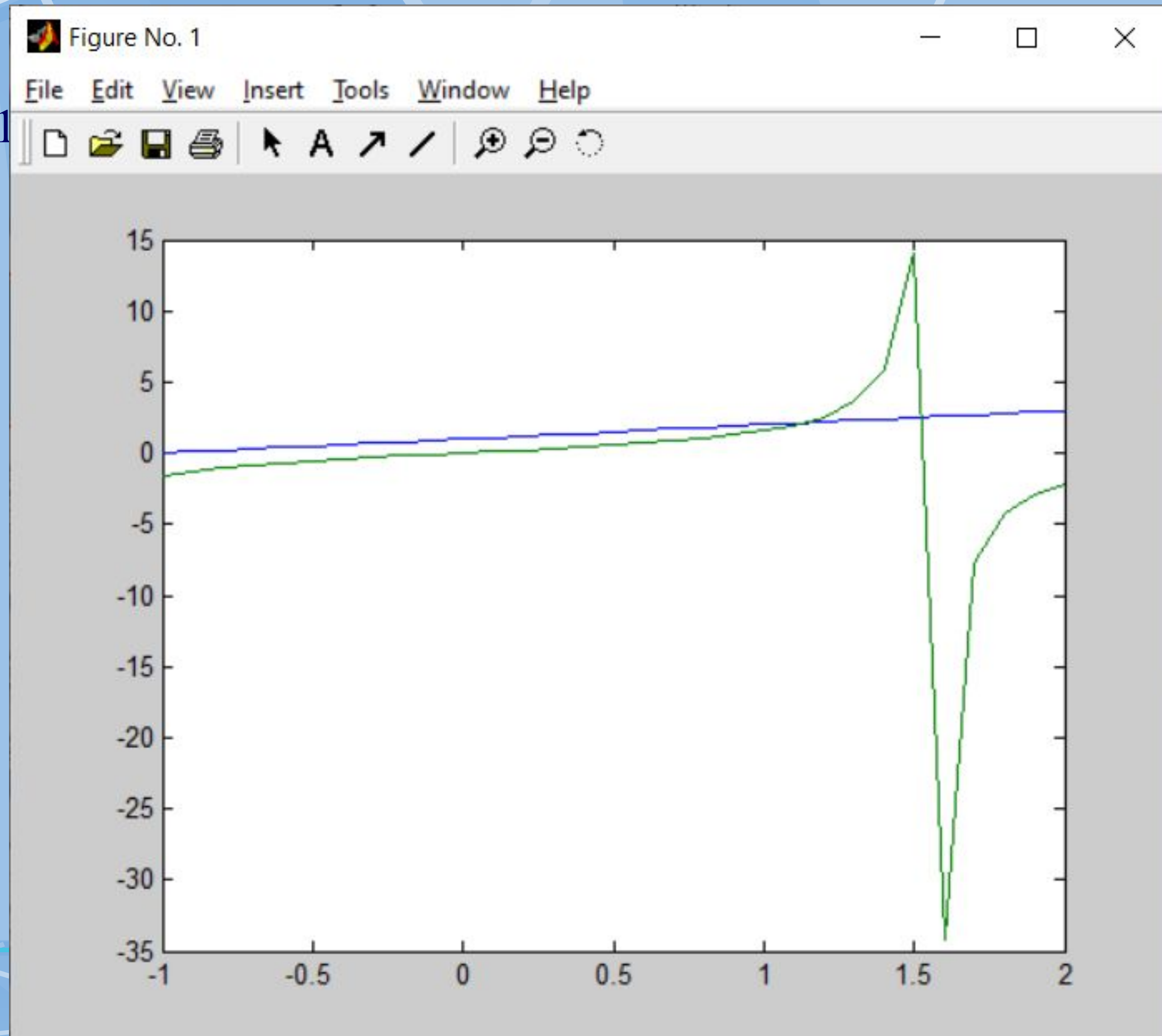


Перетин графіків двох функцій **x+1** та **tan(x)** Визначають розв'язки вихідного рівняння

>> solve('x-tan(x)+1')
1.1322677252728851316254206969360

```
>> x=-1:0.1:2;
>> plot(x,x+1,x,tan(x))
>> solve('x-tan(x)+1')
ans =

1.132267725272885131
```



Якщо існує більше одного рішення, тоді ви можете в числовій формі знайти (приблизно) рішення, показані на графіку, за допомогою команди **fzero**, яка шукає нульове значення даної функції в межах заданого значення x.
Рішення рівняння **x-tg(x)+1=0** дорівнює нулю функції x-tg(x)+1, тому, щоб знайти приблизне рішення при x = 0, введіть наступне:

```
>> fzero('x-tan(x)+1',0)

ans =

   1.1323

>> fzero('x-tan(x)+1',2)

ans =

   1.5708
```

```
>> fzero('x-tan(x)+1',5)

ans =

   4.7124

>> fzero('x-tan(x)+1',8)

ans =

   7.8540
```

```
>> fzero('x-tan(x)+1',-8)

ans =

  -7.8540
```

Вправа 1. Розв'язування алгебраїчних рівнянь та найпростіших систем рівнянь. Символьний розв'язок рівнянь на основі використання підсистеми **Symbolyc Math**.

Алгебраїчне рівняння має представлення у вигляді $P_n(x)=0$, де $P_n(x)$ – многочлен n-ї степені від декількох незалежних змінних.

Алгебраїчним рівнянням з однією невідомою називається рівняння, яке приводиться до виду $a_0x^n + a_1x^{n-1} + a_2x^{n-2} + ... + a_{n-1}x + a_n = 0$, де n – додатне ціле число (порядок многочлену).

Значення невідомого x, яке перетворює алгебраїчне рівняння у тотожність, називаються коренем (розв'язком) алгебраїчного рівняння.

Для аналітичного розв'язання алгебраїчного рівняння у системі **MATLAB** використовується команда **solve** (з підсистеми комп'ютерної алгебри **Symbolyc Math,** [20,с.45]), яка формається у наступному вигляді: **solve('eqn1','eqn2',...,'eqnN')** або **solve('eqn1','eqn2',...,'eqnN','var1','var2',..., 'varN')**, де **eqn1, eqn2**,… – одне або декілька рівнянь з символьними змінними, які використовуються для представлення рівнянь, **var1, var2**,… – шукані невідомі (символьні) змінні.

Приклад 1. Знайти розв'язок рівняння $p\sin x = r$.

Ця проста задача розв'язується наступним чином:

```
> syms p r x
>> solve('p*sin(x)=r')
ans =
asin(r/p)
```

Необхідно мати на увазі, що при розв’язані системи рівнянь для зберігання результату створюється структура з полями, яка формується за допомогою квадратних дужок, в яких виписані імена шуканих невідомих (змінних).

Приклад 2. Знайти розв’язок системи рівнянь $\begin{cases} x_1 + 2x_2 = 12 \\ x_1 + x_2 = 18 \end{cases}$ та призначити отримані результати змінним x_1, x_2.

Розв’язання прикладу можливе наступними командами:

```
>> [x1,x2]=solve('x1+2*x2=12','x1+x2=18')
x1 =
24
x2 =
–6
```

Можна також використовувати інший варіант формування команди розв’язання рівняння або системи рівнянь, а саме **s = solve(eqn1,eqn2,…)**.

Приклад 3. Розв’язати систему рівнянь $\begin{cases} x + y = 2 \\ x - 5y = 1 \end{cases}$ відносно невідомих x, y.

Використовуємо згадану вище команду:

```
>> s=solve('x+y=2','x–5*y=1')
s =
   x: [1x1 sym]
   y: [1x1 sym]
```

Для того, щоб отримати лише один з можливих розв'язків системи (наприклад, x), треба скористатися такою командою:

```
>> s.x
ans =
11/6
>> s.y
ans =
1/6
```

Зауважимо, що команда **solve** у системі **MATLAB** не розв'язує нерівностей. Аналогічна команда, яка входить до складу математичної системи **Maple,** дозволяє здійснювати цю процедуру.

Приклад розв’язування систем нелінійних рівнянь

$$\begin{cases} \cos(x-1) + y = 1 \\ \sin y + 2x = 1{,}6 \end{cases}$$

```
>> syms x y
>> [a,b]=solve('cos(x-1)+y=1','sin(y)+2*x=1.6')

a =

.78890158634890361275565773799685


b =

.22198650432731924171432392891918e-1
```

Розв’язання систем лінійних рівнянь прямими методами.

Підходи до розв’язання систем лінійних рівнянь [24] складаються з двох напрямків:

- прямі методи, які використовуються для обчислення коренів системи; до таких методів відносять: розв’язання систем за допомогою оберненої матриці, на основі використання правила Крамера, методу Гауса та інші)
- ітераційні методи, що дозволяють отримати розв’язок системи з заданою точністю шляхом ітераційних процесів, які збігаються (метод ітерації, метод Зейделя та інших).

Метод розв’язання задачі називають прямим, якщо він дозволяє отримати розв’язок після виконання скінченного числа елементарних операцій. До прямих методів розв’язання відносять метод Гауса і його модифікації (наприклад, метод Холецького (дивись далі)).

Зауважимо, що внаслідок округлень, що мають місце при розрахунках на комп’ютері, результати навіть точних методів розглядаються як наближені.

Нехай маємо алгебраїчну неоднорідну систему з n лінійних рівнянь з n невідомими $x_1, x_2, ..., x_n$:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2, \\ \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases} \qquad (1)$$

Дана система лінійних рівнянь може бути записана у матричному вигляді:

$$AX = B, \qquad (2)$$

де

$$\begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ \ldots \\ x_n \end{bmatrix}, B = \begin{bmatrix} b_1 \\ b_2 \\ \ldots \\ b_n \end{bmatrix}. \qquad (3)$$

Матриця **А**, стовпцями якої є коефіцієнти при відповідних невідомих, а рядками – коефіцієнти при невідомих у відповідному рівнянні, називається матрицею системи; матриця-стовпець В, елементами якої є праві частини рівнянь системи, називається матрицею правої.

Матриця-стовпець **Х**, елементи якої є шуканими невідомими, називається розв’язком вихідної лінійної системи. Розширена матриця системи – це матриця **D**, що утворюється з елементів матриці А і стовпця В**:** **D=[A B].**

Основними чинниками системи, від яких залежить існування і єдиність розв'язання, є:

- n – число невідомих змінних (число стовпців матриці коефіцієнтів A);
- r – ранг матриці коефіцієнтів: **r=rank(A);**
- R – ранг розширеної матриці: **R=rank([A B]).**

Нагадаємо твердження:

- система сумісна (має розв'язок) тоді і тільки тоді, коли **r=R**; у випадку **r<R** система несумісна (тобто розв'язку не існує);
- розв'язок системи єдиний тоді і тільки тоді, коли **r=R=n**; у випадку **r=R<n** система є невизначеною (тобто має нескінченно багато розв'язків).

Розглянемо матричну систему **AX=B**, у який ранг матриці коефіцієнтів **r=R=m=n**, **det(A)≠0**.

Зауважимо, що однією матричною операцією можна вирішити відразу декілька систем лінійних рівнянь з однією і тією ж матрицею коефіцієнтів системи, але з різними наборами правих частин – матриці **B** та **A**.

Якщо матриця **A** – неособлива (несингулярна, невироджена), тобто $\det A \neq 0$, то система (1) або еквівалентне їй матричне рівняння (2), має єдиний розв'язок. Дійсно, якщо $\det A \neq 0$, існує обернена матриця A^{-1}.

Множення обох частин рівняння (2) на матрицю A^{-1} дозволяє отримати наступне рівняння: $\mathbf{A^{-1}AX=A^{-1}B}$. Звідки

$$X=A^{-1}B, \qquad (4)$$

Так як $A^{-1}A=E$, $EX=X$, Е т – одинична матриця. Формула (4) дає єдиний розв'язок рівняння (2).

У системі **MATLAB** для розв'язання систем лінійних рявнянь існують декілька методів. Простішим методом розв'язання є метод вирішувачів систем лінійних рівнянь:

- оператор **X=B\A** знаходить розв'язок системи рівнянь вигляду **AX=B**, де **A** – прямокутна матриця розміром **m×n, B** – матриця розміром **m×1**;
- оператор **X=B/A** знаходить розв'язок системи рівнянь вигляду **XA=B**, де **A** – прямокутна матриця розміром **n×m**, **B** – матриця розміром n×**1.**

Приклад 1. Виконати розв'язання алгебраїчної системи

$$\begin{cases} x_1 + x_2 + x_3 + x_4 = 1 \\ x_1 + x_2 + x_3 + x_4 = 1 \\ x_1 + x_2 + x_3 + x_4 = 1 \end{cases}.$$

При розв'язуванні задачі необхідно звернути увагу, що розмір матриці А дорівнює 3×4, а розмір вектору В дорівнює 3×1. В цьому випадку можна скористатися оператором **X=B\A:**

```
>> a=[1 1 1 1;1 1 1 1;1 1 1 1]; b=[1; 1; 1]; x1=b\a
x1 =
   1    1    1    1
```

У разі, коли ранг **r=R<n,** алгебраїчна система є невизначеною (має нескінченно багато розв'зків). Загальне розв'язання такої системи може бути виписане у вигляді $x = x_0 + y$, де x_0 – частинний розв'язок неоднорідної системи **AX=B**, $B \neq 0$; y – загальний розв'язок однорідної системи **AX=0**.

Можливий частинний розв'язок неоднорідної системи **AX=B** може бути отриманий за допомогою лівого матричного ділення **X=A\B**. При виконанні операції **A\B** система **MATLAB** видає попередження, яке означає що «матриця близька до сингулярної (виродженої) матриці або погано масштабована». В цьому випадку результати розв'язання можуть бути неточними.

Приклад 2. Знайти розв'язок системи лінійних алгебраїчних рівнянь

$$\begin{cases} 2x_1 + 7x_2 + 3x_3 + x_4 = 5 \\ x_1 + 3x_2 + 5x_3 - 2x_4 = 3 \\ x_1 + 5x_2 - 9x_3 + 9x_4 = 1 \\ 5x_1 + 18x_2 + 4x_3 + 5x_4 = 12 \end{cases}.$$

Перелік команд, що наведений нижче, дозволяє здійснити процедуру розв'язання вихідної системи:

```
>> A=[2 7 3 1;1 3 5 –2;1 5 –9 8;5 18 4 5]; b=[5;3;1;12]; x=A\b
Warning: Matrix is close to singular or badly scaled.
        Results may be inaccurate. RCOND = 4.077954e-019.
(Type "warning off MATLAB:nearlySingularMatrix" to
suppress this warning.)
x=
  –5.5000
   1.2500
   1.7500
   2.0000
```

(Зауваження: матриця близька до сингулярної або погано масштабована. Результати можуть бути неточними).

Найбільший інтерес серед частинних розв'язків неоднорідної системи має так званий *нормальний розв'язок*. У системі **MATLAB** він може бути отриманий за допомогою псевдооберненої матриці згідно формули: **x=pinv(A)*B.**

Команда **pinv(A)** виконує операцію обчислення матриці, яка є псевдооберненою до матриці **A**. Вона має тіж самі розміри, що і матриця $\mathbf{A}^{-1}$. Зміст команди витікає з наступних тверджень: A*P*A=A, P*A*P=P.

Приклад 3. Знайти корені системи рівнянь приклада 2 за допомогою команди **pinv(A).**

Розв'язання здійснюється наступними командами:

```
>> pA=pinv(A)
pA =
    0.0045    0.0030    0.0001    0.0105
    0.0147    0.0071    0.0081    0.0371
    0.0138    0.0275   –0.0549    0.0139
   –0.0028   –0.0150    0.0393    0.0066
>> x=pA*b
x =
   0.1576
   0.5478
   0.2636
   0.0594
```

Вправа 3. Розв'язування системи лінійних алгебраїчних рівнянь за методом Крамера.

Про особливості методу Крамера радимо звернутися до [1].

Приклад 1. Розв'язати методом Крамера систему лінійних алгебраїчних рівнянь

$$\begin{cases} 10x_1 + x_2 + x_3 = 12 \\ 2x_1 + 10x_2 + x_3 = 13 \\ 2x_1 + 2x_2 + 10x_3 = 14 \end{cases}.$$

Команди, що допомагають розв'язати задачу, наступні:

```
A = [10 1 1; 2 10 1; 2 2 10];b = [12;13;14];
```

Перевірка невиродженості системи:

```
>> rank(A)
ans = 4
```

Описуємо допоміжні матриці згідно правилу Крамера

```
A1 = A;A2 = A;A3 = A;
A1(:,1) = b;A2(:,2) = b;A3(:,3) = b;
>> x1 = det(A1) / det(A);x2 = det(A2) / det(A);
x3 = det(A3) / det(A); x=[x1;x2;x3]
```

Перевірка розв'язку:

```
A*x–b
ans =
   0
   0
   0
```

Таким чином, шуканий розв'язок має вигляд: $x_1 = 1, x_2 = 1, x_3 = 1$.

Вправа 4. Розв'язування системи лінійних алгебраїчних рівнянь методом Гауса.

У методі Гауса для обчислення масштабуючих множників вимагається ділити відповідні елементи матриці на провідні елементи системи на кожному кроці. Якщо елемент дорівнює нулю або близький до нуля, то можливе неконтрольоване зростання похибки. Це обумовлює застосування модифікації методу Гауса, що мають кращі обчислювальні властивості.

Розглянемо метод Гауса (метод виключення провідного елементу) з вибором провідного елементу по стовпцю (схема часткового вибору). Провідний елемент вибирається на k-му кроці прямого ходу як максимальний по модулю коефіцієнт $a_{i_1 k_1}$ при невідомому x_k в рівняннях з номерами $i = k+1, \ldots, m$. Рівняння, яке відповідає вибраному коефіцієнту з номером i_k, міняють місцями з k-им рівнянням системи з метою, щоб провідний елемент зайняв місце коефіцієнту $a_{kk}^{(k-1)}$. Після цієї перестановки проводять виключення згідно схеми єдиного розподілу. В результаті всі масштабуючі множники будуть по модулю менші одиниці і схема отримує обчислювальну стійкість.

Іншими словами, метод полягає в тому, що систему (1) приводять послідовним виключенням невідомих до еквівалентної системи з верхньою трикутною матрицею:

$$\begin{cases} x_1 + \alpha_{12}x_2 + ... + \alpha_{1n}x_n = \beta_1, \\ \qquad x_2 + ... + \alpha_{2n}x_n = \beta_2, \\ .... \\ \qquad\qquad x_n = \beta_n, \end{cases} \tag{5}$$

розв'язання якої відбувається за рекурентними формулами:

$$x_n = \beta_n,\ x_i = \beta_i - \sum_{j=i+1}^{n} \alpha_{ij}x_j,\ (i = n-1, n-2, ..., 1) \tag{6}$$

В матричній формі це відповідає наступному: спочатку елементарними операціями над рядками розширену матрицю системи приводять до спеціального вигляду (прямий хід методу Гауса):

$$\begin{bmatrix} a_{11} & a_{12} & ... & a_{1n} & b_1 \\ a_{21} & a_{22} & ... & a_{2n} & b_2 \\ ... & ... & ... & ... & ... \\ a_{n1} & a_{n2} & ... & a_{nn} & b_n \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & \alpha_{12} & ... & \alpha_{12} & \beta_1 \\ 0 & 1 & ... & \alpha_{2n} & \beta_2 \\ ... & ... & ... & ... & ... \\ 0 & 0 & ... & 1 & \beta_n \end{bmatrix}, \tag{7}$$

а потім (зворотний хід методу Гауса) цю матрицю перетворюють до вигляду:

$$\begin{bmatrix} 1 & 0 & ... & 0 & x_1 \\ 0 & 1 & ... & 0 & x_2 \\ ... & ... & ... & ... & ... \\ 0 & 0 & ... & 1 & x_n \end{bmatrix} \tag{8}$$

Останній, $(n+1)$-й, стовпець цієї матриці містить в собі шуканий розв'язок вихідної системи (1).

У термінах системи **MATLAB** рівняння (4) еквівалентне виразу **x=inv(A)*B**. Зверніть увагу на те, щоб розмірності масивів **A** і **B** були погоджені**.** Для квадратних матриць відшукується точний розв'язок; у випадку перевизначених систем використовується метод найменших квадратів.

Приклад 1 Здійснити розв'язання системи лінійних алгебраїчних рівнянь

$$\begin{cases} x_1 + 4x_2 = 12 \\ 3x_1 - 2x_2 = -6 \end{cases}.$$

Для цього скористаємося наступним переліком команд:

```
>> A=[1 4;3 –2]; B=[12; –6]; X=A\B
X =
   0
   3
```

Функції для маніпуляції з елементами матриць

Назва	Значення
diag(X)	створення або виділення діагоналі
fliplr(X)	відображення елементів X відносно“вертикальної” осі
flipud(X)	відображення елементів X відносно“горизонтальної” осі
isreal(A)	Істина, якщо всі елементи матриці A дійсні
reshape(A)	розмір матриці
tril(A,k)	ліва трикутна матриця (k=0) k>0 - над діагоналлю k<0 - під діагоналлю
triu(A,k)	права трикутна матриця (k=0) k>0 - над діагоналлю k<0 - під діагоналлю
:	виділення стовпця/рядка в матриці, генерація векторів

Маніпуляції матрицями, матричний аналіз, власні числа

Назва	Значення
eye(N)	створює одиничну матрицю розміру NxN
zeros(N)	створює матрицю розміру NxN, що містить нулі. Якщо N не скаляр, то з'явиться повідомлення про помилку
det(A)	визначник (детермінант) квадратної матриці A
norm(A,n)	норма матриці/вектора, n=1,2,Inf,’fro’. norm(A)≡norm(A,2); ‘fro’-Фробеніусова
rank(A)	ранг матриці A
trace(A)	слід матриці A
orth(A)	ортонормальний базис матриці A. h=orth(A); h’*h=E
inv(A)	повертає матрицю, обернену до квадратної матриці A. Попереджуюче повідомлення видається, якщо A погано обумовлена або близька до виродженої
Size(A,1)	Повертає число рядків матриці A
Size(A,2)	Повертає число стовпців матриці A

Назва	Значення
dot(X,Y)	скалярний добуток векторівX,Y
cross(X,Y)	векторний добуток векторів X,Y
[H,L]=eig(A)	власні числа L і власні вектори H, деA*H = L*H
poly(A)	характеристичний поліном для матриці A
[U,S,V]=svd(A)	розклад A=U*S*V’, деU,V-унітарні матриці, S-діагональна

Зауваження: на практиці обчислення оберненої матриці в явному вигляді не так і необхідно. Цю операцію застосовують при розв’язку системи лінійних рівнянь виду **Ax=b.** Дійсно, один зі шляхів обчислення **x=inv(A)*b.**

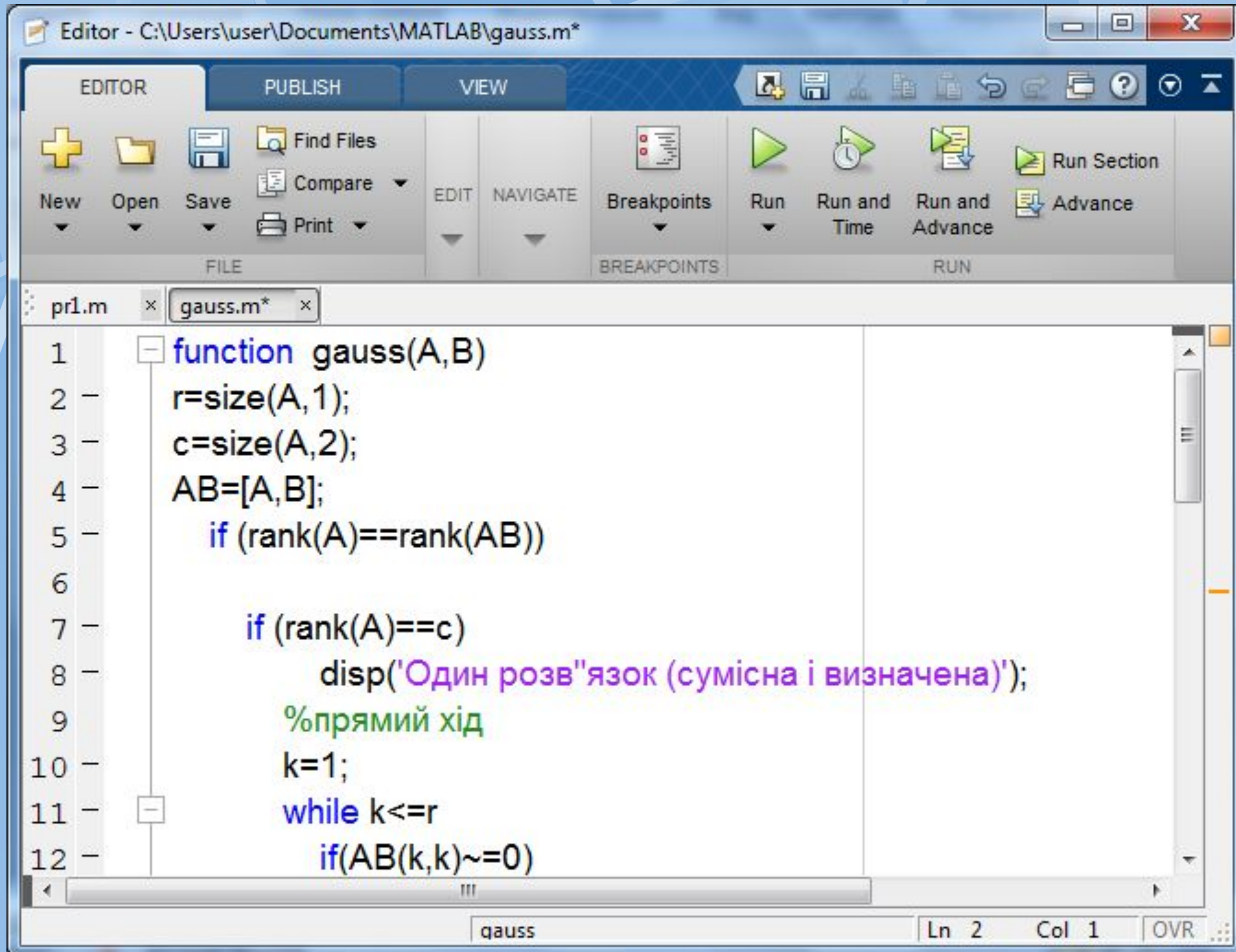
Але кращим (з погляду мінімізації часу розрахунку) є використання оператора матричного ділення **x=A\b.** Ця операція використовує метод виключення Гауса без явного формування зворотної матриці.

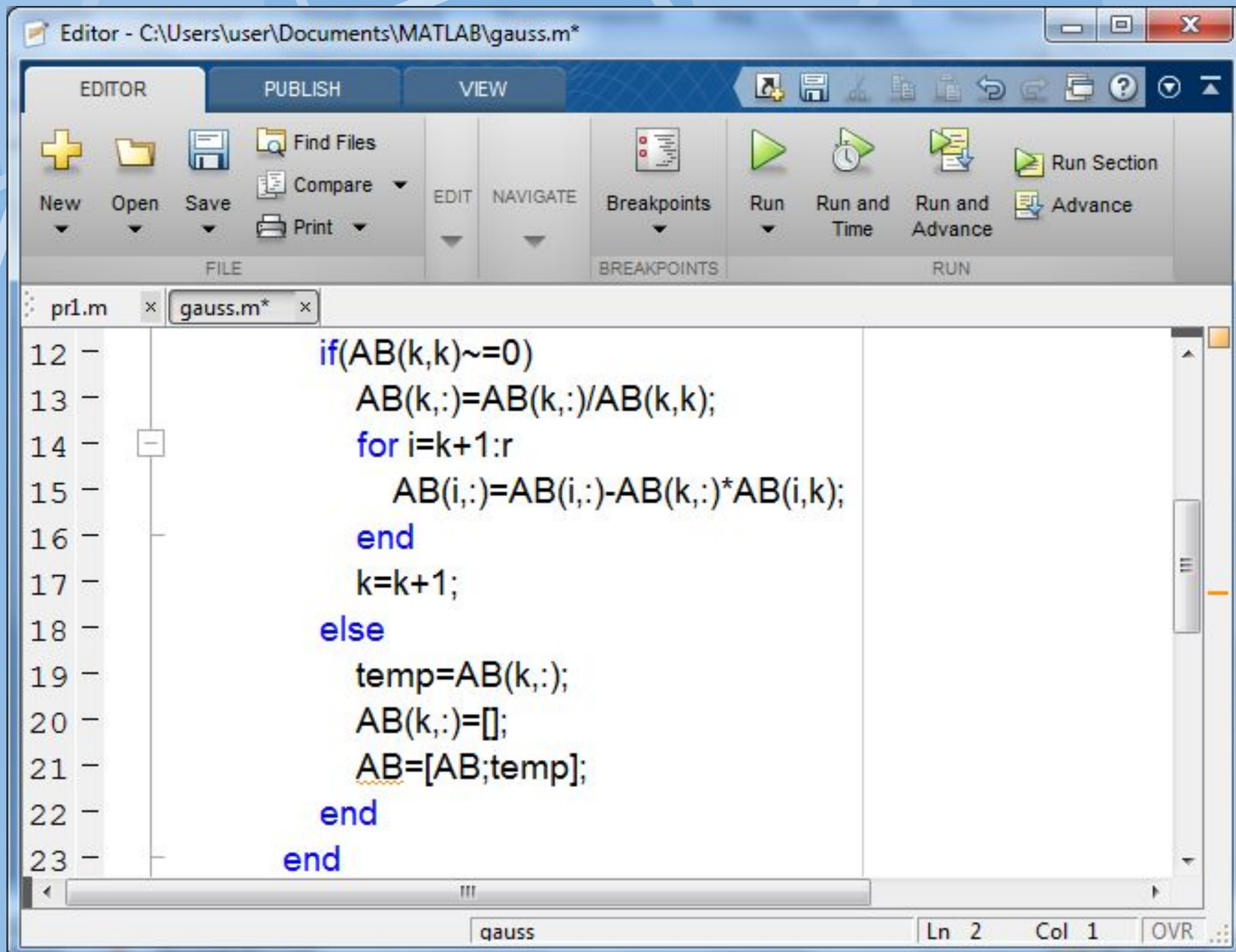
У системі Matlab розв’язування матричного рівняння здійснюється просто, з використанням дії зворотного ділення. Для прикладу розглянемо задачу відшукування коренів системи лінійних алгебричних рівнянь:

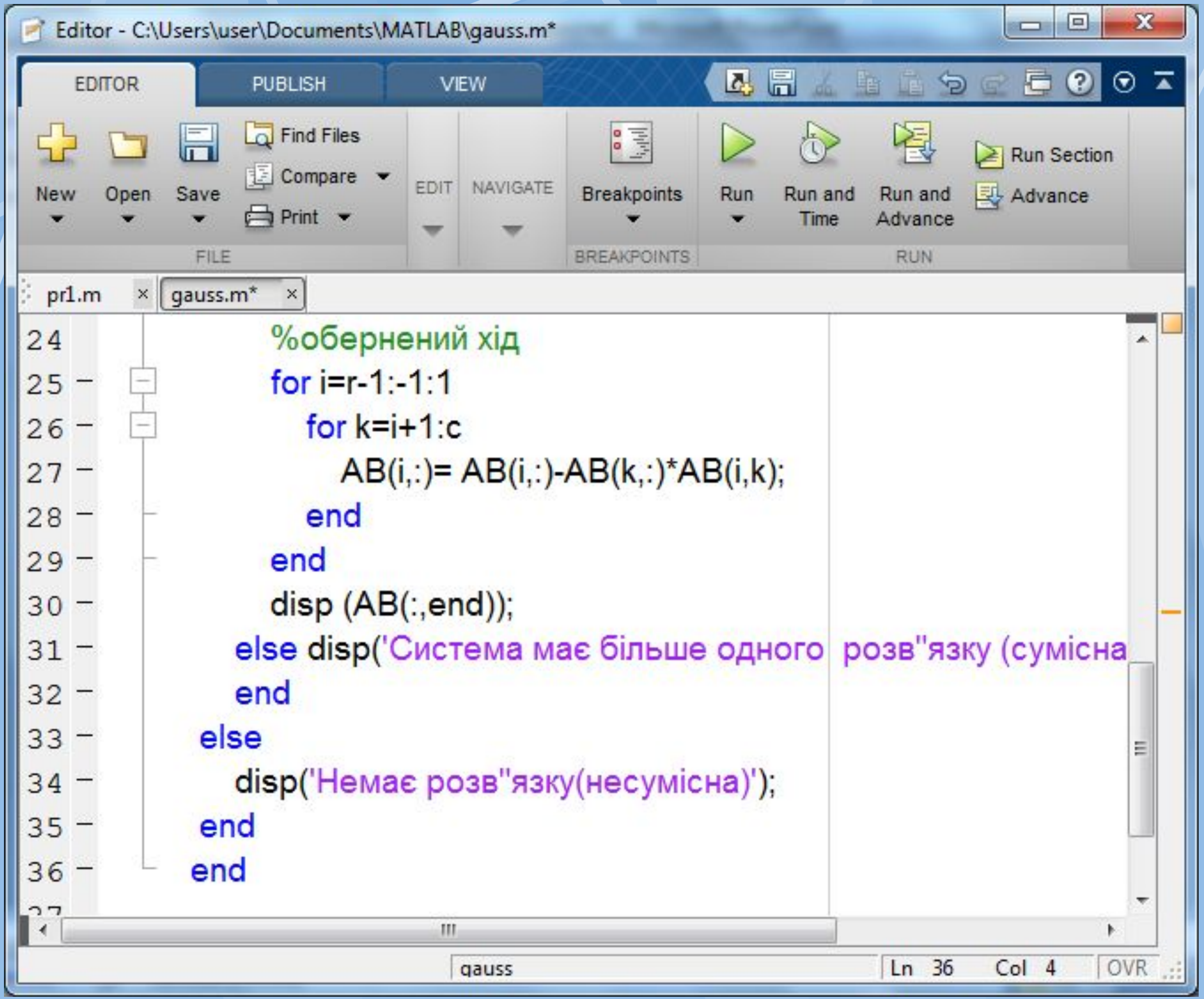
$$
\begin{aligned}
x_1 + 2x_2 + 3x_3 &= 14 \\
2x_1 - x_2 - 5x_3 &= -15 \\
x_1 - x_2 - x_3 &= -4
\end{aligned}
$$

Це можна зробити у такий спосіб:

```
» A = [ 1 2 3; 2 -1 -5; 1 -1 -1]
A =
    1    2    3
    2   -1   -5
    1   -1   -1
» B = [ 14;-15;-4]
B =
   14
  -15
   -4
» x = A \ B
x =
    1
    2
    3
```


Editor - C:\Users\user\Documents\MATLAB\gauss.m*
EDITOR
PUBLISH
VIEW
New
Open
Save
Find Files
Compare
Print
FILE
EDIT
NAVIGATE
Breakpoints
BREAKPOINTS
Run
Run and Time
Run and Advance
Run Section
Advance
RUN
pr1.m
gauss.m*
function gauss(A,B)
r=size(A,1);
c=size(A,2);
AB=[A,B];
if (rank(A)==rank(AB))
if (rank(A)==c)
disp('Один розв''язок (сумісна і визначена)');
%прямий хід
k=1;
while k<=r
if(AB(k,k)~=0)
gauss
Ln 2
Col 1
OVR


Editor - C:\Users\user\Documents\MATLAB\gauss.m*
EDITOR
PUBLISH
VIEW
New
Open
Save
Find Files
Compare
Print
FILE
EDIT
NAVIGATE
Breakpoints
BREAKPOINTS
Run
Run and Time
Run and Advance
Run Section
Advance
RUN
pr1.m
gauss.m*
if(AB(k,k)~=0)
AB(k,:)=AB(k,:)/AB(k,k);
for i=k+1:r
AB(i,:)=AB(i,:)-AB(k,:)*AB(i,k);
end
k=k+1;
else
temp=AB(k,:);
AB(k,:)=[];
AB=[AB;temp];
end
end
gauss
Ln 2
Col 1
OVR


Editor - C:\Users\user\Documents\MATLAB\gauss.m*
EDITOR
PUBLISH
VIEW
New
Open
Save
Find Files
Compare
Print
FILE
EDIT
NAVIGATE
Breakpoints
BREAKPOINTS
Run
Run and Time
Run and Advance
Run Section
Advance
RUN
pr1.m
gauss.m*
%обернений хід
for i=r-1:-1:1
for k=i+1:c
AB(i,:)= AB(i,:)-AB(k,:)*AB(i,k);
end
end
disp (AB(:,end));
else disp('Система має більше одного розв"язку (сумісна
end
else
disp('Немає розв"язку(несумісна)');
end
end
gauss
Ln 36
Col 4
OVR

```
syms x1 x2 x3
q1=2*x1+3*x2==5;
q2=0.3*x2+1.5*x3==1.6;
q3=x1+x2-15.8*x3==-5.7;

[x1 x2 x3]=solve(q1,q2,q3);
```

Особливості задач чисельного диференціювання і інтегрування функцій та основні методи їх розв' язання.

Приклади диференціювання у системі **MATLAB**.

Обчислення інтегралів та похідних у багатьох випадках досить рутинна операція при використанні пакетів аналітичних обчислень (маються на увазі пакети символьної математики). Разом з тим часто задача містить числові дані, тому для роботи необхідні ефективні обчислювальні процедури.

Як уже відомо, у системі **MATLAB** є підпакет, який призначений для аналітичних обчислень – це **Symbolic Math Toolbox** [44]. У цьому пакеті існує команда **diff**, яка обчислює скінченні різниці ($\Delta x_1 = x_2 - x_1, \ldots, \Delta x_{n-1} = x_n - x_{n-1}$). Синтаксис цієї команди:

- **diff(x) –** команда обчислює скінченні різниці, де **x** – одновимірний масив, а **diff(x) –**вектор різниць сусідніх елементів;
- **diff(x,n)** – команда обчислює скінченні різниці порядку **n**.

Приклад 1. Обчислити скінченні різниці незалежної та залежної змінних та перших похідних x, y, y', y'', якщо $y = \log x$.

Перелік команд для виконання цих операцій наданий нижче:

```
>> x=1:.1:2;
>> dx=diff(x);
>> y=log(x);
>> dy=diff(y);
>> d2y=diff(y,2);
```

Точне диференціювання функцій можна реалізувати за допомогою наступних команд:

- **diff(s)** – диференцює символьний вираз **s** по визначеній змінній;
- **diff(s,v)** – диференцює символьний вираз **s** по **v**;
- **diff (s,n)** або **diff(s,v,n) –** диференцює послідовно **n** разів символьний вираз **s.**

Приклад 2. Знайти першу та другу похідні функції $y = \frac{1}{2}\log(tg\frac{x}{2}) - \frac{1}{2}\frac{\cos x}{\sin^2 x}$.

Перелік команд для виконання вказаних операцій наведений нижче.

```
>> y=1/2*log(tan(x/2))–1/2*cos(x)/(sin(x))^2;
```

Знаходимо першу похідну:

```
>> Dy=diff(y)
Dy =
1/2*(1/2+1/2*tan(1/2*x)^2)/tan(1/2*x)+1/2/sin(x)+cos(x)^2/sin(x)^3
```

Знаходимо другу похідну:

```
>> D2y=diff(y,2)
D2y =
1/4+1/4*tan(1/2*x)^2–1/2*(1/2+1/2*tan(1/2*x)^2)^2/tan(1/2*x)^2–
5/2*cos(x)/sin(x)^2–3*cos(x)^3/sin(x)^4
```

Для отримання виразу вихідної функції виписуємо:

```
>> y=diff(y,0)
y =
1/2*log(tan(1/2*x))–1/2*cos(x)/sin(x)^2
```

Частинні похідні

Частинною похідною від функції двох змінних $z = f(x,y)$ за змінною x (позначається $z_x' = f_x'$ або $\frac{\partial z}{\partial x} = \frac{\partial f}{\partial x}$) називається границя відношення частинного прирісту функції $z = f(x,y)$ за змінною x до приросту цієї змінної Δx при умові $\Delta x \to 0$:

$$\frac{\partial z}{\partial x} = \lim_{\Delta x \to 0} \frac{\Delta_x z}{\Delta x} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x; y) - f(x; y)}{\Delta x},$$

похідна функції $z(x, y)$ по змінній x обчислена в припущенні, що інша змінна y фіксована [51].

Аналогічно визначається частинна похідна від функції $z = f(x,y)$ за змінною y (x вважається фіксованою):

$$\frac{\partial z}{\partial y} = \lim_{\Delta y \to 0} \frac{\Delta_y z}{\Delta y} = \lim_{\Delta y \to 0} \frac{f(x; y + \Delta y) - f(x,y)}{\Delta y}.$$

Приклад 1 Знайти частинні похідні функції $z = x^3 \sin y - y^2 \cos x$.

Визначимо символьні змінні та скористаємося командою **diff**:

```
>> syms x y z
>> z=sin(y)*x^3–y^2*cos(x);
```

Визначимо похідну по змінній **y**:

```
>> dy=diff(z,y)
dy =
cos(y)*x^3–2*y*cos(x)
```

Визначимо похідну по змінній **x**:

```
>> dx=diff(z,x)
dx =
3*sin(y)*x^2+y^2*sin(x)
```

Знаходження інтегралів від визначених функцій

З появленням вищевказаного підпакету **Simbolic Math** стало можливим знаходити аналітічне значення деяких інтегралів.

Приклад 1. Знайти інтеграл від функції: $y = \dfrac{1+\sin x}{1+\cos x} e^x$.

Спочатку визначимо символьні змінні, потім виконаємо операцію інтегрування за допомогою команди **int**:

```
>> syms x y
>> y=(1+sin(x))*exp(x)/(1+cos(x)); iy=int(y)
iy =
exp(x)*tan(1/2*x)
```

Приклад 2. Обчислити визначений інтеграл $\int\limits_0^{\ln 2} sh^4 dx$.

Ця задача виконується за допомогою команди i**nt**:

```
>> int((sinh(x))^4,0,log(2))
ans =
   0.0402
```

У більш складному випадку подвійних інтегралів значна увага приділяється визначенню області інтегрування.

Якщо область інтегрування σ обмежена кривими $y_1=\varphi_1(x)$, $y_2=\varphi_2(x)$, які кожна пряма, що паралельна вісі OY, перетинає не більш ніж у одній точці, то подвійний інтеграл по цій області може бути обчислений за допомогою процедури переходу до послідовного інтегрування:

$$\iint\limits_{(\sigma)} f(x,y)dxdy=\int\limits_a^b dx\int\limits_{\varphi_1(x)}^{\varphi_2(x)} f(x,y)dy.$$

Якщо область інтегрування σ обмежена кривими $x_1=\varphi_1(y)$, $x_2=\varphi_2(y)$, які кожна пряма, що паралельна вісі OX, перетинає не більш ніж у одній точці, то подвійний інтеграл по цій області може бути обчислений за допомогою наступної процедури переходу до послідовного інтегрування:

$$\iint\limits_{(\sigma)} f(x,y)dxdy=\int\limits_a^b dy\int\limits_{\varphi_1(y)}^{\varphi_2(y)} f(x,y)dx.$$

Приклад 3. Обчислити інтеграл $\int\limits_{-3}^{3} dy\int\limits_{y^2-4}^{5}(x+2y)dx$:

Спочатку обчислюємо внутрішній інтеграл:

```
in1=int(x+2*y,y^2–4,5)
in1 =
25/2–1/2*(y^2-4)^2+2*y*(9–y^2)
```

Обчислюємо зовнішній інтеграл:

```
in2=int(in1,–3,3)
in2 =
252/5
```

Формула трапецій

Відрізок $[a,b]$ розбивається на n частин з кроком $h = \frac{b-a}{n}$, при цьому точки розбиття x_i визначаються за формулою $x_i = a + i \times h$, $i = \overline{0,n}$, тобто $x_0 = a$, $x_n = b$.

Метод трапецій заснований на кусочно-лінійній інтерполяції функції $f(x)$, побудованій по точках $M_i = (x_i, f(x_i))$, $i = \overline{0,n}$ (рис. 9.1).

Оскільки площа S_i трапеції $x_i M_i M_{i+1} x_{i+1}$ на інтервалі (x_i, x_{i+1}) дорівнює

$$S_i = h\frac{f(x_i) + f(x_{i+1})}{2},$$

то, додаючи площі всіх прямолінійних трапецій, і отримуємо **формулу трапецій**

$$S_n = \sum_{i=0}^{n-1} S_i = h\sum_{i=0}^{n-1}\frac{f(x_i) + f(x_{i+1})}{2} = h\left[\frac{f(x_0) + f(x_n)}{2} + \sum_{i=1}^{n-1} f(x_i)\right]. \qquad (9.2)$$

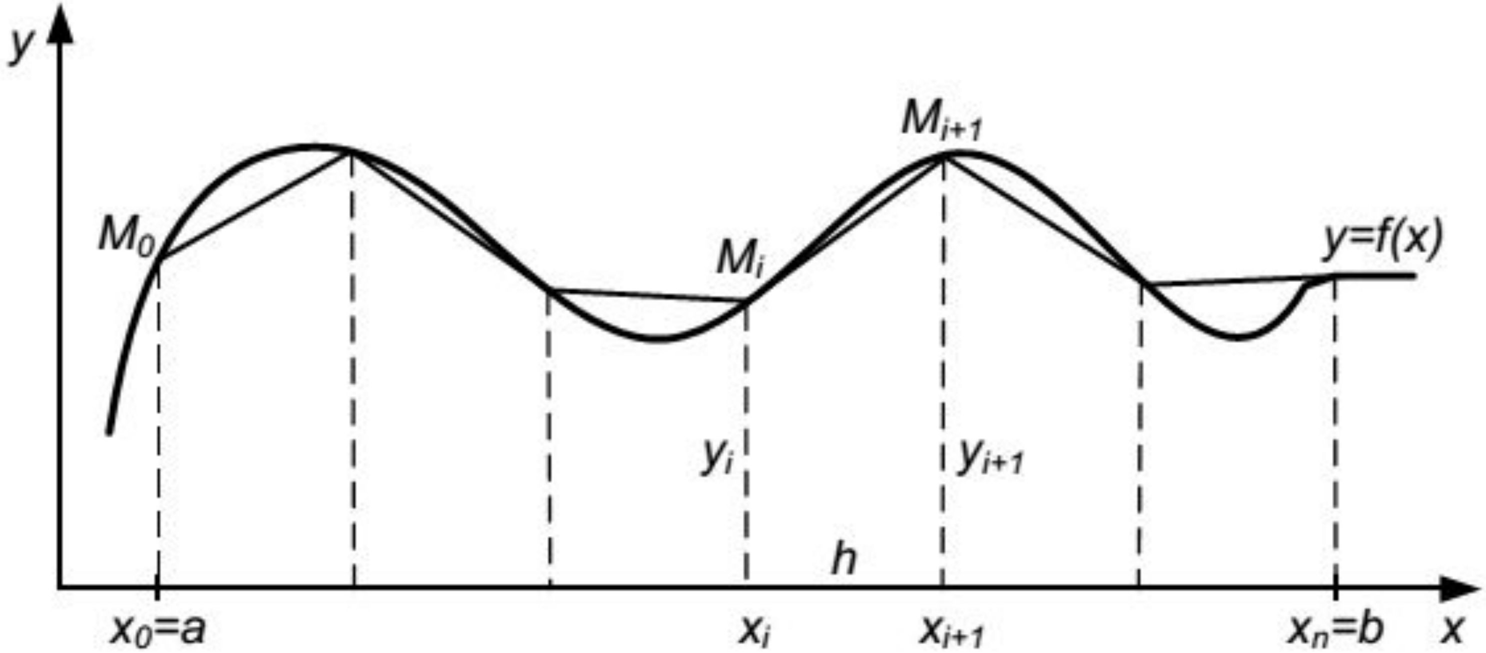


Рис. 9.1. **Графічна інтерпретація методу трапецій**

Оцінка погрішності формули трапецій [3; 5; 6]:

$$|S_n - I| \le \frac{b-a}{12} M_2 h^2, \tag{9.3}$$

де. $M_2 = \max\limits_{\xi \in (a,b)} |f''(\xi)|$.

Чисельне інтегрування у системі **MATLAB**.

У системі **MATLAB** реалізовані різні методи чисельного інтегрування функцій. Одним з найбільш розповсюджених методів наближеного обчислення визначених інтегралів є метод трапецій. Для його реалізації в системі **MATLAB** існує команда **trapz.**

Приклад 1. Обчислити визначений інтеграл $\int_{-\pi}^{\pi} \sin x^2 dx$ за допомогою методу трапецій.

Перелік команд процедури обчислення визначеного інтегралу за умови фіксованого кроку інтегрування $h = \dfrac{\pi}{10}$:

```
>> x=–pi:pi/10:pi;
>> y=sin(x.^2);
>> disp(trapz(x,y))
    1.4448
```

Якщо зменьшити крок інтегрування до $h = \dfrac{\pi}{1000}$, можна отримати більш точний результат:

```
>> x=–pi:pi/1000:pi;
>> y=sin(x.^2); disp(trapz(x,y))
    1.5453
```

Методи інтегрування більш високих порядків точності у системі **MATLAB** реалізовані за допомогою таких команд, як **quad** (метод Симпсона), **quadl** (метод Гаусса-Лобатто), **quad8** (метод Ньютона-Котеса 8-го порядку).

Загальне звертання до команди **quad**: **[s,n]=quad(f,a,b,tol,trace,...)**, де **s** – значення інтегралу, **n** – кількість обчислень підінтегральної функції, **f** – функція, **[a, b]** – інтервал інтегрування, **tol** – відносна похібка (за замовчуванням вона дорівнює **1e-3**), **trace** – при ненульовому значенні виводяться послідовність етапів обчислення інтегралу.

Приклад 2. Обчислити визначений інтеграл $\int_0^1 \frac{e^{-x}}{x^3+1}dx$.

Для завдання підінтегральної функції створюємо **m**-файл:

```
function f=int1(x)
f=exp(–x)./(x.^3+1);
```

Обчислюємо інтеграл за допомогою різних команд інтегрування **quad**, **quad8**, **quadl**:

```
>> format long
>> [in1,n]=quad('int1',0,1,1e-6,1)
     9    0.0000000000   2.71580000e-001    0.2367423251
    11    0.2715800000   4.56840000e-001    0.2475552724
    13    0.2715800000   2.28420000e-001    0.1468823197
    15    0.5000000000   2.28420000e-001    0.1006731351
    17    0.7284200000   2.71580000e-001    0.0704552012
in1 =
   0.55475298116279
n =
    17
>> [in1,n]=quad8('int1',0,1,1e-6,1)
in1 =
   0.55475295813348
n =
    33
>> [in1,n]=quadl('int1',0,1,1e-6,1)
    18    0.0000000000   5.00000000e-001    0.5547529353
in1 =
   0.55475293530993
n =
    18
```

Якість обчислення виявляється в результаті порівняння отриманих значень інтегралу з точним результатом.

Для наближеного обчислення послідовного інтегралу у системі **MATLAB** використовується команда, яка має наступний синтаксис:

dblquad(f,xmin,xmax,ymin,ymax),

де **f** – підінтегральна функція**, xmin,xmax,ymin,ymax –** інтервали інтегрування відповідно по змінним x та y.

Приклад 3. Обчислити послідовний інтеграл $\int\limits_{-\frac{\pi}{2}}^{0} dx \int\limits_{0}^{\frac{\pi}{2}} (x^2 - y^3) dy$.

Скористаємося вищенаведеною командою **dblquad**, для якої спочатку задаємо інтервали інтегрування:

```
>> f=inline('x.^2–y.^3');
>> disp(dblquad(f,–pi/2,0,0,pi/2))
  –0.3614
```

Чисельне інтегрування, диференціювання, мінімізація функцій і знаходження коренів в пакеті MATLAB

Назва	Значення
trapz(y)	чисельне інтегрування від дискретної функціїу
quad(‘f’,a,b)	чисельне інтегрування по методу Сімпсона для f(x), описаної в М-файлі f.m, де x Є [a,b]
quadl(‘f’,a,b)	чисельне інтегрування(підвищеної точності) по методу Сімпсона для f(x), описаної в М-файлі f.m, де x Є [a,b]
dblguad(‘f’,a,b,c,d)	чисельне інтегрування по методу Сімпсона для f(x,y), описаної в М-файлі f.m, де x є [a,b], а y є [c,d]
diff(x)	повертає значення x(2)-x(1),x(3)-x(2),...,x(n)-x(n-1)
fzero(f,x)	x-нуль функції f , наприклад: fzero(‘x.^3-27’,1)
fmin(‘f’,xn,xk)	Знаходження min функції однієї змінної на Інтервалі [xn,,xk]
fmins(‘f’,x0)	Знаходження min функцій декількох змінних в околі точки x0